

THEMATIC CYCLE 2:
SOFTWARE DEVELOPMENT

THEMATIC CYCLE 2: SOFTWARE DEVELOPMENT

TABLE OF CONTENTS

1. SDLC (software development life cycle) tutorial: what is, phases, model	4
What is Software Development Life Cycle (SDLC)?	4
A typical Software Development Life Cycle (SDLC) consists of the following phases:	4
Types of Software Development Life Cycle Models:	6
2. What is HTML?	7
HTML is the standard markup language for creating Web pages.	7
Example of a Simple HTML Document	7
Exercise: learn html using notepad or textedit	11
Example:	13
The <!DOCTYPE> Declaration explained	13
HTML Headings	13
Example:	13
HTML Paragraphs	14
Example:	14
HTML Links	14
Example:	14
HTML Images	14
Example:	14
How to View HTML Source?	14
Exercise: View HTML Source Code	14
Exercise: Inspect an HTML Element	15
3. Fundamentals of programming	15
Introduction to Computer Programming	16
What are Variables?	21
Creating variables	22
Example: C programming - ways of creating variables	22
Store Values in Variables	23
Example: storing values in variables	23
Access stored values in variables	24
Example: C program – how to print the values stored in its variables	24
Example: print both values using a single printf() statement	24
Variables in Java	25
Example: creating two variables a and b , assigning 10 and 20 in these variables and finally printing the values of the two variables in two ways.	25
Variables in Python	25
Example: creating two variables a and b and at the same time, assign 10 and 20 in those variables	26
Example: syntax in C and Java programming to declare variables and assign values at the same time	26
What is an Expression?	26
Examples:	26
Primary expressions	27
Example: Use this either with the dot or the bracket notation	27
Examples: validate function	27
Grouping operator	28

Example: How to use <i>new</i> -----	28
Example: Call parent constructor with Super keyword -----	29
Example: using a combination of push, splice, concat, etc. With spread syntax this becomes much more succinct -----	29
What are Loops and iterations? -----	29
Example of a loop (a for loop) in C#: -----	30
Example: simple C program to print multiple times -----	30
Example: same C program, but with the help of a while loop -----	31
The two most important loops available in C programming language -----	32
Example: using do...while loop, then Hello, World -----	33
The break statement -----	34
Example: the same above program, but it will come out after printing Hello World! only three times -----	34
The continue statement -----	35
Example: the same above program, but it will skip printing when the variable has a value equal to 3 -----	35
Loops in Java -----	36
Example: Java version of a program printing Hello, World! five times -----	36
Loops in Python -----	36
Example: Python version of a program printing Hello, World! five times -----	37
Data Types -----	37
Example: Adding data types -----	37
C and Java Data Types -----	38
Python Data Types -----	39
Examples/ exercises: Basic programing -----	39
Hello World Program in C -----	40
Hello World Program in Java -----	44
Hello World Program in Python -----	44
4. Best Software Development Tools -----	45
5. Best Code Coverage Tools, Review Tools, Code Editors -----	53
6. Programming Languages -----	75
Types of Programming Languages -----	75
The Difference Between Different Programming Languages -----	77
7. Competitive Programming for Beginners: Topcoder Challenges -----	79
What is Competitive Programming? -----	79
Best practices for competitive coding exams -----	83
A how to: code challenges at Topcoder -----	83
Exercise: Topcoder Learning Series - Flutter - Challenge 1 - Write your first app -----	84
8. Writing the Code – steps -----	87
Step 1: Choose The Right Languages -----	87
Step 2: Choose The Right Resources To Help You Learn -----	88
Step 3: Download A Code Editor -----	89
Step 4: Practice Writing Your Programs -----	89
Step 5: Join An Online Community -----	89
Step 6: Hack Someone Else's Code -----	90
Step 7: Keep Learning -----	90
9. Testing and Deployment -----	90
What is Deployment in SDLC? -----	90
The Three Phases of the Deployment Testing Cycle -----	91

10. Development of Small Systems	92
Analysis phase	93
Design phase	93
Implementation phase	94
Test and verification (t&v) phase	94
Maintenance phase	95
11. Database Development Life Cycle	96
Requirements Gathering	96
Analysis	97
Logical Design	97
Implementation	99
Realizing the Design	99
Populating the Database	99
12. System Integration	100
Definition - What does <i>System Integration (SI)</i> mean?	100
Methods of integration	100
Challenges of integration	101
13. Additional Exercises:	101
java Exercises:	102
HTML Exercises:	104
C# Exercises	106
Python Exercises	110

1. SDLC (SOFTWARE DEVELOPMENT LIFE CYCLE) TUTORIAL: WHAT IS, PHASES, MODEL

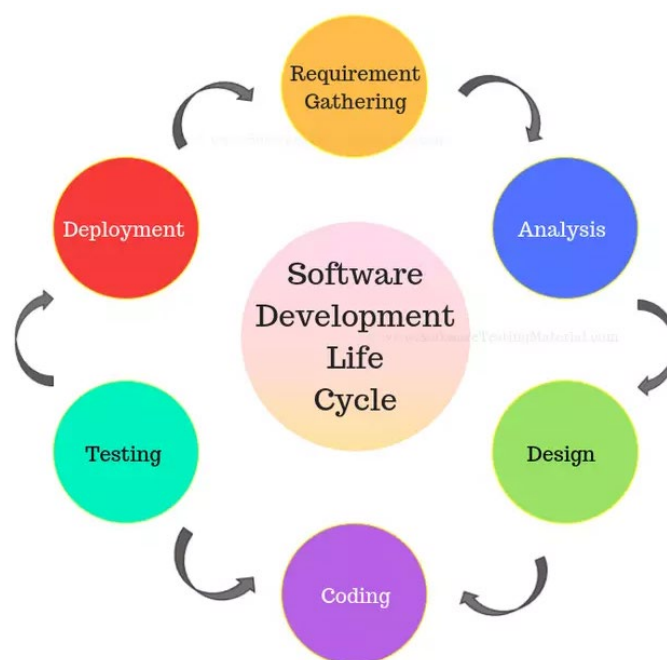
WHAT IS SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)?

Software Development Life Cycle (SDLC) aims to produce a high-quality system that meets or exceeds customer expectations, works effectively and efficiently in the current and planned information technology infrastructure, and is inexpensive to maintain and cost-effective to enhance.

DETAILED EXPLANATION: A process followed in software projects is SDLC. Each phase of SDLC produces deliverables required by the next phase in the life cycle. Requirements are translated into design. Code is produced according to the design. Testing should be done on a developed product based on the requirement. The deployment should be done once the testing was completed. It aims to produce a high-quality system that meets or exceeds customer expectations, works effectively and efficiently in the current and planned information technology infrastructure, and is inexpensive to maintain and cost-effective to enhance.

SDLC PROCESS: SDLC is a process which follows in Software Projects to develop a product in a systematic way and to deliver a high-quality product. By following proper SDLC process, Software companies can react well to the market pressure and release high-quality software. This process involves different stages of SDLC right from the requirement stage to deployment and maintenance phase.

A TYPICAL SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC) CONSISTS OF THE FOLLOWING PHASES:



REQUIREMENT PHASE:

Requirement gathering and analysis is the most important phase in the software development lifecycle. Business Analyst collects the requirement from the Customer/Client as per the client's business needs and documents the requirements in the Business Requirement Specification (document name varies depends upon the Organization. Some examples are Customer Requirement Specification (CRS), Business Specification (BS), etc., and provides the same to Development Team.

ANALYSIS PHASE:

Once the requirement gathering and analysis is done the next step is to define and document the product requirements and get them approved by the customer. This is done through the SRS (Software Requirement Specification) document. SRS consists of all the product requirements to be designed and developed during the project life cycle. Key people involved in this phase are Project Manager, Business Analyst and Senior members of the Team. The outcome of this phase is the Software Requirement Specification.

DESIGN PHASE:

It has two steps:

- 1) HLD – High-Level Design – It gives the architecture of the software product to be developed and is done by architects and senior developers
- 2) LLD – Low-Level Design – It is done by senior developers. It describes how each and every feature in the product should work and how every component should work. Here, only the design will be there and not the code

The outcome from this phase is High-Level Document and Low-Level Document which works as an input to the next phase.

DEVELOPMENT PHASE:

Developers of all levels (seniors, juniors, freshers) involved in this phase. This is the phase where we start building the software and start writing the code for the product. The outcome from this phase is Source Code Document (SCD) and the developed product.

TESTING PHASE:

When the software is ready, it is sent to the testing department where Test team tests it thoroughly for different defects. They either test the software manually or using automated testing tools depends on the process defined in STLC (Software Testing Life Cycle) and ensure that each and every component of the software works fine. Once the QA makes sure that the software is error-free, it goes to the next stage, which is Implementation. The outcome of this phase is the Quality Product and the Testing Artifacts.

DEPLOYMENT & MAINTENANCE PHASE:

After successful testing, the product is delivered/deployed to the customer for their use. Deployment is done by the Deployment/Implementation engineers. Once when the customers start using the developed system then the actual problems will come up and needs to be solved from time to time. Fixing the issues found by the customer comes in the maintenance phase. 100% testing is not possible – because, the way testers test the product is different from the way customers use the product. Maintenance should be done as per SLA (Service Level Agreement)

TYPES OF SOFTWARE DEVELOPMENT LIFE CYCLE MODELS:

There are various Software Development Life Cycle models in the industry which are followed during the software development process. These models are also referred to as Software Development Process Models.

Each SDLC model might have a different approach but the Software Development Phases and activities remain the same in all models.

Some of the Software Development Life Cycle Models (SDLC Models) followed in the industry are as follows:

1. WATERFALL MODEL:

Waterfall Model is a traditional model. It is aka Sequential Design Process, often used in SDLC, in which the progress is seen as flowing downwards like a waterfall, through the different phases such as Requirement Gathering, Feasibility Study/Analysis, Design, Coding, Testing, Installation, and Maintenance. Every next phase is begun only once the goal of the previous phase is completed. This methodology is preferred in projects where quality is more important as compared to schedule or cost. This methodology is best suitable for short term projects where the requirements will not change. (E.g. Calculator, Attendance Management)

2. SPIRAL:

Spiral model works in an iterative nature. It is a combination of both Prototype development process and Linear development process (waterfall model). This model place more emphasis on risk analysis. Mostly this model is adopted to the large and complicated projects where risk is high. Every Iteration starts with planning and ends with the product evaluation by client.

3. V MODEL:

V-model is also known as Verification and Validation (V&V) model. In this, each phase of SDLC must be completed before the next phase starts. It follows a sequential design process same like waterfall model.

4. PROTOTYPE:

The Prototype Model is one of the mostly used Software Development Life Cycle Models (SDLC models). A prototype of the end product is first developed prior to the actual product. Usually this SDLC model is used when the customers don't know the project requirements beforehand. By developing the prototype of the end product, it gives the customers an opportunity to see the product early in the life cycle.

It starts by getting the inputs (requirements) from the customers and undergoes developing the prototype. By getting the customers feedback, requirements are refined. Actual product development starts once the customer approves the prototype. The developed product is released for customer's feedback. Released product is refined as per the customers. This process goes on until the model is accepted by the customer.

5. AGILE:

Agile Scrum Methodology is one of the popular Agile software development methods. There are some other Agile software development methods but the popular one which is used widely is Agile Scrum Methodology. The Agile Scrum Methodology is a combination of both Incremental and Iterative model for managing product development.

OTHER RELATED SOFTWARE DEVELOPMENT LIFECYCLE MODELS ARE:

Agile Model, Rapid Application Development, Rational Unified Model, Hybrid Model etc.

2. WHAT IS HTML?

HTML IS THE STANDARD MARKUP LANGUAGE FOR CREATING WEB PAGES.

- HTML stands for Hyper Text Markup Language
- HTML is the standard markup language for creating Web pages
- HTML describes the structure of a Web page
- HTML consists of a series of elements
- HTML elements tell the browser how to display the content
- HTML elements label pieces of content such as "this is a heading", "this is a paragraph", "this is a link", etc.

EXAMPLE OF A SIMPLE HTML DOCUMENT

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>
```



```
<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Example Explained

- The `<!DOCTYPE html>` declaration defines that this document is an HTML5 document
- The `<html>` element is the root element of an HTML page
- The `<head>` element contains meta information about the HTML page
- The `<title>` element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab)
- The `<body>` element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The `<h1>` element defines a large heading
- The `<p>` element defines a paragraph

WHAT IS AN HTML ELEMENT?

An HTML element is defined by a start tag, some content, and an end tag:

```
<tagname>Content goes here...</tagname>
```

The HTML **element** is everything from the start tag to the end tag:

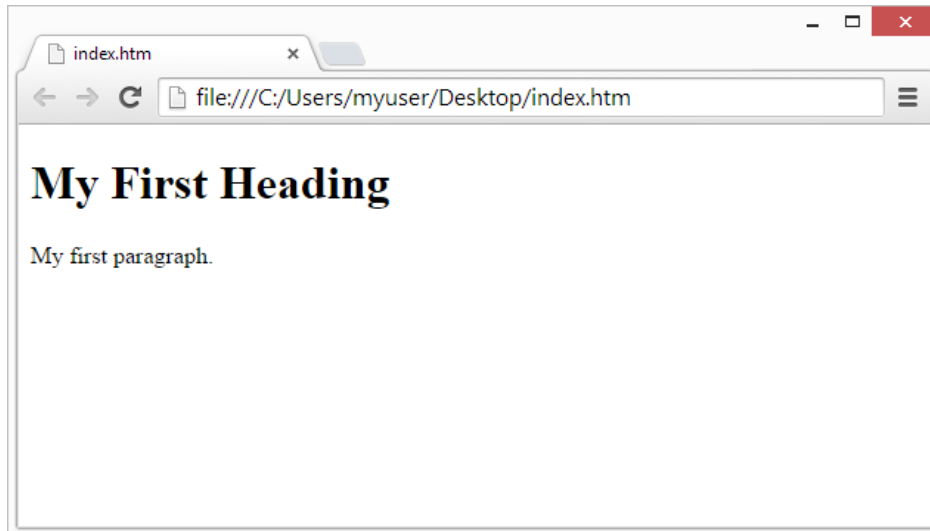
```
<h1>My First Heading</h1>
<p>My first paragraph.</p>
```

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
<p>	My first paragraph.	</p>
 	<i>none</i>	<i>none</i>

- **Note:** Some HTML elements have no content (like the `
` element). These elements are called empty elements. Empty elements do not have an end tag!

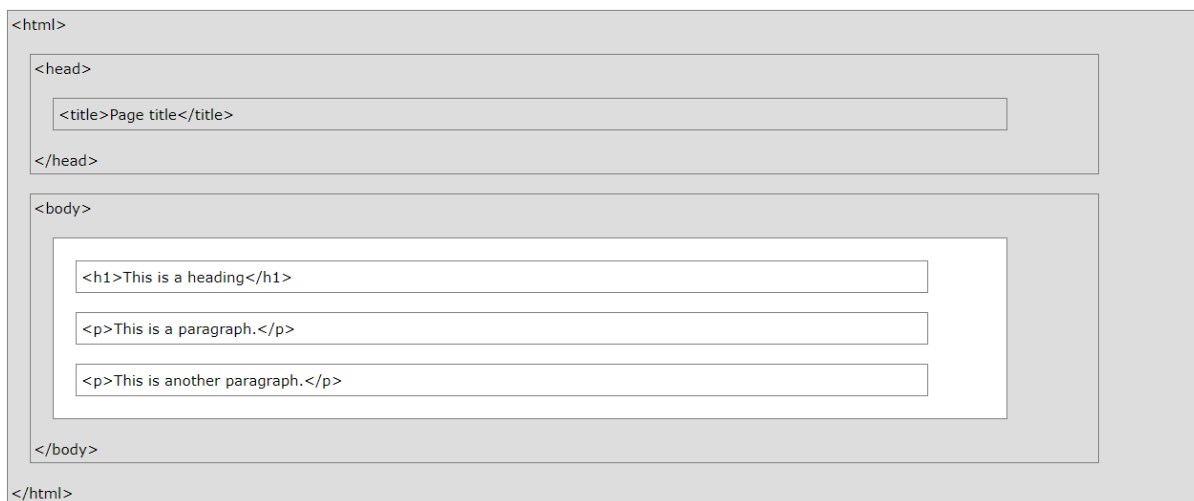
WEB BROWSERS

The purpose of a web browser (Chrome, Edge, Firefox, Safari) is to read HTML documents and display them correctly. A browser does not display the HTML tags, but uses them to determine how to display the document:



HTML PAGE STRUCTURE

Below is a visualization of an HTML page structure:



Note: Only the content inside the <body> section (the white area above) will be displayed in a browser.

HISTORY OF HTML

In the late 1980's, a physicist, Tim Berners-Lee who was a contractor at CERN, proposed a system for CERN researchers. In 1989, he wrote a memo proposing an internet-based hypertext system.

Tim Berners-Lee is known as the father of HTML. The first available description of HTML was a document called "HTML Tags" proposed by Tim in late 1991. The latest version of HTML is HTML5.

HTML VERSIONS

Since the time HTML was invented there are lots of HTML versions in market, the brief introduction about the HTML version is given below:

HTML 1.0: The first version of HTML was 1.0, which was the barebones version of HTML language, and it was released in 1991.

HTML 2.0: This was the next version which was released in 1995, and it was standard language version for website design. HTML 2.0 was able to support extra features such as form-based file upload, form elements such as text box, option button, etc.

HTML 3.2: HTML 3.2 version was published by W3C in early 1997. This version was capable of creating tables and providing support for extra options for form elements. It can also support a web page with complex mathematical equations. It became an official standard for any browser till January 1997. Today it is practically supported by most of the browsers.

HTML 4.01: HTML 4.01 version was released on December 1999, and it is a very stable version of HTML language. This version is the current official standard, and it provides added support for stylesheets (CSS) and scripting ability for various multimedia elements.

HTML5: HTML5 is the newest version of HyperText Markup language. The first draft of this version was announced in January 2008. There are two major organizations one is W3C (World Wide Web Consortium), and another one is WHATWG (Web Hypertext Application Technology Working Group) which are involved in the development of HTML 5 version, and still, it is under development.

FEATURES OF HTML

- 1) It is a very **easy and simple language**. It can be easily understood and modified.
- 2) It is very easy to make an **effective presentation** with HTML because it has a lot of formatting tags.
- 3) It is a **markup language**, so it provides a flexible way to design web pages along with the text.
- 4) It facilitates programmers to add a **link** on the web pages (by html anchor tag), so it enhances the interest of browsing of the user.
- 5) It is **platform-independent** because it can be displayed on any platform like Windows, Linux, and Macintosh, etc.
- 6) It facilitates the programmer to add **Graphics, Videos, and Sound** to the web pages which makes it more attractive and interactive.
- 7) HTML is a case-insensitive language, which means we can use tags either in lower-case or upper-case.

A simple text editor is all you need to learn HTML.

EXERCISE: LEARN HTML USING NOTEPAD OR TEXTEDIT

Web pages can be created and modified by using professional HTML editors. However, for learning HTML a simple text editor like Notepad (PC) or TextEdit (Mac) is enough.

Follow the steps below to create your first web page with Notepad or TextEdit.

STEP 1: OPEN NOTEPAD (PC)

Windows 8 or later: Open the Start Screen (the window symbol at the bottom left on your screen). Type Notepad.

Windows 7 or earlier: Open Start > Programs > Accessories > Notepad

STEP 1: OPEN TEXTEDIT (MAC)

Open Finder > Applications > TextEdit

Also change some preferences to get the application to save files correctly. In Preferences > Format > choose "Plain Text"

Then under "Open and Save", check the box that says "Display HTML files as HTML code instead of formatted text". Then open a new document to place the code.

STEP 2: WRITE SOME HTML

Write the following HTML code into Notepad:

```
<!DOCTYPE html>
<html>
<body>

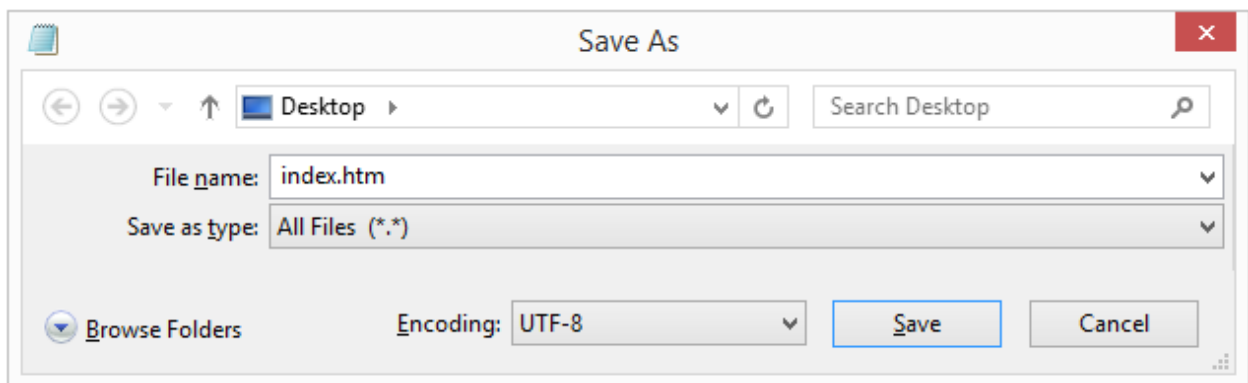
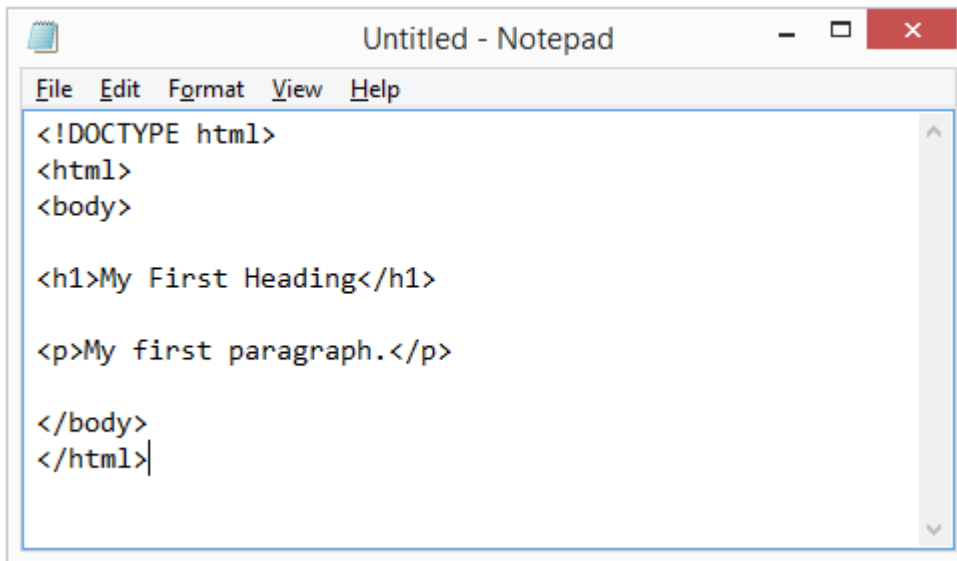
<h1>My First Heading</h1>

<p>My first paragraph.</p>

</body>
</html>
```

STEP 3: SAVE THE HTML PAGE

Save the file on your computer. Select **File > Save as** in the Notepad menu. Name the file "**index.htm**" and set the encoding to **UTF-8** (which is the preferred encoding for HTML files).

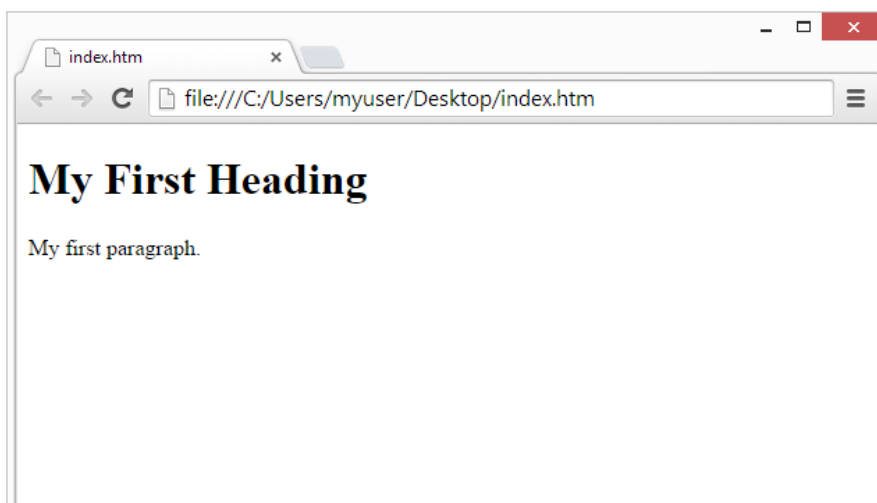


Tip: You can use either .htm or .html as file extension. There is no difference, it is up to you.

STEP 4: VIEW THE HTML PAGE IN YOUR BROWSER

Open the saved HTML file in your favorite browser (double click on the file, or right-click - and choose "Open with").

The result will look much like this:



HTML DOCUMENTS

All HTML documents must start with a document type declaration: `<!DOCTYPE html>`. The HTML document itself begins with `<html>` and ends with `</html>`. The visible part of the HTML document is between `<body>` and `</body>`.

EXAMPLE:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

THE `<!DOCTYPE>` DECLARATION EXPLAINED

The `<!DOCTYPE>` declaration represents the document type, and helps browsers to display web pages correctly. It must only appear once, at the top of the page (before any HTML tags). The `<!DOCTYPE>` declaration is not case sensitive.

The `<!DOCTYPE>` declaration for HTML5 is:

```
<!DOCTYPE html>
```

HTML HEADINGS

HTML headings are defined with the `<h1>` to `<h6>` tags. `<h1>` defines the most important heading. `<h6>` defines the least important heading:

EXAMPLE:

```
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
```

HTML PARAGRAPHS

HTML paragraphs are defined with the `<p>` tag:

EXAMPLE:

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

HTML LINKS

HTML links are defined with the `<a>` tag:

EXAMPLE:

```
<a href="https://www.w3schools.com">This is a link</a>
```

The link's destination is specified in the `href` attribute. Attributes are used to provide additional information about HTML elements.

HTML IMAGES

HTML images are defined with the `` tag. The source file (`src`), alternative text (`alt`), `width`, and `height` are provided as attributes:

EXAMPLE:

```

```

HOW TO VIEW HTML SOURCE?

Have you ever seen a Web page and wondered "Hey! How did they do that?"

EXERCISE: VIEW HTML SOURCE CODE

1) Right-click in an HTML page and select "View Page Source" (in Chrome) or "View Source" (in Edge), or similar in other browsers. This will open a window containing the HTML source code of the page.

EXERCISE: INSPECT AN HTML ELEMENT

1) Right-click on an element (or a blank area), and choose "Inspect" or "Inspect Element" to see what elements are made up of (you will see both the HTML and the CSS). You can also edit the HTML or CSS on-the-fly in the Elements or Styles panel that opens.

3. FUNDAMENTALS OF PROGRAMMING

A computer program is a sequence of instructions written using a Computer Programming Language to perform a specified task by the computer.

The two important terms that we have used in the above definition are –

- Sequence of instructions
- Computer Programming Language

To understand these terms, consider a situation when someone asks you about how to go to a nearby KFC. What exactly do you do to tell him the way to go to KFC?

You will use Human Language to tell the way to go to KFC, something as follows –

First go straight, after half kilometer, take left from the red light and then drive around one kilometer and you will find KFC at the right.

Here, you have used English Language to give several steps to be taken to reach KFC. If they are followed in the following sequence, then you will reach KFC –

1. Go straight
2. Drive half kilometer
3. Take left
4. Drive around one kilometer
5. Search for KFC at your right side

Now, try to map the situation with a computer program. The above sequence of instructions is actually a **Human Program** written in **English Language**, which instructs on how to reach KFC from a given starting point. This same sequence could have been given in Spanish, Hindi, Arabic, or any other human language, provided the person seeking direction knows any of these languages.

Now, let's go back and try to understand a computer program, which is a sequence of instructions written in a Computer Language to perform a specified task by the computer. Following is a simple program written in **Python** programming Language –

```
print "Hello, World!"
```

The above computer program instructs the computer to print "Hello, World!" on the computer screen.

- A computer program is also called a **computer software**, which can range from two lines to millions of lines of instructions.

- Computer program instructions are also called program source code and **computer programming** is also called **program coding**.
- A computer without a computer program is just a dump box; it is programs that make computers active.

As we have developed so many languages to communicate among ourselves, computer scientists have developed several computer-programming languages to provide instructions to the computer (i.e., to write computer programs). We will see several computer programming languages in the subsequent chapters.

INTRODUCTION TO COMPUTER PROGRAMMING

If you understood what a **computer program** is, then we will say: *the act of writing computer programs is called computer programming*.

As we mentioned earlier, there are hundreds of programming languages, which can be used to write computer programs and following are a few of them –

- Java
- C
- C++
- Python
- PHP
- Perl
- Ruby

USES OF COMPUTER PROGRAMS

Today computer programs are being used in almost every field, household, agriculture, medical, entertainment, defense, communication, etc. Listed below are a few applications of computer programs –

- MS Word, MS Excel, Adobe Photoshop, Internet Explorer, Chrome, etc., are examples of computer programs.
- Computer programs are being used to develop graphics and special effects in movie making.
- Computer programs are being used to perform Ultrasounds, X-Rays, and other medical examinations.
- Computer programs are being used in our mobile phones for SMS, Chat, and voice communication.

COMPUTER PROGRAMMER

Someone who can write computer programs or in other words, someone who can do computer programming is called a Computer Programmer.

Based on computer programming language expertise, we can name a computer programmer as follows –

- C Programmer
- C++ Programmer
- Java Programmer
- Python Programmer
- PHP Programmer
- Perl Programmer
- Ruby Programmer

BASICS

We assume you are well aware of English Language, which is a well-known **Human Interface Language**. English has a predefined grammar, which needs to be followed to write English statements in a correct way. Likewise, most of the Human Interface Languages (Hindi, English, Spanish, French, etc.) are made of several elements like verbs, nouns, adjectives, adverbs, propositions, and conjunctions, etc.

Similar to Human Interface Languages, Computer Programming Languages are also made of several elements. We will take you through the basics of those elements and make you comfortable to use them in various programming languages. These basic elements include:

- Programming Environment
- Basic Syntax
- Data Types
- Variables
- Keywords
- Basic Operators
- Decision Making
- Loops
- Numbers
- Characters
- Arrays
- Strings
- Functions
- File I/O

ENVIRONMENT SETUP

Environment Setup is not an element of any Programming Language, it is the first step to be followed before setting on to write a program.

When we say Environment Setup, it simply implies a base on top of which we can do our programming. Thus, we need to have the required software setup, i.e., installation on our PC which will be used to write computer programs, compile, and execute them. For example, if you need to browse Internet, then you need the following setup on your machine –

- A working Internet connection to connect to the Internet
- A Web browser such as Internet Explorer, Chrome, Safari, etc.

If you are a PC user, then you will recognize the following screenshot, which we have taken from the Internet Explorer while browsing [tutorialspoint.com](http://www.tutorialspoint.com).



Similarly, you will need the following setup to start with programming using any programming language.

- A text editor to create computer programs.
- A compiler to compile the programs into binary format.
- An interpreter to execute the programs directly.

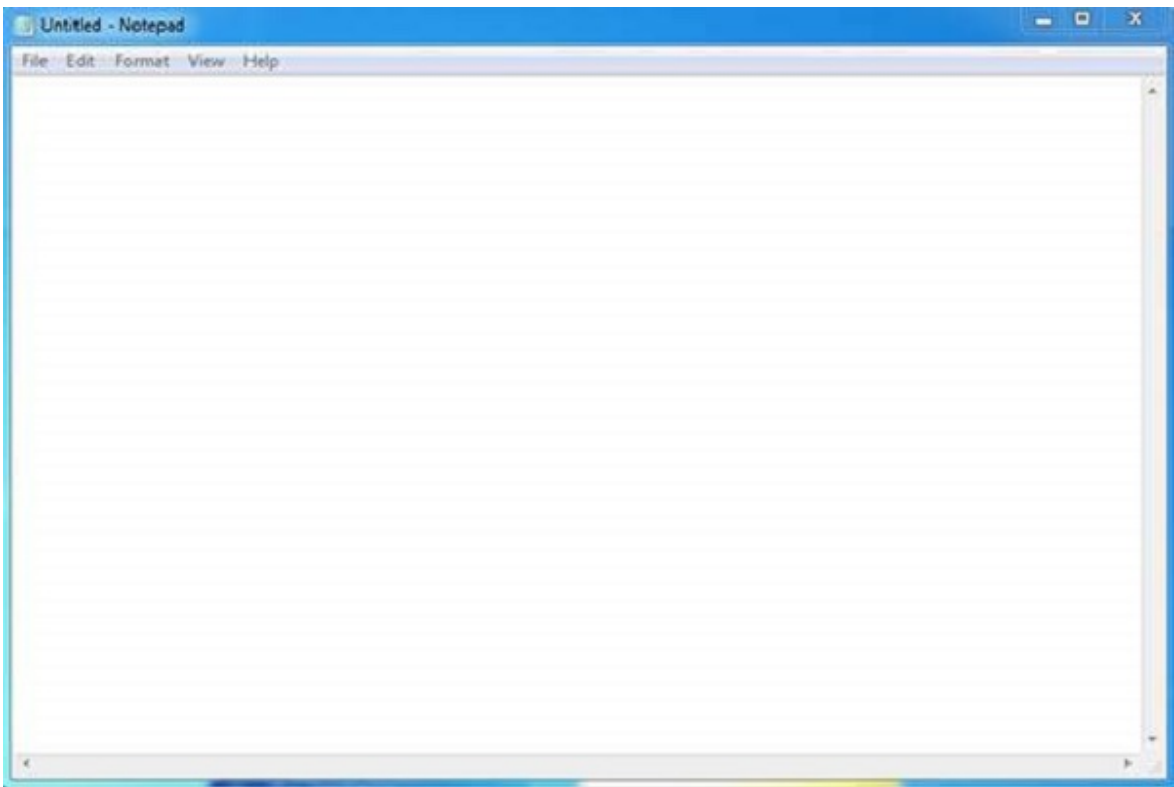
TEXT EDITOR

A text editor is a software that is used to write computer programs. Your Windows machine must have a Notepad, which can be used to type programs.

You can launch it by following these steps –

Start Icon → All Programs → Accessories → Notepad → Mouse Click on Notepad

It will launch Notepad with the following window –



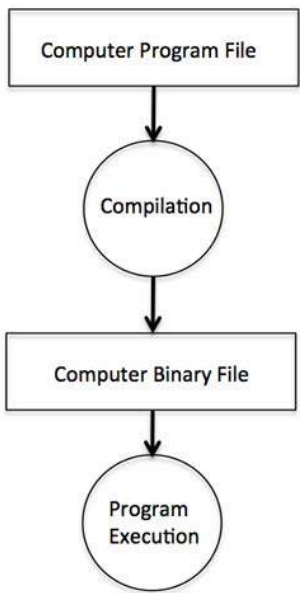
You can use this software to type your computer program and save it in a file at any location. You can download and install other good editors like **Notepad++**, which is freely available.

If you are a Mac user, then you will have **TextEdit** or you can install some other commercial editor like **BEdit** to start with.

COMPILER?

You write your computer program using your favorite programming language and save it in a text file called the program file.

Now let us try to get a little more detail on how the computer understands a program written by you using a programming language. Actually, the computer cannot understand your program directly given in the text format, so we need to convert this program in a binary format, which can be understood by the computer.

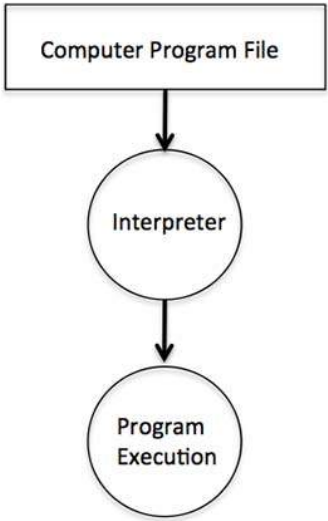


The conversion from text program to binary file is done by another software called Compiler and this process of conversion from text formatted program to binary format file is called program compilation. Finally, you can execute binary file to perform the programmed task.

The following flow diagram gives an illustration of the process –

So, if you are going to write your program in any such language, which needs compilation like C, C++, Java and Pascal, etc., then you will need to install their compilers before you start programming.

INTERPRETER



We just discussed about compilers and the compilation process. Compilers are required in case you are going to write your program in a programming language that needs to be compiled into binary format before its execution.

There are other programming languages such as Python, PHP, and Perl, which do not need any compilation into binary format, rather an interpreter can be used to read such programs line by line and execute them directly without any further conversion.

So, if you are going to write your programs in PHP, Python, Perl, Ruby, etc., then you will need to install their interpreters before you start programming.

ONLINE COMPILATION

If you are not able to set up any editor, compiler, or interpreter on your machine, then *tutorialspoint.com* provides a facility to compile and run almost all the programs online with an ease of a single click.

So do not worry and proceed further to have a thrilling experience to become a computer programmer in simple and easy steps.

APPLICATIONS OF COMPUTER PROGRAMMING LANGUAGES	
Python	Web and Internet Development, Scientific and Numeric applications, Desktop GUIs, Business applications. It is widely used in AI and Machine Learning space.
Java	Mostly used for developing Android apps, web apps, and Big data.
R	Data Science projects, Statistical computing, Machine learning
Javascript	JavaScript usage include web/mobile app development, game development, and desktop app development.
Swift	Swift is a specially designed language which works with Apple's Cocoa

	and Cocoa Touch frameworks to create all types of iOS apps.
C++	C++ is widely used in Game Development, Advance Computations, and Graphics Compilers
C#	Widely used in Enterprise Cross-Applications Development, Web Applications
PHP	Web Development, Content Management Systems, eCommerce Applications
SQL	Used in Any Database
Go	Console utilities, GUI applications, and web applications

ALGORITHM

From programming point of view, an **algorithm** is a step-by-step procedure to resolve any problem. An algorithm is an effective method expressed as a finite set of well-defined instructions.

Thus, a computer programmer lists down all the steps required to resolve a problem before writing the actual code. Following is a simple example of an algorithm to find out the largest number from a given list of numbers:

1. Get a list of numbers $L_1, L_2, L_3, \dots, L_N$
2. Assume L_1 is the largest, Largest = L_1
3. Take next number L_i from the list and do the following
4. If Largest is less than L_i
5. Largest = L_i
6. If L_i is last number from the list then
7. Print value stored in Largest and come out
8. Else repeat same process starting from step 3

The above algorithm has been written in a crude way to help beginners understand the concept. You will come across more standardized ways of writing computer algorithms as you move on to advanced levels of computer programming.

WHAT ARE VARIABLES?

Variables are the names you give to computer memory locations which are used to store values in a computer program.

For example, assume you want to store two values 10 and 20 in your program and at a later stage, you want to use these two values. Let's see how you will do it. Here are the following three simple steps –

- Create variables with appropriate names.
- Store your values in those two variables.
- Retrieve and use the stored values from the variables.

CREATING VARIABLES

Creating variables is also called **declaring variables** in C programming. Different programming languages have different ways of creating variables inside a program.

EXAMPLE: C PROGRAMMING - WAYS OF CREATING VARIABLES

```
#include <stdio.h>

int main() {
    int a;
    int b;
}
```

The above program creates two variables to reserve two memory locations with names **a** and **b**. We created these variables using **int** keyword to specify variable **data type** which means we want to store integer values in these two variables. Similarly, you can create variables to store **long**, **float**, **char** or any other data type.

EXAMPLE: CREATE VARIABLES TO STORE **LONG**, **FLOAT**, **CHAR** OR ANY OTHER DATA TYPE

```
/* variable to store long value */
long a;

/* variable to store float value */
float b;
```

EXAMPLE: CREATE VARIABLES OF SIMILAR TYPE BY PUTTING THEM IN A SINGLE LINE BUT SEPARATED BY COMMA AS FOLLOWS

```
#include <stdio.h>

int main() {
    int a, b;
}
```

Listed below are the key points about variables that you need to keep in mind –

- A variable name can hold a single type of value. For example, if variable **a** has been defined **int** type, then it can store only integer.

- C programming language requires a variable creation, i.e., declaration before its usage in your program. You cannot use a variable name in your program without creating it, though programming language like Python allows you to use a variable name without creating it.
- You can use a variable name only once inside your program. For example, if a variable **a** has been defined to store an integer value, then you cannot define **a** again to store any other type of value.
- There are programming languages like Python, PHP, Perl, etc., which do not want you to specify data type at the time of creating variables. So you can store integer, float, or long without specifying their data type.
- You can give any name to a variable like **age**, **sex**, **salary**, **year1990** or anything else you like to give, but most of the programming languages allow to use only limited characters in their variables names. For now, we will suggest to use only **a....z**, **A....Z**, **0....9** in your variable names and start their names using alphabets only instead of digits.
- Almost none of the programming languages allow to start their variable names with a digit, so **1990year** will not be a valid variable name whereas **year1990** or **ye1990ar** are valid variable names.

Every programming language provides more rules related to variables and you will learn them when you go in further detail of that programming language.

STORE VALUES IN VARIABLES

You have seen how we created variables in the previous section. Now, let's store some values in those variables

EXAMPLE: STORING VALUES IN VARIABLES

```
#include <stdio.h>

int main() {
    int a;
    int b;

    a = 10;
    b = 20;
}
```

The above program has two additional statements where we are storing 10 in variable **a** and 20 is being stored in variable **b**. Almost all the programming languages have similar way of storing values in variable where we keep variable name in the left hand side of an equal sign = and whatever value we want to store in the variable, we keep that value in the right hand side.

Now, we have completed two steps, first we created two variables and then we stored required values in those variables. Now variable **a** has value 10 and variable **b** has value 20. In other words, we can say, when above program is executed, the memory location named **a** will hold 10 and memory location **b** will hold 20.

ACCESS STORED VALUES IN VARIABLES

If we do not use the stored values in the variables, then there is no point in creating variables and storing values in them. We know that the above program has two variables **a** and **b** and they store the values 10 and 20, respectively. So let's try to print the values stored in these two variables.

EXAMPLE: C PROGRAM – HOW TO PRINT THE VALUES STORED IN ITS VARIABLES

```
#include <stdio.h>

int main() {
    int a;
    int b;

    a = 10;
    b = 20;

    printf( "Value of a = %d\n", a );
    printf( "Value of b = %d\n", b );
}
```

When the above program is executed, it produces the following result –

Value of a = 10

Value of b = 20

You must have seen **printf()** function in the previous chapter where we had used it to print "Hello, World!". This time, we are using it to print the values of variables. We are making use of **%d**, which will be replaced with the values of the given variable in **printf()** statements.

EXAMPLE: PRINT BOTH VALUES USING A SINGLE PRINTF() STATEMENT

```
#include <stdio.h>

int main() {
    int a;
    int b;

    a = 10;
```

```
b = 20;

printf( "Value of a = %d and value of b = %d\n", a, b );
}
```

When the above program is executed, it produces the following result –

Value of a = 10 and value of b = 20

If you want to use **float** variable in C programming, then you will have to use **%f** instead of **%d**, and if you want to print a character value, then you will have to use **%c**. Similarly, different data types can be printed using different % and characters.

VARIABLES IN JAVA

Following is the equivalent program written in Java programming language. This program will create two variables **a** and **b** and very similar to C programming, it will assign 10 and 20 in these variables and finally print the values of the two variables in two ways.

EXAMPLE: CREATING TWO VARIABLES **A** AND **B**, ASSIGNING 10 AND 20 IN THESE VARIABLES AND FINALLY PRINTING THE VALUES OF THE TWO VARIABLES IN TWO WAYS.

```
public class DemoJava {
    public static void main(String []args) {
        int a;
        int b;

        a = 10;
        b = 20;

        System.out.println("Value of a = " + a);
        System.out.println("Value of b = " + b);
        System.out.println("Value of a = " + a + " and value of b = " + b);
    }
}
```

When the above program is executed, it produces the following result –

Value of a = 10

Value of b = 20

Value of a = 10 and value of b = 20

VARIABLES IN PYTHON

Following is the equivalent program written in Python. This program will create two variables **a** and **b** and at the same time, assign 10 and 20 in those variables.

Python does not want you to specify the data type at the time of variable creation and there is no need to create variables in advance.

EXAMPLE: CREATING TWO VARIABLES **A** AND **B** AND AT THE SAME TIME, ASSIGN 10 AND 20 IN THOSE VARIABLES

```
a = 10
b = 20

print "Value of a = ", a
print "Value of b = ", b
print "Value of a = ", a, " and value of b = ", b
```

When the above program is executed, it produces the following result –

```
Value of a = 10
Value of b = 20
Value of a = 10 and value of b = 20
```

EXAMPLE: SYNTAX IN C AND JAVA PROGRAMMING TO DECLARE VARIABLES AND ASSIGN VALUES AT THE SAME TIME

```
#include <stdio.h>

int main() {
    int a = 10;
    int b = 20;

    printf( "Value of a = %d and value of b = %d\n", a, b );
}
```

When the above program is executed, it produces the following result –

```
Value of a = 10 and value of b = 20
```

WHAT IS AN EXPRESSION?

An *expression* is any valid unit of code that resolves to a value.

Every syntactically valid expression resolves to some value but conceptually, there are two types of expressions:

1. with side effects (for example: those that assign value to a variable)
2. those that in some sense evaluate and therefore resolve to a value

EXAMPLES:

The expression `x = 7` is an example of the first type. This expression uses the `=` *operator* to assign the value seven to the variable `x`. The expression itself evaluates to seven.

The code `3 + 4` is an example of the second expression type. This expression uses the `+` operator to add three and four together without assigning the result, seven, to a variable.

For example JavaScript has the following expression categories:

- Arithmetic: evaluates to a number, for example 3.14159. (Generally, uses arithmetic operators.)
- String: evaluates to a character string, for example, "Fred" or "234". (Generally, uses string operators.)
- Logical: evaluates to true or false. (Often involves logical operators.)
- Primary expressions: Basic keywords and general expressions in JavaScript.
- Left-hand-side expressions: Left values are the destination of an assignment.

PRIMARY EXPRESSIONS

Basic keywords and general expressions *in JavaScript*.

this

Use the *this* keyword to refer to the current object. In general, this refers to the calling object in a method.

EXAMPLE: USE **THIS** EITHER WITH THE DOT OR THE BRACKET NOTATION

```
this['propertyName']
```

```
this.propertyName
```

Suppose a function called `validate` validates an object's value property, given the object and the high and low values.

EXAMPLES: VALIDATE FUNCTION

```
function validate(obj, lowval, hival) {  
  if ((obj.value < lowval) || (obj.value > hival))  
    console.log('Invalid Value!');  
}
```

You could call `validate` in each form element's `onChange` event handler, using `this` to pass it to the form element, as in the following example:

```
<p>Enter a number between 18 and 99:</p>
```

```
<input type="text" name="age" size=3 onChange="validate(this, 18, 99);">
```

GROUPING OPERATOR

The grouping operator () controls the precedence of evaluation in expressions. For example, you can override multiplication and division first, then addition and subtraction to evaluate addition first.

```
var a = 1;
var b = 2;
var c = 3;

// default precedence
a + b * c // 7
// evaluated by default like this
a + (b * c) // 7

// now overriding precedence
// addition before multiplication
(a + b) * c // 9

// which is equivalent to
a * c + b * c // 9
```

LEFT-HAND-SIDE EXPRESSIONS

Left values are the destination of an assignment.

new

You can use the [new operator](#) to create an instance of a user-defined object type or of one of the built-in object types.

EXAMPLE: HOW TO USE NEW

```
var objectName = new objectType([param1, param2, ..., paramN]);
```

super

The `super` keyword is used to call functions on an object's parent. It is useful with *classes* to call the parent constructor, for example.

EXAMPLE: CALL PARENT CONSTRUCTOR WITH SUPER KEYWORD

```
super([arguments]); // calls the parent constructor.  
  
super.functionOnParent([arguments]);
```

Spread operator

The `spread` operator allows an expression to be expanded in places where multiple arguments (for function calls) or multiple elements (for array literals) are expected.

EXAMPLE: USING A COMBINATION OF PUSH, SPLICE, CONCAT, ETC. WITH SPREAD SYNTAX THIS BECOMES MUCH MORE SUCCINCT

Today if you have an array and want to create a new array with the existing one being part of it, the array literal syntax is no longer sufficient and you have to fall back to imperative code.

```
var parts = ['shoulders', 'knees'];  
var lyrics = ['head', ...parts, 'and', 'toes'];
```

Similarly, the `spread` operator works with function calls:

```
function f(x, y, z) { }  
var args = [0, 1, 2];  
f(...args);
```

WHAT ARE LOOPS AND ITERATIONS?

A **LOOP** repeatedly executes code in its body until the loop conditional statement becomes false.

ITERATION DEFINITION: When using a loop of any kind in programming, a single cycle of the loop, i.e. one *turn* of the loop, is known as an *iteration of the loop*.

A loop is divided into two parts:

- Loop Statement: This defines the time limit to be true for the continuous loop that is contingent on the attached conditional statement.
- Loop Body: This holds the statement's code or instruction; it is executed with each loop cycle.

EXAMPLE OF A LOOP (A FOR LOOP) IN C#:

```
int i;  
string numbers = "";  
for (i = 1; i <= 9; i++)  
    numbers += i.ToString();  
Console.WriteLine(numbers);
```

The output of the example is: "123456789."

Let's consider a situation when you want to print **Hello, World!** five times.

EXAMPLE: SIMPLE C PROGRAM TO PRINT MULTIPLE TIMES

```
#include <stdio.h>  
  
int main() {  
    printf( "Hello, World!\n");  
    printf( "Hello, World!\n");  
    printf( "Hello, World!\n");  
    printf( "Hello, World!\n");  
    printf( "Hello, World!\n");  
}
```

When the above program is executed, it produces the following result –

```
Hello, World!  
Hello, World!  
Hello, World!  
Hello, World!  
Hello, World!
```

It was simple, but again, let's consider another situation when you want to write **Hello, World!** a thousand times. We can certainly not write `printf()` statements a thousand times. Almost all the programming languages provide a concept called **loop**, which helps in executing one or more statements up to a desired number of times. All high-level programming languages provide various forms of loops, which can be used to execute one or more statements repeatedly.

EXAMPLE: SAME C PROGRAM, BUT WITH THE HELP OF A **WHILE LOOP**

```
#include <stdio.h>

int main() {
    int i = 0;

    while ( i < 5 ) {
        printf( "Hello, World!\n");
        i = i + 1;
    }
}
```

When the above program is executed, it produces the following result –

```
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
```

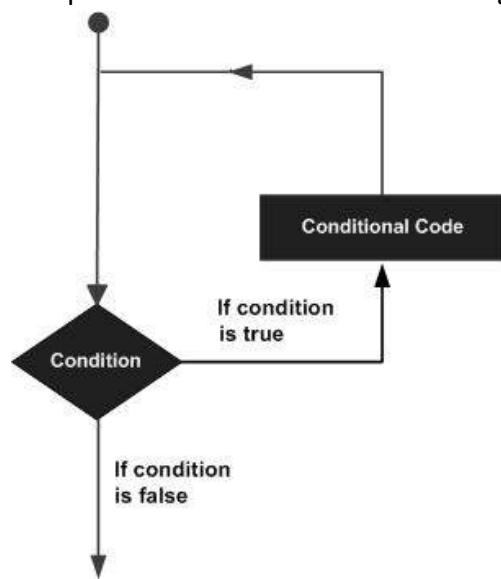
The above program makes use of a **while loop**, which is being used to execute a set of programming statements enclosed within {...}. Here, the computer first checks whether the given condition, i.e., variable "a" is less than 5 or not and if it finds the condition is true, then the loop body is entered to execute the given statements.

Here, we have the following two statements in the loop body –

- First statement is *printf()* function, which prints Hello World!
- Second statement is $i = i + 1$, which is used to increase the value of variable *i*

After executing all the statements given in the loop body, the computer goes back to while ($i < 5$) and the given condition, ($i < 5$), is checked again, and the loop is executed again if the condition holds true. This process repeats till the given condition remains true which means variable "a" has a value less than 5.

To conclude, a loop statement allows us to execute a statement or group of statements multiple times. Given below is the general form of a loop statement in most of the programming languages –



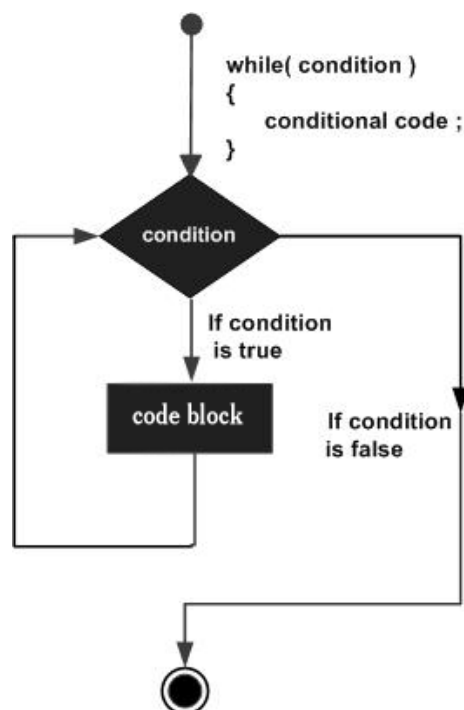
THE TWO MOST IMPORTANT LOOPS AVAILABLE IN C PROGRAMMING LANGUAGE

THE WHILE LOOP

A **while loop** available in C Programming language has the following syntax –

```
while ( condition ) {  
    /* ....while loop body .... */  
}
```

The above code can be represented in the form of a flow diagram as shown here –



The following important points are to be noted about a while loop –

- A while loop starts with a keyword **while** followed by a **condition** enclosed in ().
- Further to the while() statement, you will have the body of the loop enclosed in curly braces {...}.
- A while loop body can have one or more lines of source code to be executed repeatedly.
- If the body of a while loop has just one line, then its optional to use curly braces {...}.

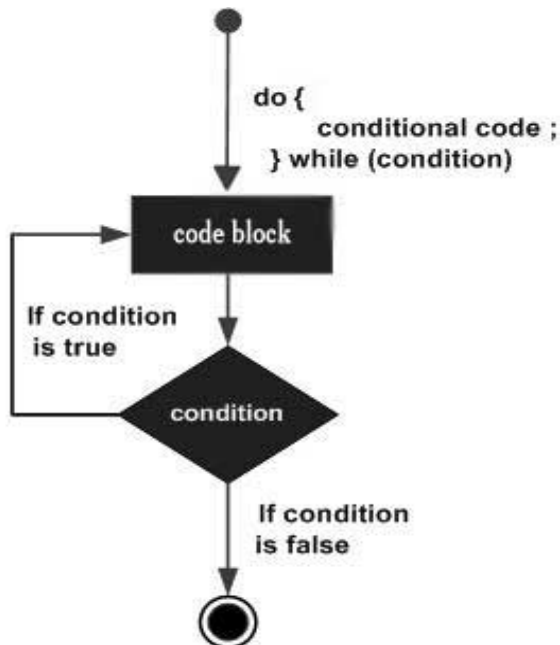
- A while loop keeps executing its body till a given **condition** holds true. As soon as the condition becomes false, the while loop comes out and continues executing from the immediate next statement after the while loop body.
- A condition is usually a relational statement, which is evaluated to either true or false. A value equal to zero is treated as false and any non-zero value works like true.

THE DO...WHILE LOOP

A while loop checks a given condition before it executes any statements given in the body part. C programming provides another form of loop, called **do...while** that allows to execute a loop body before checking a given condition. It has the following syntax –

```
do {
    /* ....do...while loop body .... */
}
while ( condition );
```

The above code can be represented in the form of a flow diagram as shown here



EXAMPLE: USING DO...WHILE LOOP, THEN HELLO, WORLD

```
#include <stdio.h>

int main() {
    int i = 0;

    do {
```

```

printf( "Hello, World!\n");
i = i + 1;
}
while ( i < 5 );
}

```

When the above program is executed, it produces the following result –

```

Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!

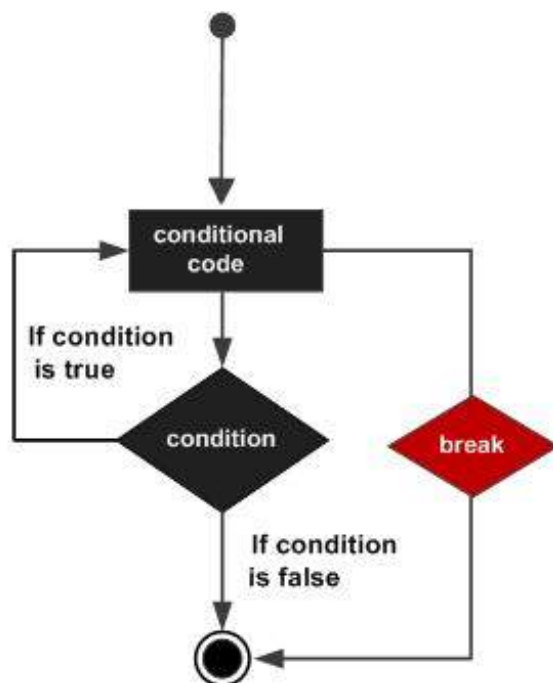
```

THE BREAK STATEMENT

When the **break** statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop. The syntax for a **break** statement in C is as follows –

```
break;
```

A **break** statement can be represented in the form of a flow diagram as shown below –



EXAMPLE: THE SAME ABOVE PROGRAM, BUT IT WILL COME OUT AFTER PRINTING HELLO WORLD! ONLY THREE TIMES

```

#include <stdio.h>

int main() {
    int i = 0;
    do {

```

```

printf( "Hello, World!\n");
i = i + 1;

if( i == 3 ) {
    break;
}
while ( i < 5 );
}

```

When the above program is executed, it produces the following result –

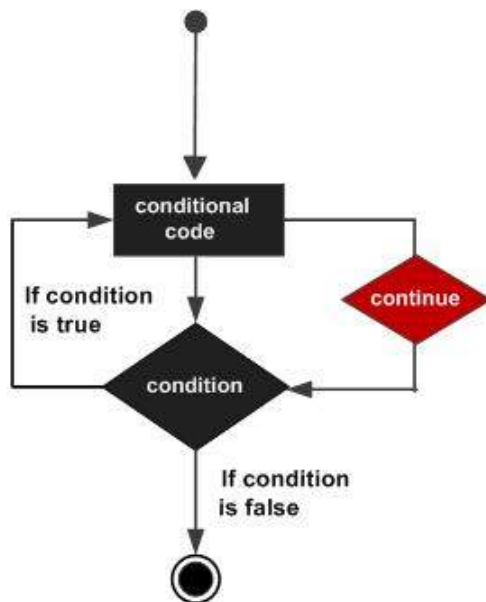
```

Hello, World!
Hello, World!
Hello, World!

```

THE CONTINUE STATEMENT

The **continue** statement in C programming language works somewhat like the **break** statement. Instead of forcing termination, **continue** forces the next iteration of the loop to take place, skipping any code in between. The syntax for a **continue** statement in C is as follows – `continue`; A **continue** statement can be represented in the form of a flow diagram as shown below –



EXAMPLE: THE SAME ABOVE PROGRAM, BUT IT WILL SKIP PRINTING WHEN THE VARIABLE HAS A VALUE EQUAL TO 3

```

#include <stdio.h>

int main() {
    int i = 0;
    do {

```

```

    if( i == 3 ) {
        i = i + 1;
        continue;
    }
    printf( "Hello, World!\n");
    i = i + 1;
}
while ( i < 5 );
}

```

When the above program is executed, it produces the following result –

```

Hello, World!
Hello, World!
Hello, World!
Hello, World!

```

LOOPS IN JAVA

Following is the equivalent program written in Java that too supports **while** and **do...while** loops. The following program prints **Hello, World!** five times as we did in the case of C Programming.

EXAMPLE: JAVA VERSION OF A PROGRAM PRINTING **HELLO, WORLD!** FIVE TIMES

```

public class DemoJava {
    public static void main(String []args) {
        int i = 0;

        while ( i < 5 ) {
            System.out.println("Hello, World!");
            i = i + 1;
        }
    }
}

```

You can try to execute the following program to see the output, which must be identical to the result generated by the above example.

The **break** and **continue** statements in Java programming work quite the same way as they work in C programming.

LOOPS IN PYTHON

Following is the equivalent program written in Python. Python too supports **while** and **do...while** loops. The following program prints **Hello, World!** five times as we did in case of C Programming. Here you must note that Python does not make use of

curly braces for the loop body, instead it simply identifies the body of the loop using indentation of the statements.

You can try to execute the following program to see the output. To show the difference, we have used one more print statement, which will be executed when the loop will be over.

EXAMPLE: PYTHON VERSION OF A PROGRAM PRINTING **HELLO, WORLD!** FIVE TIMES

```
i = 0

while (i < 5):
    print "Hello, World!"
    i = i + 1
print "Loop ends"
```

When the above program is executed, it produces the following result –

```
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Hello, World!
Loop ends
```

The **break** and **continue** statements in Python work quite the same way as they do in C programming.

DATA TYPES

Let's discuss a very simple but very important concept available in almost all the programming languages which is called **data types**. As its name indicates, a data type represents a type of the data which you can process using your computer program. It can be numeric, alphanumeric, decimal, etc.

Let's keep Computer Programming aside for a while and take an easy example of adding two whole numbers 10 & 20, which can be done simply as follows:

EXAMPLE: ADDING DATA TYPES

10 + 20

Let's take another problem where we want to add two decimal numbers 10.50 & 20.50, which will be written as follows –

10.50 + 20.50

The two examples are straightforward. Now let's take another example where we want to record student information in a notebook. Here we would like to record the following information –

Name:

Class:

Section:

Age:

Sex:

Now, let's put one student record as per the given requirement –

Name: Zara Ali

Class: 6th

Section: J

Age: 13

Sex: F

The first example dealt with whole numbers, the second example added two decimal numbers, whereas the third example is dealing with a mix of different data. Let's put it as follows –

- Student name "Zara Ali" is a sequence of characters which is also called a string.
- Student class "6th" has been represented by a mix of whole number and a string of two characters. Such a mix is called alphanumeric.
- Student section has been represented by a single character which is 'J'.
- Student age has been represented by a whole number which is 13.
- Student sex has been represented by a single character which is 'F'.

This way, we realized that in our day-to-day life, we deal with different types of data such as strings, characters, whole numbers (integers), and decimal numbers (floating point numbers).

Similarly, when we write a computer program to process different types of data, we need to specify its type clearly; otherwise the computer does not understand how different operations can be performed on that given data. Different programming languages use different keywords to specify different data types. For example, C and Java programming languages use **int** to specify integer data, whereas **char** specifies a character data type.

Subsequent chapters will show you how to use different data types in different situations. For now, let's check the important data types available in C, Java, and Python and the keywords we will use to specify those data types.

C AND JAVA DATA TYPES

C and Java support almost the same set of data types, though Java supports additional data types. For now, we are taking a few common data types supported by both the programming languages –

Type	Keyword	Value range which can be represented by this data type
Character	char	-128 to 127 or 0 to 255
Number	int	-32,768 to 32,767 or -2,147,483,648 to 2,147,483,647
Small Number	short	-32,768 to 32,767
Long Number	long	-2,147,483,648 to 2,147,483,647
Decimal Number	float	1.2E-38 to 3.4E+38 till 6 decimal places

These data types are called primitive data types and you can use these data types to build more complex data types, which are called user-defined data type, for example a string will be a sequence of characters.

PYTHON DATA TYPES

Python has five standard data types but this programming language does not make use of any keyword to specify a particular data type, rather Python is intelligent enough to understand a given data type automatically.

- Numbers
- String
- List
- Tuple
- Dictionary

Here, Number specifies all types of numbers including decimal numbers and string represents a sequence of characters with a length of 1 or more characters. For now, let's proceed with these two data types and skip List, Tuple, and Dictionary, which are advanced data types in Python.

EXAMPLES/ EXERCISES: BASIC PROGRAMING

Let's start with a little coding, which will really make you a computer programmer. We are going to write a single-line computer program to write **Hello, World!** on your screen. Let's see how it can be written using different programming languages.

HELLO WORLD PROGRAM IN C

Try the following example using a free online compiler option (www.compileonline.com)

Try to change the content inside `printf()`, i.e., type anything in place of **Hello World!** and then check its result. It just prints whatever you keep inside the two double quotes.

```
#include <stdio.h>

int main() {
    /* printf() function to write Hello, World! */
    printf( "Hello, World!" );
}
```

which produces the following result –

Hello, World!

This little Hello World program will help us understand various basic concepts related to C Programming.

PROGRAM ENTRY POINT

For now, just forget about the `#include <stdio.h>` statement, but keep a note that you have to put this statement at the top of a C program.

Every C program starts with `main()`, which is called the main function, and then it is followed by a left curly brace. The rest of the program instruction is written in between and finally a right curly brace ends the program.

The coding part inside these two curly braces is called the program body. The left curly brace can be in the same line as `main(){` or in the next line like it has been mentioned in the above program.

FUNCTIONS

Functions are small units of programs and they are used to carry out a specific task. For example, the above program makes use of two functions: **main()** and **printf()**. Here, the function `main()` provides the entry point for the program execution and the other function `printf()` is being used to print an information on the computer screen.

You can write your own functions which we will see in a separate chapter, but C programming itself provides various built-in functions like `main()`, `printf()`, etc., which we can use in our programs based on our requirement.

Some of the programming languages use the word **sub-routine** instead of function, but their functionality is more or less the same.

COMMENTS

A C program can have statements enclosed inside `/*.....*/`. Such statements are called comments and these comments are used to make the programs user friendly and easy to understand. The good thing about comments is that they are completely ignored by compilers and interpreters. So you can use whatever language you want to write your comments.

WHITESPACES

When we write a program using any programming language, we use various printable characters to prepare programming statements. These printable characters are **a, b, c,.....z, A, B, C,.....Z, 1, 2, 3,..... 0, !, @, #, \$, %, ^, &, *, (,), -, _ , +, =, \, |, {, }, [,], :, ;, <, >, ?, /, \, ~, ` , " , ' .** Hope I'm not missing any printable characters from your keyboard.

Apart from these characters, there are some characters which we use very frequently but they are invisible in your program and these characters are spaces, tabs (`\t`), new lines(`\n`). These characters are called **whitespaces**.

These three important whitespace characters are common in all the programming languages and they remain invisible in your text document –

Whitespace	Explanation	Representation
New Line	To create a new line	<code>\n</code>
Tab	To create a tab.	<code>\t</code>
Space	To create a space.	empty space

A line containing only whitespace, possibly with a comment, is known as a blank line, and a C compiler totally ignores it. Whitespace is the term used in C to describe blanks, tabs, newline characters, and comments. So you can write `printf("Hello, World!");` as shown below. Here all the created spaces around "Hello, World!" are useless and the compiler will ignore them at the time of compilation.

```
#include <stdio.h>

int main() {
```

```
/* printf() function to write Hello, World! */

printf( "Hello, World!" );

}
```

which produces the following result –

Hello, World!

If we make all these whitespace characters visible, then the above program will look like this and you will not be able to compile it –

```
#include <stdio.h>\n\n\nint main()\n{\n\n\n\t/* printf() function to write Hello, World! */\n\n\n\tprintf(\t"Hello, World!"\t);\n\n\n}\n
```

SEMICOLONS

Every individual statement in a C Program must be ended with a semicolon (;), for example, if you want to write "Hello, World!" twice, then it will be written as follows:

```
#include <stdio.h>

int main() {
    /* printf() function to write Hello, World! */
    printf( "Hello, World!\n" );
    printf( "Hello, World!" );
}
```

This program will produce the following result –

Hello, World!

Hello, World!

Here, we are using a new line character `\n` in the first `printf()` function to create a new line. Let us see what happens if we do not use this new line character –

```
#include <stdio.h>

int main() {
    /* printf() function to write Hello, World! */
    printf( "Hello, World!" );
    printf( "Hello, World!" );
}
```

This program will produce the following result –

Hello, World! Hello, World!

PROGRAM EXPLANATION

Let us understand how the above C program works. First of all, the above program is converted into a binary format using C compiler. So let's put this code in `test.c` file and compile it as follows –

```
$gcc test.c -o demo
```

If there is any grammatical error (Syntax errors in computer terminologies), then we fix it before converting it into binary format. If everything goes fine, then it produces a binary file called **demo**. Finally, we execute the produced binary demo as follows –

```
$/demo
```

which produces the following result –

Hello, World!

Here, when we execute the binary **a.out** file, the computer enters inside the program starting from `main()` and encounters a `printf()` statement. Keep a note that the line inside `/*....*/` is a comment and it is filtered at the time of compilation. So `printf()` function instructs the computer to print the given line at the computer screen. Finally, it encounters a right curly brace which indicates the end of `main()` function and exits the program.

SYNTAX ERROR

If you do not follow the rules defined by the programming language, then at the time of compilation, you will get syntax errors and the program will not be compiled. From syntax

point of view, even a single dot or comma or a single semicolon matters and you should take care of such small syntax as well. In the following example, we have skipped a semicolon, let's try to compile the program –

```
#include <stdio.h>
main() {
    printf("Hello, World!")
}
```

This program will produce the following result –

main.c: In function 'main':

main.c:7:1: error: expected ';' before '}' token

```
}
^
```

The bottom-line is: if you are not following proper syntax defined by the programming language in your program, then you will get syntax errors. Before attempting another compilation, you will need to fix them and then proceed.

HELLO WORLD PROGRAM IN JAVA

Following is the equivalent program written in Java. This program will also produce the same result **Hello, World!**.

```
public class HelloWorld {
    public static void main(String []args) {
        /* println() function to write Hello, World! */
        System.out.println("Hello, World!");
    }
}
```

which produces the following result –

Hello, World!

HELLO WORLD PROGRAM IN PYTHON

Following is the equivalent program written in Python. This program will also produce the same result **Hello, World!**.

```
# print function to write Hello, World! */
print "Hello, World!"
```

which produces the following result –

Hello, World!

Hope you noted that for C and Java examples, first we are compiling the programs and then executing the produced binaries, but in Python program, we are directly executing it. As we explained in the previous chapter, Python is an interpreted language and it does not need an intermediate step called compilation.

Python does not require a semicolon (;) to terminate a statement, rather a new line always means termination of the statement.

4. BEST SOFTWARE DEVELOPMENT TOOLS

Below is a review and comparison of the best software programming and development tools.

#1) LINX

Linx is low code tool to build and automate backend applications and web services. The tool accelerates the design, development and automation of custom business processes, including easy integration of applications, systems and databases.

- Easy-to-use, drag-and-drop IDE and Server
- Over 100 pre-built plugins programming functions and services for rapid development
- One-click deployment to any local or cloud server
- Input and outputs include nearly any SQL & NoSQL databases, numerous file formats (text and binary) or REST and SOAP Web services
- Live debugging with step through logic
- Automate processes via timer, directory events or message queue or expose web services, and call APIs via HTTP requests

#2) ATOM

Atom is an open source and free desktop editor cum source code editor that is up-to-date, friendly and hackable to the core.

Key Features:

- Atom supports cross-platform editing and works for various operating systems like Windows, Linux and OS X.
- Atom is a customizable tool with which one can effectively edit the look & feel of the User Interface, add few important features etc., without editing the configuration file.
- Important features of Atom which made it a remarkable tool are its built-in package manager, smart autocomplete, multiple panes, file system browser, find & replace feature etc.
- Atom is used to build cross-platform applications with web technologies using a framework called '**Electron**'.

#3) CLOUD 9

Initially in 2010 Cloud 9 was an open source, cloud-based IDE (Integrated Development Environment) that supports various programming languages like C, Perl, Python, JavaScript,

PHP etc. Later in 2016, AWS (Amazon Web Service) acquired it for further improvement and made it chargeable as per the usage.

Key Features:

- Cloud 9 IDE is a web-based platform that is used for scripting, running and debugging the code in the cloud.
- Using Cloud 9, the users can work with serverless applications which help to switch between remote and local testing and debugging activities.
- The features like code completion suggestions, debugging, file dragging etc., makes Cloud 9 a powerful tool.
- Cloud 9 is an IDE for web and mobile developers that help to collaborate together.
- Developers using AWS Cloud 9 can share the environment with the workmates for projects.
- Cloud 9 IDE lets to replica the entire development environment.

#4) GITHUB

GitHub is a powerful collaboration tool and development platform for code review and code management. With this GitHub, the users can build applications and software, manage the projects, host the code, review the code etc.

Key Features:

- With GitHub, developers can easily document their code and can host the same from the repositories.
- GitHub's project management tools help its users to stay aligned, co-ordinate easily and get their task done accordingly.
- Few features of GitHub that make it a useful tool are its code security, access control among the team members, integration with other tools etc.
- Few developers use GitHub for experimenting new programming languages in their personal projects.
- GitHub can be hosted on servers and on a cloud platform. It runs on Windows and Mac OS.
- GitHub is free for open source projects and public use. For developers it is charged @ \$7/month, for teams @ \$9/month and for organizations it is \$21/month.

#5) NETBEANS

NetBeans is an open source and a free software development tool written in Java that develops world-class web, mobile, and desktop applications easily and quickly. It uses C / C++, PHP, JavaScript, Java etc.

Key Features:

- NetBeans supports cross-platform and works on any operating system like Linux, Mac OS, Solaris, Windows etc.
- NetBeans offers features like Smart Code Editing, writing bug-free code, easy management process, and quick user interface development.

- Java applications can be easily updated to its newer editions using the code analyzers, editors and converters offered by NetBeans 8 IDE.
- Features of NetBeans IDE that made it the best tool are debugging, profiling, dedicated support from the community, powerful GUI builder, out of box working, support for Java platforms etc.
- The well-organized code in NetBeans allows its new developers to understand the structure of the application.

#6) BOOTSTRAP

Bootstrap is an open source and free framework for developing responsive websites and mobile-first projects using CSS, HTML, and JS. Bootstrap is widely used to design faster and simpler websites.

Key Features:

- As Bootstrap is an open source toolkit, one can customize it according to their project's requirement.
- Bootstrap is provided with built-in components which are used in accumulating responsive websites by a smart drag and drop facility.
- Powerful features of Bootstrap like a responsive grid system, plug-ins, pre-built components, sass variables & mixins allow its users to build their applications.
- Bootstrap is a front-end web framework that is used for quick modeling of the ideas and building of the web applications.
- This tool guarantees consistency among all the developers or users working on the project.

#7) NODE.JS

Node.js is an open source, cross-platform and JavaScript run-time environment that is built to design a variety of web applications and to create web servers and networking tools.

Key Features:

- Node.js applications run on Windows, Linux, Mac OS, Unix etc.
- Node.js is efficient and lightweight as it uses non-blocking and event-driven I/O model.
- Node.js is used by developers to write server-side applications in JavaScript.
- Node.js modules are used to provide rapid and well-organized solutions for developing back-end structure and integrating with the front-end platforms.
- The largest ecosystem of open source libraries is available with node.js package.
- Various IT Companies, software developers, small & large business organizations use node.js for developing web and network server applications in their projects.

#8) BITBUCKET

Bitbucket is a distributed, web-based version control system that is used for collaboration between software development teams (code and code review). It is used as a repository for source code and development projects.

Key Features:

- Useful features of Bitbucket that makes it a powerful tool are its flexible deployment models, unlimited private repositories, code collaboration on steroids etc.
- Bitbucket supports few services like code search, issue tracking, Git large file storage, bitbucket pipelines, integrations, smart mirroring etc.
- Using Bitbucket, one can organize the repositories into the projects with which they can focus easily on their goal, process or product.
- To rationalize the development process of any software it can integrate into the prevailing workflow.
- Bitbucket offers a free plan for 5 users with unlimited private repositories, standard plan @ \$2/user/month for growing teams and premium plan @ \$5/user/month for large teams.

#9) CODECHARGE STUDIO

CodeCharge Studio is the most creative and leading IDE and RAD (Rapid Application Development) that is used to create data-driven web applications or enterprise internet and intranet systems with minimal coding.

Key Features:

- CodeCharge Studio supports various platforms like Windows, Mac, Linux etc.
- Using CodeCharge Studio, one can analyze and modify the code generated to study the web technologies which are used to work with programming projects in any environment.
- It supports various Databases like MySQL, Postgre SQL, Oracle, MS Access, MS SQL etc.
- Few important features of CodeCharge Studio are Visual IDE & Code Generator, web reports, online calendar, gallery builder, flash charts, AJAX, menu builder, database-to-web converter etc.
- By using CodeCharge Studio, one can minimize the errors, reduce the development time, reduce the learning curve etc.
- CodeCharge Studio can be used for a 20-day free trial and then it can be purchased at \$139.95.

#10) CODELOBSTER

CodeLobster is a free as well as a convenient PHP IDE that is used to develop fully-featured web applications. It supports HTML, JavaScript, Smarty, Twig, and CSS.

Key Features:

- CodeLobster PHP Edition rationalizes & makes things easier in the development process and also supports CMS like Joomla, Magento, Drupal, WordPress etc.
- Few important and advanced features of CodeLobster PHP IDE are PHP Debugger, PHP Advanced autocomplete, CSS code inspector, DOM elements, auto-completing of keywords etc.
- PHP Debugger facilitates the users in debugging the programs at the time of coding and before executing the code.
- CodeLobster offers its users to enjoy the file explorer facilities and browser previews.

- CodeLobster is available in 3 versions namely free version, lite version and professional version.

#11) CODENVY

Codenvy is a cloud development environment used for coding and debugging the applications. It can support sharing projects in real-time and can collaborate with others.

Key Features:

- As Codenvy is a cloud-based IDE there is no need for any installation and configuration of this software development tool.
- Codenvy can be integrated with Jira, Jenkins, Eclipse Che extensions and to any private toolchain.
- Codenvy can be customized in many ways using IDE extensions, Eclipse Che, commands, stacks, editors, assemblies, RESTful APIs, and server-side extension plugins.
- Codenvy can run on any operating system like Windows, Mac OS, and Linux. It can also run in the public or private cloud.
- Command-line installers generated by Codenvy are used for deploying in any environment.
- It is available at a free of cost up to 3 developers and for more users, it has a monthly fee.

#12) ANGULARJS

AngularJS is an open source, structural and JavaScript based framework used by web developers to design web applications in a dynamic manner.

Key Features:

- AngularJS is fully expandable and works easily with other libraries. Each feature can be replaced or edited as per the development workflow and project needs.
- AngularJS works well with data-driven applications if the site is updated regularly as per the changes in the data.
- Advanced features of AngularJS are Directives, localization, dependency injection, reusable components, form validation, deep linking, data binding etc.
- AngularJS is not a plug-in or browser extension. It is 100% client-side and works on both mobile and desktop browsers like Safari, iOS, IE, Firefox, Chrome etc.
- AngularJS offers built-in protection against basic security holes which include HTML injection attacks and cross-site scripting.

#13) ECLIPSE

Eclipse is the most popular IDE that is used by Java developers in computer programming. It is used to develop applications not only in Java but also in other programming languages like C, C++, C#, PHP, ABAP etc.

Key Features:

- Eclipse is an open source group of projects, tools and collaborative working groups which play a key role in the development of new solution and innovations.
- Eclipse Software Development Kit (SDK) is a free and open source software which is used by the developers in programming as per their respective programming languages.
- Eclipse is used in creating web, desktop and cloud IDEs which in turn delivers the wide collection of add-on tools for software developers.
- Advantages of Eclipse are refactoring, code completion, syntax checking, rich client platform, error debugging, industrial level of development etc.
- One can easily integrate Eclipse with other frameworks like TestNG, JUnit, and other plug-ins.

#14) DREAMWEAVER

Adobe Dreamweaver is an exclusive software program and programming editor that is used for creating simple or complex websites. It supports many markup languages like CSS, XML, HTML, and JavaScript.

Key Features:

- Dreamweaver is used across Linux and Windows operating systems including iOS devices.
- Dreamweaver CS6 provides you with a preview option with which one can look at the preview of the designed website on any desired device.
- The latest version of Dreamweaver is used to design responsive websites.
- Another version of Dreamweaver, named Dreamweaver CC combines a code editor and a design surface termed as a Live view to offer some advanced features like auto-completion of code, code collapsing, real-time syntax checking, syntax highlighting and code inspection.
- Dreamweaver offers various plans, for individuals, for business and for schools or Universities.

#15) CRIMSON EDITOR

Crimson Editor is a freeware, lightweight text editing tool and an epic of software development tools only for Microsoft Windows which are used as HTML editor and source code editor.

Key Features:

- Crimson Editor is the specialized source code editor that offers an amazing feature of editing the score of programming languages like HTML, Perl, C / C++ and Java.
- Features of Crimson Editor include print & print preview, syntax highlighting, multi-level undo/redo, editing multiple documents, user tools & macros, editing remote files directly using built-in FTP client etc.

- Size of Crimson Editor software is also small but the loading time is fast.
- The learning curve of this software is so fast. It comes with a complete help manual which makes the navigation part easy.

#16) ZEND STUDIO

Zend Studio is a next-generation PHP IDE that is used for coding, debugging, prototyping and testing of mobile & web applications.

Key Features:

- Zend Studio's 3x faster performance helps in indexing, searching, and validation of PHP code.
- Zend Studio helps in deploying PHP applications on any server that includes cloud support for Microsoft Azure and Amazon AWS.
- Debugging capabilities offered by Zend Studio are using Z-Ray integration, Zend Debugger and Xdebug.
- It supports best-in-class development tools like Docker and Git Flow.
- Zend Studio works on Windows, Mac OS, and Linux platforms.
- Zend Studio software offers pricing for personal and for commercial use.

#17) JIRA

Jira is the most popular software development tool that is used by agile teams for planning, tracking and releasing the software.

Key Features:

- This tool is customizable and also has some prevailing features that are used in every development phase.
- Using Jira, we can accomplish the work in progress, generate reports, backlogs etc.
- Few other important features of Jira software are Scrum boards, Kanban boards, GitHub integration, Disaster recovery, Code Integration, Portfolio Management, Sprint Planning, Project Management etc.
- Jira works for Windows and Linux/Solaris operating systems.
- Jira software pricing in the cloud varies for small teams up to 10 users and for 11 – 100 users. For a free trial, this tool is available for 7 days.

#18) CLOUDFORGE

CloudForge is a SaaS (Software as a service) product that is used for application development. It is used for collaborative application development in the cloud.

Key Features:

- CloudForge is a secured and single cloud platform that is used by developers for coding, connecting and deploying the applications.
- CloudForge elastically balances your projects, teams, and processes.
- It is used to manage and integrate various development tools.
- Features of CloudForge are Version control hosting, Bugs & issue tracking, Agile planning, Visibility & reporting, deploying code to public & private clouds, etc.

- CloudForge is available for a 30 days free trial. Standard pack for small teams is available and Professional pack for small business & enterprise groups.

#19) AZURE

Microsoft Azure is a cloud computing service that is used for designing, deploying, testing and managing web applications or hybrid cloud applications through Microsoft's global network of data centers.

Key Features:

- Microsoft Azure offers various services like mobile services, data management, storage services, messaging, media services, CDN, caching, virtual network, business analytics, migrate apps & infrastructure etc.
- It supports various programming languages (.NET, Python, PHP, JavaScript etc), widest range of operating systems (Linux, Windows etc), devices and frameworks.
- Detailed pricing information is available on their website.
- Using Azure, we can easily spot the threats and lessen them, deliver the mobile apps flawlessly, manage the apps proactively etc.

#20) SPIRALOGICS APPLICATION ARCHITECTURE (SAA)

SAA is a cloud-based development tool that is used to define, design, customize, and publish their software applications online without any coding.

Key Features:

- Using SAA, the developers can preview the changes before issuing or deploying the applications.
- Even the users can select any pre-built application and customize them as per their requirement or can build it from scratch.
- Important features of SAA are drag & drop controls, customizing the controls, embed & built-in HTML editor, Interactive dashboard builder, predefined processes, a graphical representation of workflows & seamless integration etc.
- SAA supports various platforms like Windows, Android, Linux, iOS etc.
- SAA is available for a 30 days free trial.

#21) DELPHI

Embarcadero Delphi is a powerful object Pascal IDE that is used to develop cross-platform Native Applications with adjustable cloud services and comprehensive IoT connectivity.

Key Features:

- Delphi is used to deliver powerful and fast native apps for Linux, Android, iOS, Mac OS, Windows, IoT, and cloud.
- Delphi is 5 times quicker in designing Hyper-Connected Apps using FireUI preview for multiple database platforms, desktops, and mobiles.
- Delphi supports RAD and its features like native cross-compilation, visual window layout, application framework, refactoring etc.

- Delphi provides an integrated debugger, source control, strong database, code editor with code completion, real-time error-checking, in-line documentation, best code quality, code collaboration etc.
- Latest versions of Delphi includes features like Quick Edit support, new VCL control, FireMonkey, installer, multi-tenancy support in RAD server etc.

#22) ZOHO CREATOR

Zoho Creator is a low-code platform that enables rapid development and delivery of web and mobile applications and assists to Build powerful enterprise software applications 10x faster. You no longer have to write endless lines of code to build an application.

It also provides key features like Artificial Intelligence, JavaScript, Cloud functions, third-party integrations, multi-language support, offline mobile access, integration with a payment gateway and more.

With over 4 million users worldwide and 60+ apps, our platform enhances business productivity. Zoho Creator is featured in Gartner Magic Quadrant for Enterprise Low-Code Application Platforms (LCAP), 2019.

Features:

- Create more applications with less effort.
- Connect your business data and collaborate across teams.
- Create insightful reports.
- Gain instant access to mobile apps.
- Uncompromising security.

Verdict: Zoho Creator provides the low-code application development platform to build enterprise applications. It involves building applications with minimal coding which drastically reduces app-development time and effort.

5. BEST CODE COVERAGE TOOLS, REVIEW TOOLS, CODE EDITORS

BEST CODE COVERAGE TOOLS FOR JAVA, C, C++, C#, PYTHON

Code coverage is an important measure that quantifies the degree to which the source code of the program has been tested. There is a plethora of Code Coverage Tools in the market and selecting one for your project.

Following is a curated list of Top 13 handpicked Code Coverage tools with popular features and latest download links

1) COCO

Coco-platform is a multi-language code coverage tool. Automatic source code instrumentation helps you to measure test coverage of statements, branches, and conditions.

Features:

- Coco allows you to perform statement coverage, branch coverage, MC/DC and other levels
- Coco supports C, C++, C#, Tcl, and QML
- Helps you to merge multiple execution reports to provide advanced analysis

2) COBERTURA

Cobertura is a popular open source code coverage tool. It allows you to execute tasks via Maven and Ant, or the Cobertura CLI. You can embed with other multiple QA tools.

Features:

- Allows you to measure coverage without having the source code
- Helps you to find which parts of your Java program are lacking test coverage
- Allows you to represent reports in HTML and XML format
- Helps you to test lines and branches of the class & method

3) COVERAGE.PY

Coverage.py is another useful code coverage tool. It helps you to monitor Python programs, notes which are parts of the code have been executed.

Features:

- Coverage.py helps you to specify what source files you want it to analyze via the configuration file
- It also helps you to analyze the source to find out code that which could have been executed but was not.

4) JACOCO

JaCoCo is a free Java code coverage tool distributed under the Eclipse Public License. It is an open source free code coverage tools for Java, which has been made by the EclEmma.

Features:

- JaCoCo offers instructions, line and branch coverage
- It supports Java 7 and Java 8
- Helps you to test lines and branches of the class & method
- Offers easy to navigate HTML or XML report

5) OPENCLOVER

OpenClover tool helps you to measure code coverage for Java and Groovy and collects over 20 code metrics. It helps you to display untested areas of your application. It helps you to combine coverage and metrics to find the riskiest code.

Features:

- Helps you to run your test faster
- Allows you to focus what's necessary for your test
- Both branch and statement coverage support
- Allows you to generate XML-based report that, combined with ReportGenerator, produces TML-based report on coverage
- Helps you to keep the balance between application and tests

6) BULLSEYE COVERAGE

BullseyeCoverage is a code coverage software for C++ and C that tells you how much of your source code was tested. This tool allows you to perform unit testing, integration testing, and final release.

Features:

- Provide better coverage measurement
- Helps you to create more reliable code and save time
- Allows you to include or exclude any portion of the project code
- Merge results from distributed testing

7) NCOVER

NCover is an advanced level code coverage tool for .NET programs and applications. It provides support for statement coverage and branch coverage. This code coverage tool is available on open source and as well as on commercial license.

Features:

- .NET code coverage according to your customized needs
- Helps you to test, track and manage a unified coverage number across entire teams
- Detailed and centralized data about coverage
- Extensive documentation and user support
- This tool helps you to perform manual and coverage tests
- Deliver products to market faster and confidently in agile environments

8) VECTOR SOFTWARE

VectorCAST allows you to implement consistent processes for managing test activities and reporting key quality metrics. This software testing tool offers a cost-effective approach to unit testing, regression management, and code coverage analysis.

Features:

- Easy Test Collaboration
- System Test Automation
- Quality trend & change Impact Analysis
- Allows parallel Testing
- Web-based Quality Dashboard

9) DEVEL:COVER

Devel cover offers code coverage metrics for Perl. With the help of this code cover tool, you can discover areas of code not exercised by your tests. It helps you to tests to create to increase coverage.

Features:

- Offers some reports which include various types of HTML output, textual reports
- Statement, branch, condition, subroutine, and pod coverage information is reported

10) DOTCOVER

dotCover offers by JetBrains is a .NET unit test runner and code coverage tool. It allows you to integrate with Jet brains rider with Visual Studio. You can also calculate reports statement-level code coverage in applications for .NET, Silverlight or .NET Core.

Features:

- Allows you to visualize code coverage according to your business needs
- Provides a console utility for use with a Continuous Integration server
- Helps you to customize your coverage analysis with a coverage filter
- Allows you to use a shortcut to finds out which unit tests cover a particular statement

11) VISUAL STUDIO

Code coverage feature of Visual Studio helps you to determine what part of your project's code is tested by coded tests like unit tests. The tool allows you to view the previous set of results.

Features:

- Helps you to view a previous set of results
- Make results readable as text, select export code coverage results
- Allows you to merge results of several runs
- Allows you to export Code Coverage Results to save the results of a merge operation

12) ISTANBUL

Istanbul is a code coverage tool used for Javascript. It supports ES6/ES2015+ using babel-plugin. The tool offers all javascript instrumentation library which helps you to tracks statement, branch, and function coverage.

Features:

- Collection of reporters offers both terminal and HTML output
- Support for the most popular JavaScript testing frameworks
- Support for instrumenting subprocesses, using the NYC command-line-interface

CODE REVIEW TOOLS

Code Review tool automates the code audit process. They help in static code analysis which is essential to deliver a reliable software application. There is plethora of Code Review Tools in the market and selecting one for your project could be a challenge. Following is a curated list of Top 14 Code Review tools with popular features. The list contains open-source as well as commercial tools.

1) REVIEW ASSISTANT

Review Assistant is an extension to Visual Studio. It supports Visual Studio 2019, 2017, 2015, 2013, 2012, and 2010. Review Assistant helps to create review requests and respond to them without leaving IDE. It supports TFS, Subversion, Git, Mercurial, Perforce. Review Assistant adds the Code Review Board window to an IDE. The window is designed to manage all reviews available to a user.

Key features:

- Flexible code reviews
- Discussions in code
- Iterative review with defect fixing
- Email notifications
- Rich integration features
- Reporting and Statistics
- Drop-in Replacement for Visual Studio Code Review Feature.

2) RESHIFT

Reshift is a SaaS-based software platform that helps software development teams identify more vulnerabilities faster in their own code before deploying to production. Reducing the cost and time of finding and fixing vulnerabilities, identifying the potential risk of data breaches, and helping software companies achieve compliance and regulatory requirements.

Features:

- Integrates with Github and Bitbucket
- Offers security in the team's processes through the pull-request workflow and avoids switching to other dashboards
- Smart triaging that reduces false positives overtime with the labelling of issues
- Tracks vulnerabilities per developer feature branch
- Know critical vulnerabilities before merging into your master branch
- Gating the build if a new vulnerability is introduced

3) GERRIT

This open-source, lightweight tool, built over the "Git version control system,". This tool is useful in project environments where all users are trusted committers as this tool allows them to check overall changes made in the project.

Features:

- Gerrit stops the users from pushing directly to the Git repository
- Allows you to find the error in the source code
- Helps you create a new change or makes an update of an existing one
- Allows you to intermediate between developers and git repositories

4) CODESTRIKER

Codestriker is an open-source, online source code reviewing web application. This code review tool helps you to record issues, comments, and decisions in a database. It can also be used for code inspections.

Features:

- Support for traditional documents review
- It can be integrated with Bugzilla, ClearCase, CVS, etc.
- Codestriker tool is licensed under GPL

5) PHABRICATOR:

Phabricator is an open-source source code scanner. It also includes light-weight web-based code review, planning, testing, finding bugs, etc.

Features:

- Pre-Commit Code Review
- Keeps track of a large number of bugs
- Helps you to build unique task forms for each department
- Customizable Task Management
- Allows you to write helpful comments and anecdotes

6) CRUCIBLE:

Crucible is a web-based code quality tool. It is used by developers for code review, finding bugs and defects, discussing the changes and knowledge sharing. The tool helps them to catch major defects and improve their code architecture.

Features:

- Collaborate right code with inline comments, threaded mentions, and conversations
- Allows to track project in real time with activity streams showing the latest comments and review updates
- Ensure all files you are reviewing are current as the code is refactored and changed throughout the review process
- Automatically update Jira Software problems based on review activity and or turn review comments into issues with a single click
-

7) REVIEW BOARD

Review Board is a secure code review tool. It is used for code review and document review by open source projects and companies.

Features:

- Review Board can be integrated with ClearCase, Perforce, CVS, Plastic, etc
- The code is syntax highlighted which makes it more readable
- Support for pre-commit reviews and post-commit reviews

8) BARKEEP

Barkeep is a friendly code review system tool. It offers the easiest method to review code. It allows you to view commits made to any Git repository, see diffs, write comments.

Features:

- The tool allows you to emailed to your associate committers
- Supports post-commit workflows
- Offers a clean User Interface which is easily navigable

9) REVIEWABLE

Reviewable is a light-weight and powerful code analysis tool which makes the code review faster and thorough. It helps you to improve the code quality by cleaning the User Interface, finding bugs and highlighting the syntax.

Features:

- Fully customizable logic helps you to determine when a review is complete
- Works only with GitHub and GitHub Enterprise, making for seamless integration. Minimal admin work
- Line comments map across file revisions and stays in place until it is resolved

- Helps you to keep complete track of who reviews which revision of each file to make sure no changes are missed

10) PEER REVIEW PLUGIN

Peer review plugin eliminates the requirement for time-consuming code review meetings as it allows you to review code in a user-friendly web-based environment.

Features:

- Improved knowledge transfer
- Helps you to review files from the repository and comment on them
- Export data in MS Word docx format
- Better code and fewer defects
- Support for Git, SVN, and GitHub

11) CODACY

Codacy tool automatically identifies issues through static code analysis. Get a quick notification on security issues, code duplication, and code complexity in every pull and commit request.

Features:

- Helps you to identify new issues early in the process and prevent your product from being affected
- Get visibility into the quality of your code
- Seamlessly integrated into your workflow
- The self-hosted solution, packed with first class security on your servers

12) CODEFACTOR.IO:

Code factor tool allows you to have a glance of code quality for the whole project, recent commits, and the most problematic files. You can track and fixed issues for every commit & pull request.

Feature:

- Get an overview of your code base
- Seamless integration into your development process
- Allows you to control what should be analyzed
- Helps you to capture every line of code
- Streamline Code Review process and improves actionable reports
- It offers analytical data to understand, contribute and communicate with your peers

13) HELIX SWARM:

Helix swarm is a code review tool that arranges reviews, shares content and sees code review changes. It allows continuous integration deployment. It helps you to monitor progress, automate design process and improves release quality of the project.

Features:

- Allows you to do priority filtering
- Customize notification settings to your profile page
- Review independent components simultaneously by attaching multiple changelists to a pre-commit review
- Helps you to keep your code safe by integrating Helix Core with Ping Identity, Okta, and other tools

14) RHODECODE:

Rhodecode is an open source, secure enterprise source code management tool. The tool offers as an integrated tool for Git, Subversion, and Mercurial.

Features:

- Team Collaboration for Better Code Quality
- Rhodecode offers workflow Automation for Faster Collaboration
- Permission Management for Secure Software Development
- Helps you to Integrate an existing code base with new issue tracker tools

15) VERACODE:

Veracode is a code review and static analysis tool. It is built on the SaaS model. This tool allows you to analyze the code from a Security point of view. This tool uses binary code/bytecode and ensures 100% test coverage.

Features:

- Test desktop, web, or desktop apps of any size with consistent, processes and policies even when the source code is not available
- Test multiple applications without a manual and automate configuration
- Automate different workflows with streamlined and integrated testing in your SDLC
- Improve the productivity of your code with the continuous review process

16) JARCHITECT

JArchitect is easy to use code review tool for analyzing the Java code. After each review, it sends a report about the development of your project. The tool also helps you to improve code maintainability.

Features:

- JArchitect code rules are LINQ queries which can be generated within the second
- JArchitect helps you to find hundreds or even thousands of issues affecting a real-world code base
- The developer is immediately informed when new issues are found

WHAT IS A CODE EDITOR?

Code editors or source code editors are the software that is designed specifically to help developers in coding. These are actually the text editors with some additional functionalities to manage and edit code. It can be standalone or it can be a part of an IDE.

Using the best code editor can improve the speed of coding.

Code editors are programming language specific. Some editors support one or two programming languages whereas some support multiple programming languages. It can give suggestions and highlights only based on language support.

Structure editor is a type of coding editor or we can say that it is the functionality which is included in the editors. Structure editing is used for manipulating the structure of a code based on the syntax tree. The syntax tree is nothing but the structure of the code which is written in a programming language.

The code editors don't compile the code. It just allows you to write and edit the source code.

Functions: When developers write code using these editors, it takes care of syntax.

Code editors immediately warn for any syntax errors. Developers don't have to worry about syntax. Auto indentation & auto-completion saves a lot of time. Some editors like sublime text and visual studio code have an integrated terminal.

Core Features:

Enlisted below are the various Features of these Editors:

- Syntax highlighting
- Auto indentation
- Auto-completion
- Brace matching

How are Code Editors Different from IDE and Text Editors?

Code editors have more functionalities than the plain text editors. Plain text editors don't offer features like syntax highlighting and auto indentations. Also, Code editors are not IDE.

IDE includes debugging functionalities, code generators, and many other complex functionalities to help the developers, whereas code editors help developers in coding. According to programming languages, it highlights keywords and syntax errors.

Pros and Cons of Using These Editors:

The code editors are helpful if you are writing the code from scratch. But if you have to edit the existing code which is written by someone else then IDE is the best option. IDE is helpful for understanding the code which is written by others as code editors cannot compile and debug the code.

Some features of these editors are better than IDE like theme selection and searches, which are important while writing the code. Meanwhile, instead of editing few lines and debugging constantly, with code editors you can concentrate more on coding.

Another reason for using these editors instead of IDE is that IDE uses more resources like CPU, memory, and disk space. The coding editors don't use many resources, hence they are fast.

Points to consider while selecting the best editor for your project:

- Supported languages
- Supported operating systems or platform.
- Features
- Price

COMPARISON OF TOP SOURCE CODE EDITORS

Tool Name	Programming Languages	Operating Systems	Best Features	Cost	Written In
<u>Atom</u>	Supports many languages.	Windows,Linux, Mac OS	Cross-platform editing. Built-in package manager	Free	Built using web technologies
<u>Sublime Text</u>	Supports many programming languages.	Windows,Linux, Mac OS	Provides instant switching between projects. Cross platform support.	\$ 80	C++ & Python
<u>Notepad++</u>	PHP JavaScript	Windows,Linux, UNIX, Mac OS	Syntax Highlighting	Free	C++ And uses

Tool Name	Programming Languages	Operating Systems	Best Features	Cost	Written In
	HTML CSS	(Using third party tool)	Auto indentation Auto completion		Win 32 API & STL
<u>Brackets</u>	JavaScript HTML CSS	Windows,Linux, Mac OS	Live Preview Inline Editor	Free	JavaScript, HTML CSS
<u>Visual Studio Code</u>	Supports many languages like C++, Java, TypeScript, JSON and many more.	Windows,Linux, Mac OS	Auto-completion Debugging with breakpoints.	Free	TypeScript JavaScript CSS
<u>Vim</u>	Supports many programming languages.	Windows,Linux, UNIX, Mac OS, Android	Editing of compressed files Mouse interaction.	Free	C Vim Script
<u>Bluefish</u>	HTML, C, C++, Go, Java, JSP, and many more languages.	Cross-platform	Auto-completion. Code Navigation.	Free	C
<u>TextMate</u>	Supports many languages.	Mac OS	Auto-paring for brackets. & Can record Macros without programming.	Free	-
<u>Text Wrangler</u>	ANSI C, C++ Java, Ruby, PHP, Python, Perl, and many more.	Mac OS	Can split the editing windows. Multiple undo. Compares 2 text files.	\$49.99	-

Tool Name	Programming Languages	Operating Systems	Best Features	Cost	Written In
UltraEdit	HTML,PHP CSS C++ SAS code PL/SQL UNIX Shell Scripts Visual Basic	Windows,Linux, Mac OS	Integrated SSH, FTP, and Telnet. Multi-caret editing. Support editing in column mode also.	\$79.95 per year	

Following is a list of the best code editors for programmers. The list includes online editors for Windows and Mac users.

#1) ATOM



Atom, text and source code editor is developed by GitHub. It is an open source tool and the user can use it as an IDE.

Features:

- It has a built-in package manager.
- It provides smart options for auto-completion.
- You can open the tool in multiple panes for comparing.
- It supports cross-platform editing.
- It provides more features for finding & replacing text, and for searching files.

Pros:

- Easy to use.
- It provides some special developer tools.

- Easy to customize UI.

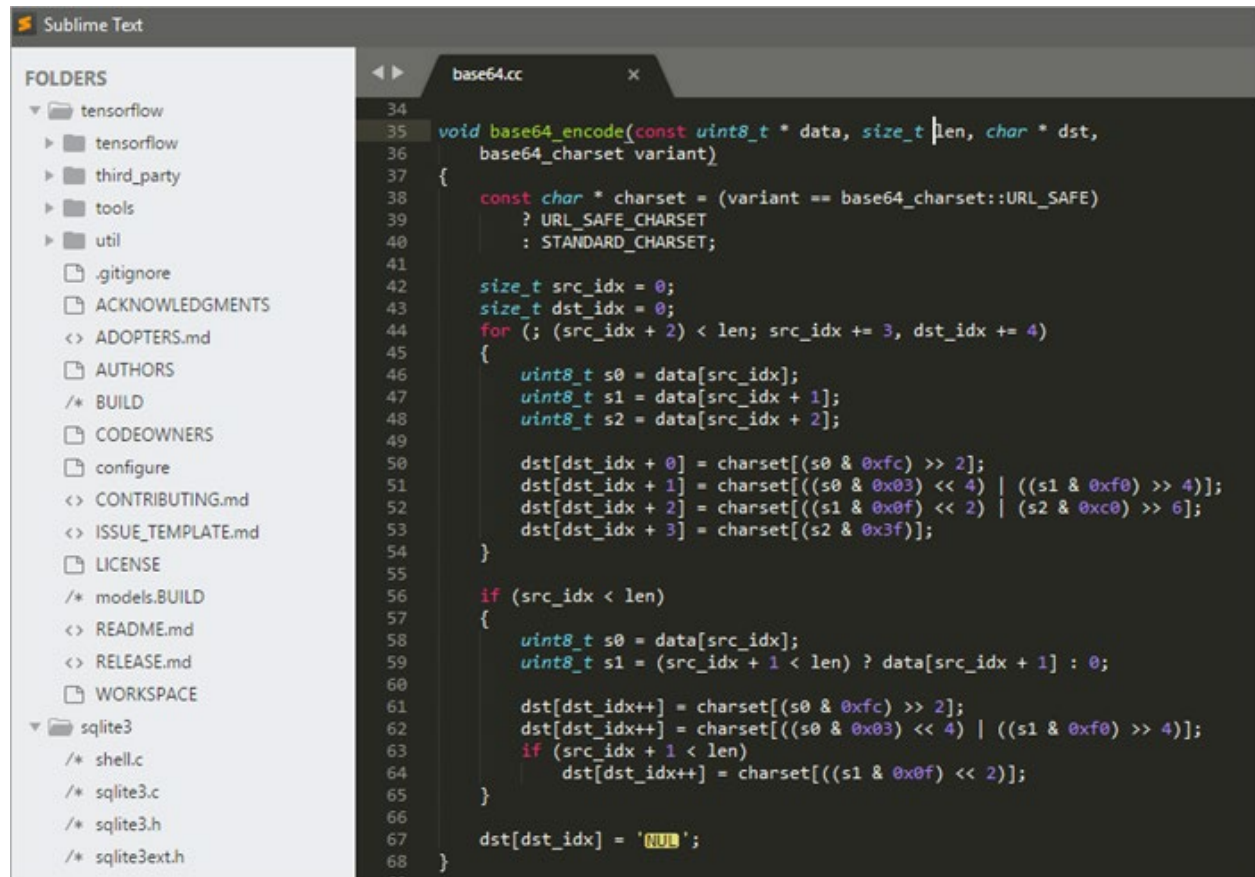
Cons:

- For writing packages, you will need a Coffee Script.

Tool Cost /Plan Details: Free

Findings: Atom is an open source tool. It has very good community support. You can use Atom for basic and advanced programming. It can be used as an IDE.

#2) SUBLINE TEXT



The sublime text editor is for Windows, Linux, and Mac.

Following is the list of supported languages:

ActionScript	AppleScript	ASP	C	C++
Python	YAML	TCL	PHP	Clojure
R	XSL	Shell Script	Lua	CSS
Rails	XML	Objective-C	HTML	Go

Scala	SQL	Makefile	JavaScript	Java
-------	-----	----------	------------	------

Features:

- It can detect and highlight syntax errors.
- For the whole project, it can automatically create an index for class, method, and function.
- Provides smart options for opening files, jumping to symbols etc.
- It allows you to edit multiple lines simultaneously.
- With the help of a JSON file, you can almost customize everything.

Cons:

- No package manager.
- Print option is not available

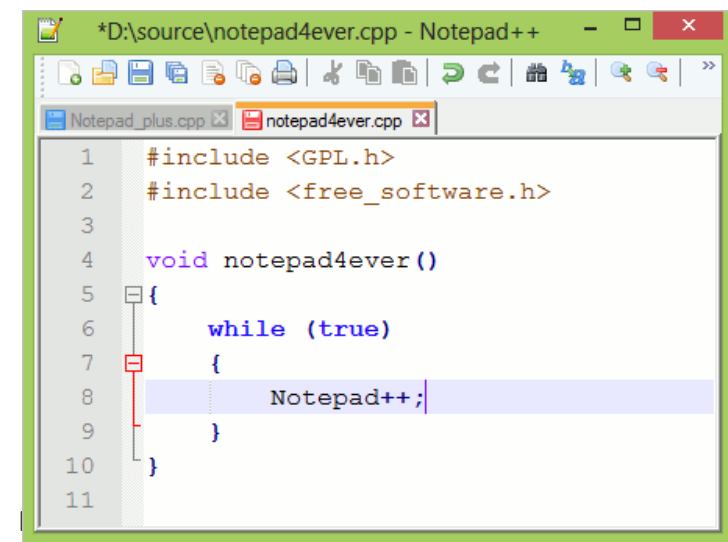
Pros:

- You can select and edit multiple lines.
- Easy to use for beginners.

Tool Cost /Plan Details: You have to buy the license for USD \$80. A free trial is also available.

Findings: Sublime Text is the best coding editor for beginners. Especially it is best for HTML and PHP. Apart from that, it supports many other programming languages as well.

#3) NOTEPAD++



and UNIX. It can also be used on Mac using a third party tool. Its latest available version is 7.5.8.

Features:

- It supports Macros recording and playback.
- For the ease of use, it provides many features like adding bookmarks, finding and replacing tasks, auto-completion, and syntax highlighting.
- It supports Multi-View, and Tab interface for Multi-Document.

Pros:

- Spelling check option is provided.
- Easy to use for beginners as well.
- Good community support from GitHub.

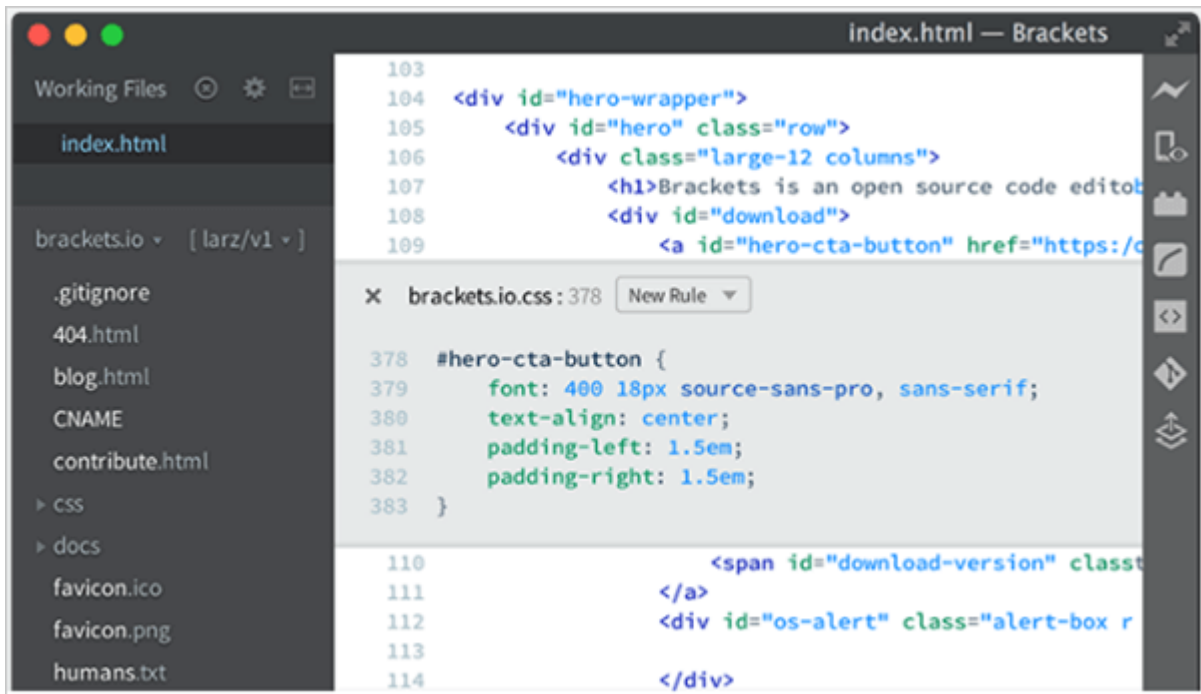
Cons:

- Remote file editing is not available for HTTP, SSH, and WebDAV.
- If you want to use Notepad++ on Mac, then you will have to use a third-party tool.

Tool Cost /Plan Details: Free

Findings: Notepad++ is a free code editor. It is used for coding in HTML, CSS, JavaScript, and PHP. Its code highlighting functionality helps in writing the code without error.

#4) BRACKETS



Brackets is a text editor for web designing or web development. It is an open source tool. Its latest release is 1.13. It can be used on Windows, Linux, and Mac OS.

Features:

- It is Extensible.
- It is lightweight and powerful.
- Brackets provide the feature of 'Live Preview', with the help of which you will be able to see the changes made to your CSS or HTML in your browser instantly.
- It provides Inline Editors.

Pros:

- Provides new features and bug fixes through regular updates.
- Easy to customize.
- You can preview the code which is going to establish the connection.

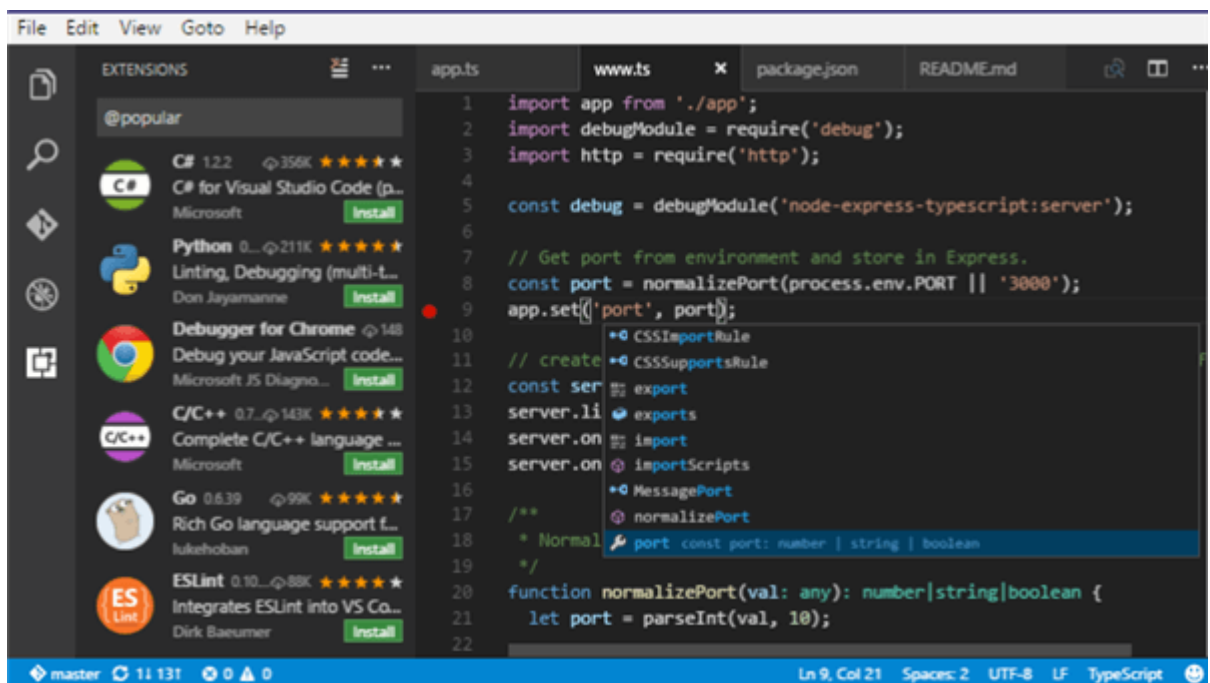
Cons:

- Auto-complete option is not available.
- Third party plugins are not supported.
- Full screen, split screen options are not supported.

Tool Cost /Plan Details: Free

Findings: It is an open source text editor for Mac. Brackets have good community support.

#5) VISUAL STUDIO CODE



Visual Studio Code is an open source tool. It can be used on Windows, Linux, and Mac and You can run it everywhere.

Visual Studio can be used for:

C#	R	Objective-C	Go
CSS	Visual Basic	SQL	Swift
F#	Power shell	Rust	JSON
XML	PHP	Java	HTML
Python	Jade	Razor	TypeScript
Lua	Ini	Groovy	Dockerfile
Batch	Sass	Less	Coffee Script
Perl	Clojure	Makefile	Markdown

Features:

- Syntax highlighting.

- Auto-completion for variable types, function definitions, and imported modules.
- Debugging with breakpoints, call stacks, and interactive console.
- Visual studio Code is extensible and customizable.

Pros:

- Lightweight tool.
- It doesn't take much space on RAM.

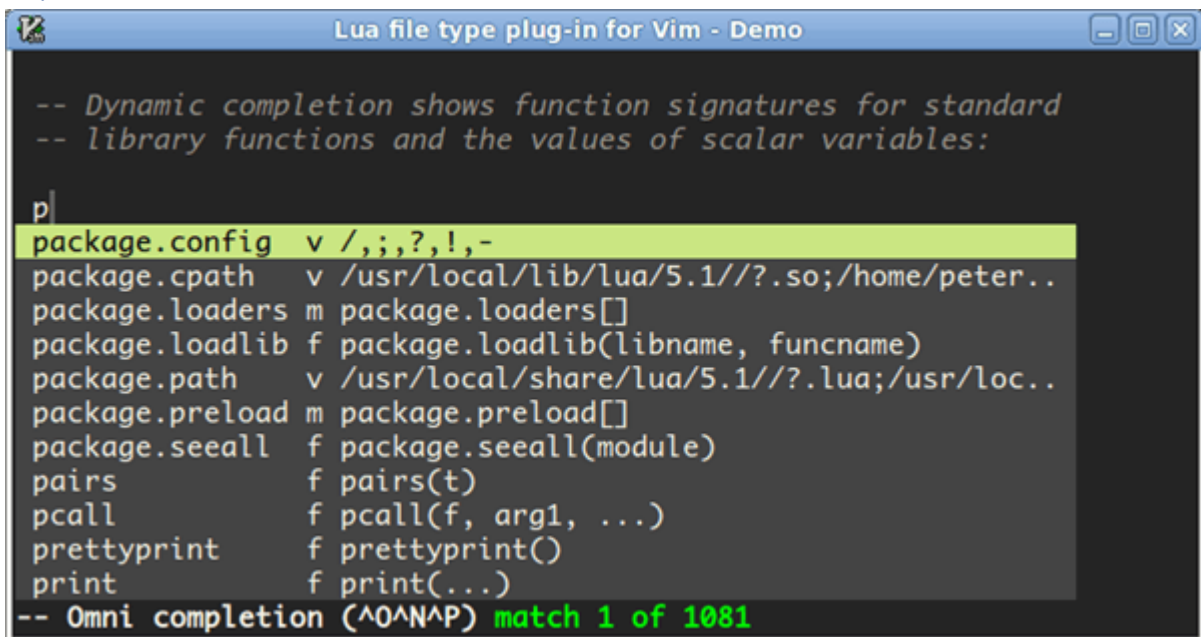
Cons:

- Sometimes the system runs slow on Linux.

Tool Cost /Plan Details: Free

Findings: Visual Studio Code is a lightweight application and is mainly used for ASP.Net and C#.

#6) VIM



```
-- Dynamic completion shows function signatures for standard
-- library functions and the values of scalar variables:

p|
package.config v /,;,?!,,-
package.cpath v /usr/local/lib/lua/5.1/??.so;/home/peter..
package.loaders m package.loaders[]
package.loadlib f package.loadlib(libname, funcname)
package.path v /usr/local/share/lua/5.1/??.lua;/usr/loc..
package.preload m package.preload[]
package.seecall f package.seecall(module)
pairs f pairs(t)
pcall f pcall(f, arg1, ...)
prettyprint f prettyprint()
print f print(...)
-- Omni completion (^O^N^P) match 1 of 1081
```

Vim text editor provides support for hundreds of programming languages. In UNIX and Mac, it is known as vi. Its latest available version is 8.1.

Features:

- Syntax highlighting.
- It supports editing of compressed files.
- It provides support for mouse interaction.
- Spelling check.

Pros:

- Recording macros.
- It supports many programming languages.
- Availability of search and replace functionality.

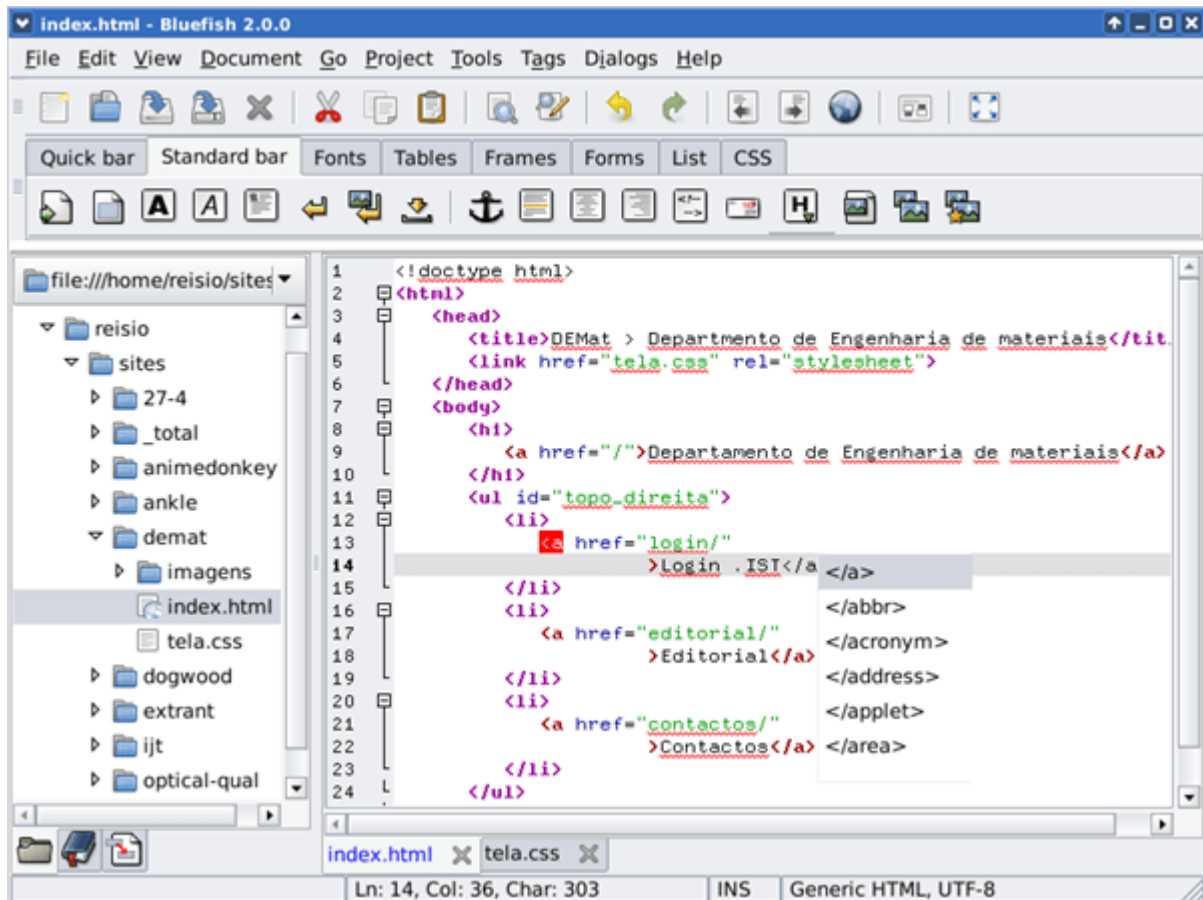
Cons:

- It is difficult to learn.
- It provides limited IDE features.

Tool Cost /Plan Details: Free

Findings: Vim is a good text editor, however it has a steep learning curve.

#7) BLUEFISH



Bluefish is a free text editor. It can be used on many operating systems like Windows, Linux, Mac OS, and Solaris. This easy to use system can be used for programming and websites developments.

Bluefish can be used for:

HTML	JavaScript	Java	ColdFusion	JSP
XHTML	C++	Google Go	Perl	Python
CSS	C	Vala	SQL	Ruby

XML

PHP

Ada

D

Shell

Features:

- Syntax highlighting.
- Auto-completion & Code Folding.
- Code Navigation.
- Bookmarks.
- Bluefish is an extensible system.

Pros:

- It supports multiple encoding.
- It has Unicode character browser.

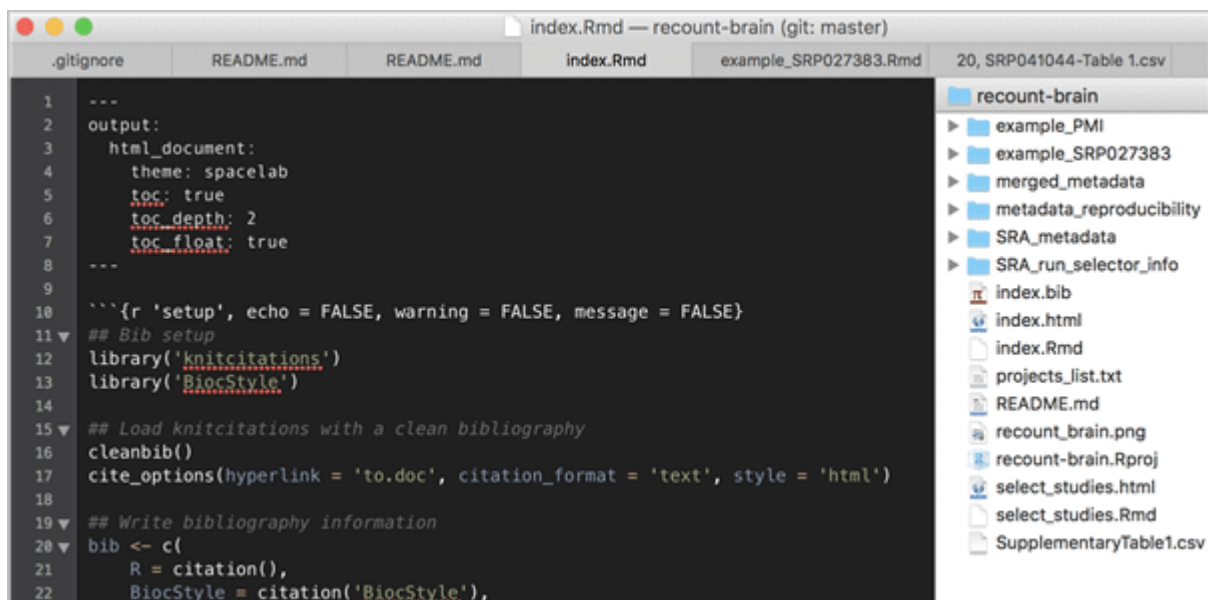
Cons:

- Sometimes the system gets slow.

Tool Cost/Plan Details: Free

Findings: Bluefish supports many markup languages and it is well known for its high speed.

#8) TEXTMATE



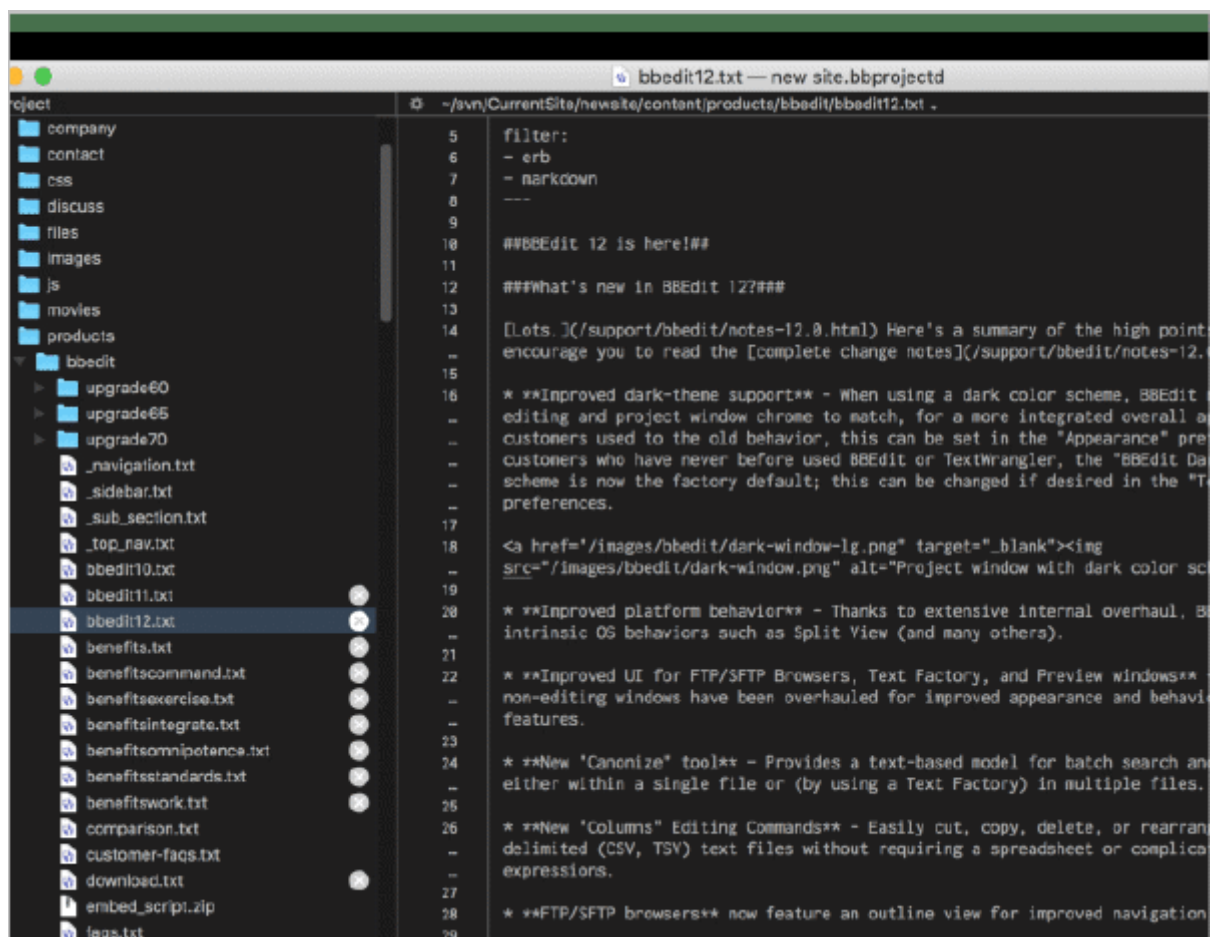
TextMate is a Mac text editor. You can use TextMate for more than 50 languages.

Features:

- Search and replace functionality within a project.
- Auto-parsing for brackets.
- You can record Macros without programming.
- It provides some project management features.

- You can select a theme for syntax highlighting.
- Pros:**
- You can search and replace regular expressions.
 - It supports switching between the files within the project in just a few keystrokes.
- Cons:**
- It doesn't provide guided code completion facility.
 - It doesn't have a built-in HTML validator.
- Tool Cost/Plan Details:** Free
- Findings:** TextMate is one of the best free text editors for Mac. The option of Smart switching between the files helps a lot.

#9) TEXTWRANGLER



TextWrangler is a text and code editor for Mac OS. It is now known as BBEdit. It has integrated support from Mac OS X's Spelling service.

For syntax coloring and function navigation, it supports the following languages:

ANSI C	C++	Fortran	Java	Markdown
--------	-----	---------	------	----------

Objective C	Perl	Tcl	Tex	Object Pascal
Python	PHP	Rez	Ruby	Unix Shell Scripts

Features:

- It supports the comparison of text files.
- It allows for multiple Undo.
- It supports auto indentations.
- You can split the editing windows.
- Extensive FTP & SFTP support.

Pros:

- It supports full-screen mode.
- Powerful search and replace option.
- It has rectangular text selection.

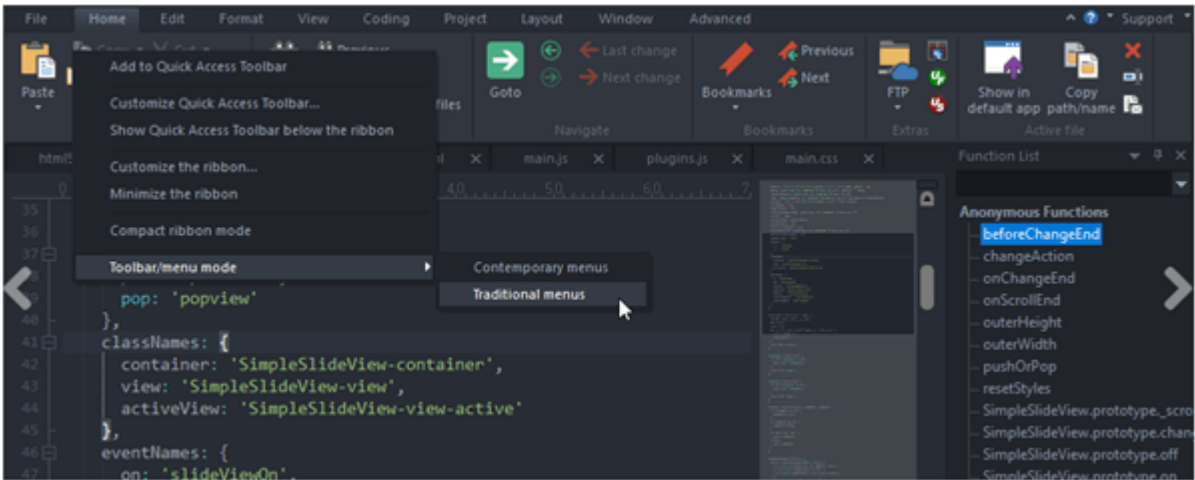
Cons:

- It is available only for Mac OS.

Tool Cost/Plan Details: \$49.99

Findings: TextWrangler is the text editor for Mac. It is not free but offers good features at a small price.

#10) ULTRAEDIT



UltraEdit is a text editor by IDE Computer Solutions. It can be used on Windows, Linux, and Mac OS. It has features for developers and enterprises as well. One UltraEdit license can be used on three machines of any platform.

Features:

- It has integrated SSH, FTP, and Telnet.
- It supports file and data sorting.
- It supports Macros and scripts.
- It supports file encryption and decryption.
- Multi-caret editing.

Pros:

- UltraEdit can handle large files. It can work with the files of size 4GB and more.
- You can select or set the theme as per your choice.
- The smart search option which is provided can search any word or regular expression across files.

Cons:

- It is expensive.

Tool Cost /Plan Details: \$79.95 per year.

Findings: It is a little expensive solution but offers good features and it is worth the price.

ADDITIONAL EDITORS TO CONSIDER

#11) Light Table: It can be used on Windows, Linux, and Mac. It is a lightweight open source application. It provides many features like inline evaluation, watches, malleable, and plugin manager.

#12) Coda: Coda is a text editor for Mac OS. It provides you with a feature of opening and managing local and remote files.

It provides many features like touch bar, fast syntax highlighting, guidance in vertical indentation, plugins, and helps in syncing your sites and passwords. You can buy it for \$99.

#13) jEdit: jEdit can be used on Windows, Mac, UNIX, and VMS. For auto indentation and syntax highlighting it supports more than 200 languages. It is available for free. It has a plugin manager, for managing plugins.

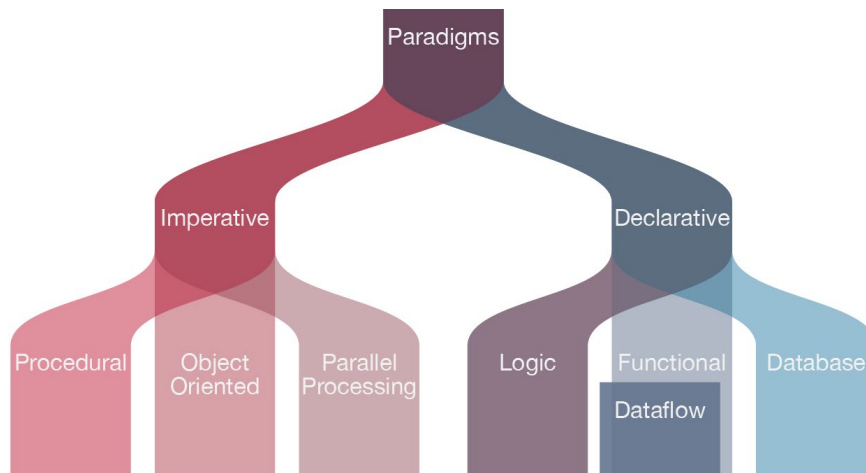
#14) gedit: gedit is an open source text editor. It can be used on Windows and Mac. It provides many features like editing from remote locations, auto indentation, undo, file reverting, and much more.

#15) CoffeeCup: CoffeeCup HTML editor is easy to use. You can start website designing from scratch or you can use it to edit the existing one. It will provide several features in a cost-effective way. It has two versions, one is free and you can buy another one for \$49.

6. PROGRAMMING LANGUAGES

TYPES OF PROGRAMMING LANGUAGES

The different types of programming languages are discussed below.



PROCEDURAL PROGRAMMING LANGUAGE

The procedural programming language is used to execute a sequence of statements which lead to a result. Typically, this type of programming language uses multiple variables, heavy loops and other elements, which separates them from functional programming languages. Functions of procedural language may control variables, other than function's value returns. For example, printing out information.

FUNCTIONAL PROGRAMMING LANGUAGE

Functional programming language typically uses stored data, frequently avoiding loops in favor of recursive functions. The functional programming's primary focus is on the return values of functions, and side effects and different suggests that storing state are powerfully discouraged. For example, in an exceedingly pure useful language, if a function is termed, it's expected that the function not modify or perform any o/p. It may, however, build algorithmic calls and alter the parameters of these calls. Functional languages are usually easier and build it easier to figure on abstract issues, however, they'll even be "further from the machine" therein their programming model makes it difficult to know precisely, but the code is decoded into machine language (which is often problematic for system programming).

OBJECT-ORIENTED PROGRAMMING LANGUAGE

This programming language views the world as a group of objects that have internal data and external accessing parts of that data. The aim this programming language is to think about the fault by separating it into a collection of objects that offer services which can be used to solve a specific problem. One of the main principles of object-oriented programming language is encapsulation that everything an object will need must be inside of the object. This language also emphasizes reusability through inheritance and the capacity to spread current implementations without having to change a great deal of code by using polymorphism.

SCRIPTING PROGRAMMING LANGUAGE

These programming languages are often procedural and may comprise object-oriented language elements, but they fall into their own category as they are normally not full-fledged programming languages with support for development of large systems. For example, they may not have compile-time type checking. Usually, these languages require tiny syntax to get started.

LOGIC PROGRAMMING LANGUAGE

These type of language lets programmers make declarative statements and then allows the machine to reason about the consequences of those statements. In a sense, this language doesn't tell the computer how to do something, but employing restrictions on what it must consider doing.

To call these group "type of languages" is really a bit confusing. It's easy to program in an object-oriented style in C language. In truth, most of the languages include ideas and features from various domains, which only helps to increase the usefulness of these type of languages. Nevertheless, most of the programming languages do not best in all styles of programming.

THE DIFFERENCE BETWEEN DIFFERENT PROGRAMMING LANGUAGES

C++ LANGUAGE

The C++ language has an object oriented structure which is used in large projects. Programmers can collaborate to code one application into different parts or even one individual work on each part of the program. The structure of object oriented also permit code to be reused many times. This language is an efficient language. But, many programmers will disagree

C LANGUAGE

The C language is a basic programming language and it is a very popular language, particularly used in game programming, Because C language includes the additional packing of the C++, Every programmer uses this language because it makes programs faster. However the value of this language gives the reusability of C++ to get the slight increase in performance with C language.

PASCAL LANGUAGE

Pascal language is mostly a teaching language and few industries use this language to write the programs. This language tends to use keywords instead of symbols and braces in C language. So this language is very easy for beginners to understand than a programming language like C, C++. Borland is a compiler software company, which is using Delphi

programming language for industrial strength. Delphi is an object oriented language of Pascal, and presently Borland compilers only use it.

FORTRAN LANGUAGE

Fortran language is a number crunching language and still it is used by scientists. This language allows different sizes of variables up to the memory limit in the machine. This language is suitable for engineers, who have to calculate values with high precision. Program in Fortran is inflexible and sometimes it makes difficult to read.

JAVA LANGUAGE

The Java language is a multi-platform language that's particularly helpful in networking. Of course, mostly this language is used on the web with Java applets. However, this language is used to design cross platform programs, Since it similar to C++ in structure and syntax. For C++ programmers, Java language is very easy to learn and it offers some advantages provided by object-oriented programming. Like reusability and it can be difficult to write efficient code in Java. But, nowadays the speed of the Java language has increased and 1.5 version offers some good features for easy program making.

PERL LANGUAGE

Perl language is a file management language for UNIX. But it is more popular for its common gateway interface programming (CGI). It is a term for programs that web servers can perform to allow additional capabilities of web pages. Pearl language is a method for searching text and it is used for useful server functions and other databases, and it is very easy to pick up the fundamentals if you have any experience in any language. As a CGi language, web hosting services select Perl language over C++ language, because the web hosts can review Perl script files.

PHP LANGUAGE

The PHP language is used to design web pages and sometimes it is also used as scripting language. This language is designed to develop a rapid website, and as a result comprises features which make it easy generate HTTP headers and link to databases. As a scripting language, it includes a set of components permit the programmer to easily get up to speed. However, it has more sophisticated object-oriented features.

LISP LANGUAGE

LISP language is mostly used in computer science research and it stores all data in lists such as arrays. The syntax of list is very simple and easy to make for developers to implement structures.

SCHEME LANGUAGE

The scheme language is an alternate of LISP language, and it has a simpler syntax and features. Any project under the scheme language will result in the re implementation of most of the LISP language. But this is very popular introductory language in the computer science department of MIT's. This language easily solves the problems instead of worrying about syntaxes of programming language.

7. COMPETITIVE PROGRAMMING FOR BEGINNERS: TOPCODER CHALLENGES

WHAT IS COMPETITIVE PROGRAMMING?

Competitive programming is solving coding problems using algorithm and data structure. The participants need to write code under various restrictions like memory limits, execution time, limit of coding, space, etc.

The winner is declared base on some problems solved and time spent on writing successful programming solutions. However, it also includes other factors like quality of output produced, execution time, program size, etc.

BENEFITS OF COMPETITIVE PROGRAMMING

Here, are benefits of taking parts in competition:

- Competitive programming helps you to improve your logical and analytical skills
- It improves your algorithmic knowledge.
- It is an excellent addition to your CV.
- Improve your network of friends who are also passionate about programming
- It is supported by world famous organizations like Google, Amazon, Facebook, IBM, and others.

HISTORY OF COMPETITIVE PROGRAMMING

In the competitive programming world, the oldest contest is ACM-ICPC (International Collegiate Programming Contest). It was originated in the 1970s and has grown to span across 88 countries.

Interest in programming challenges has grown extensively after 2000. It was possible due to the growth of the Internet. It allows coding contest sites to hold international contests online.

SKILLS REQUIRED IN THE COMPETITIVE PROGRAMMING EXAMS

Skill	Details
Important skills	<ul style="list-style-type: none"> • Algorithm Skills • Basic Math Skills • Programming Skills • Typing Speed Skills • Debugging Skills
Complete search	<ul style="list-style-type: none"> • Brute force searching using the entire search space • Beating the time limit • Optimizing the search
Divide and Conquer	<ul style="list-style-type: none"> • Dividing the coding problem into smaller parts • Binary search
Dynamic programming	<ul style="list-style-type: none"> • Determine problem state • Faster and more elaborate recursive backtracking

HOW TO GET STARTED?

Here, are some useful tips that you should know before getting started with Competitive programming:

- It's vital that you select a programming language which you are most comfortable with
- Sign up on any online coding challenge website and start solving their practice problems
- You need to learn about Data Structures which is vital to solving complex coding issues.
- You need to understand the concept of complexity in programs.
- Learn important programming patterns like recursion, Dynamic Programming, etc.

TOP COMPETITIVE PROGRAMMING SITES

The most important best coding challenge sites are:

- TopCoder
- Coderbyte

- Codewars
- CodeChef
- Codeforce
- Hackerearth

TOPCODER:



Topcoder is a widely popular platform for competitive programming. It offers a lot of algorithmic challenges that you can complete on your own using their code editor tool. The website is better suited for an advanced audience.

Visit Here: <https://www.topcoder.com/>

CODERBYTE



Coderbyte offers more than 200 various types of coding challenges which you can solve in an online editor using 10 different programming languages. It provides official solutions and more than 800,000 user created solutions.

Visit Here: <https://coderbyte.com/>

CODEWARS



Codewars offer an extensive collection of coding challenges. You can submit your challenges which will be edited by the community. It allows you to solve these challenges online using their editor of 20+ programming languages.

Visit Here: <https://www.codewars.com/>

CODECHEF



CodeChef offers hundreds of competitive programming challenges. It allows you to write code in their online editor and view a collection of challenges. The coding problems are separated into different categories based on your skill level.

Visit Here: <https://www.codechef.com/>

CODEFORCES




Codeforces is a Russian base competitive programming website. It regularly hosts coding contest where competitive programmers from all over

the world.

Visit Here: <https://codeforces.com/>

HACKEREARTH:

 HackerEarth offers enterprise software solutions. It helps companies or individual with their innovation management and technical recruitment needs. HackerEarth has conducted 10,000+ programming challenges to date.

Visit Here: <https://www.hackerearth.com/>

Myths and reality of competitive programming

Myths	Reality
It's too late to start competitive programming	There is no fixed age for this best to start earlier in your programming career.
It is an excellent way to get a software programming job	No, it is not true as it is a sport which may benefits. However, it doesn't offer a job guarantee.
You need to solve lots of computing programs before starting competitive programming.	You can learn theory, but you solving computing challenges will not help as every competition is unique with its unique challenges.
You need to an expert in algorithm	You need to be able to solve the problems
Competitive programmers are all experienced programmers.	No, it is for everyone even beginner code can participate
It is just a hobby or a game	Competitive programming, in contrast, covers some of the same skills taught in the computer science curriculum, but at a much deeper level. So, you can't call it a game.

BEST PRACTICES FOR COMPETITIVE CODING EXAMS

Although there is no fixed rule, still some best practices that you should follow are:

- Try to learn new and efficient algorithms and implement them when needed.
- Develop a good knowledge of data structures like lists, trees, and graphs
- Read the privacy information from the competitive programming website and contact them before signup.
- You should select a problem for which you can find a solution and possibly an explanation of those solutions.
- Write test cases for the problem and test your logic using those test cases before submitting.
- You need to maintain pre-written templates for Standard algorithms, necessary headers, functions, etc.
- Use functions/methods and minimize repetition of code.
- Test your code for worst case and try to figure out the best solution for this situation

Drawbacks of competitive programming

- Runtime Errors on problems with bad inputs are difficult to debug.
- You are so used to small rewards and short-term goals. In real life when you work on any project, it mostly takes two-three months before you get the reward for the project you're working on
- You will likely become master of coding, but you won't know any frameworks (important for real projects)
- It is time-consuming and more tedious to participate, and you need to do it repeatedly.

Summary

- Competitive programming is solving coding problems using algorithm and data structure.
- Coding contests help you to improve your logical and analytical skills.
- ICPC was originated in the 1970s and has grown to span across 88 countries.
- Algorithm Skills, Basic Math Skills, Programming Skills, Typing Speed Skills are required in the competitive Programming Exams.
- It's crucial that you select a programming language which you are most comfortable with
- The most important best coding challenge sites are: 1)TopCoder 2) Coderbyte 3) Codewars 4) CodeChef 5) Codeforce 6) Hackerearth.
- It is a common myth that it's too late to start competitive programming. However, the reality is that there is no fixed age for it.
- Develop a good knowledge of data structures like lists, trees, and graphs is a good best practice for competitive coding exams

A HOW TO: CODE CHALLENGES AT TOPCODER

In a Code challenge, members are required to create a fully functional application from scratch or improve an existing application by adding new features and fixing bugs. There are many challenges for many of the most popular languages / technologies (like C++, Java, Python, JavaScript and more) being launched for you to register and start competing.

The best way to get started with Code challenges is:

1. Choose the one language you feel the most comfortable with.
2. Focus on learning the latest best practices for that language.
3. Register on challenges that require that specific language and see how well you can perform.
4. Read the review responses carefully and try to understand why you lost some points and how you can prevent this in future challenges.
5. If you get a “passing score” you can download the winning submission(s). This will help you understand how top-competitors work and how they write their code.
6. Last but not least, never give up! Is absolutely normal for a beginner to lose his/her first challenge. If you lose a challenge, make sure to learn from your mistakes and improve your skills from the reviewer’s feedback because that is what will make you a better programmer.

Keys to success:

- Read every challenge carefully and understand submission requirements.
- Ask as soon as possible questions in the challenge forum.
- Understand how winners are chosen / how submissions are scored.
- Register and submit within the deadlines.

EXERCISE: TOPCODER LEARNING SERIES - FLUTTER - CHALLENGE 1 - WRITE YOUR FIRST APP

CHALLENGE OVERVIEW

Flutter is Google's UI toolkit for building beautiful, natively compiled applications for mobile, web, and desktop from a single codebase. Flutter works with existing code, is used by developers and organizations around the world, and is free and open source.

THE MAJOR COMPONENTS OF FLUTTER INCLUDE:

- Dart platform: Flutter apps are written in the Dart language and make use of many of the language's more advanced features.
- Flutter engine: Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform-specific SDKs such as those provided by Android and iOS.
- Foundation library: The Foundation library, written in Dart, provides basic classes and functions which are used to construct applications using Flutter, such as APIs to communicate with the engine.

- Design-specific widgets: The Flutter framework contains two sets of widgets which conform to specific design languages. Material Design widgets implement Google's design language of the same name, and Cupertino widgets implement Apple's Human Interface Guidelines iOS design

Compared to other frameworks, Flutter has the following advantages

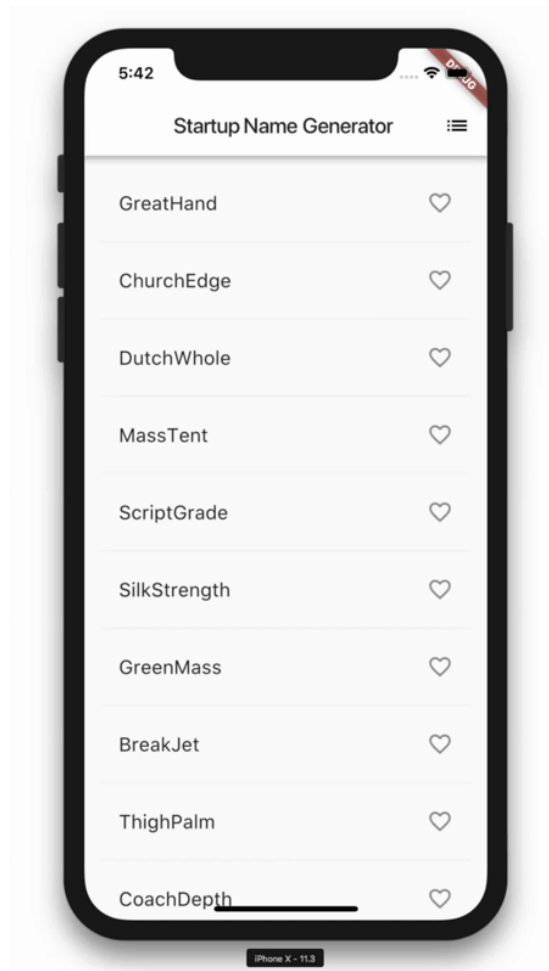
- High productivity. Since Flutter is cross-platform, you can use the same code base for your iOS and Android app. This can definitely save you both time and resources.
- Great performance. Dart compiles into native code and there is no need to access OEM widgets as Flutter has its own. This means less mediated communication between the app and the platform. Flutter is the only mobile SDK that provides reactive views without requiring a JavaScript bridge. Other SDK, like React Native requires a Javascript bridge to translate JSX to the commands that drives the native elements.
- Fast and simple development. One of the most lauded features of Flutter is hot reload which allows you to instantly view the changes made in the code on emulators, simulators and hardware. In less than a second, the changed code is reloaded while the app is running with no need for a restart.
- Compatibility. Since widgets are part of the app and not the platform, you'll likely experience less or no compatibility issues on different OS versions. This in turn means less time spent on testing.
- Open-source. Both Flutter and Dart are open-source and free to use, and provide extensive documentation and community support to help out with any issues you may encounter.

Flutter is the next generation cross-platform development framework. The stable version was released in 2018, and it is more and more popular during the recent years. Flutter is a young but useful technology, you will never regret spending time learning it.

Learning Series - Flutter series challenges are exactly for guiding you to enter the Flutter world! The series will consist of 3 Challenges, Blogs, Tutorials and Forum Discussions to help you Learn Flutter and Make Quick Apps.

REQUIREMENTS FOR THIS CHALLENGE

In this challenge, you will follow the guide in Google Codelabs to write your first Flutter app. Finally, you will build an app like below:



You will need to complete both Part 1 and Part 2 of the tutorial.

WHAT WILL YOU LEARN?

From Part 1, you will learn

- How to write a Flutter app that looks natural on iOS, Android, and the web
- Basic structure of a Flutter app
- Finding and using packages to extend functionality
- Using hot reload for a quicker development cycle
- How to implement a stateful widget
- How to create an infinite, lazily loaded list

From Part 2, you will learn

- How to add interactivity to a stateful widget.
- How to create and navigate to a second screen.
- How to change the look of an app using themes.

You will also learn some basics of the Dart language.

SCORING GUIDELINE

You need to follow the guideline step by step in Google Codelabs to write your first Flutter app, then you need to submit your code and a video to showcase your demo app (in either Android or iOS emulator/real devices).

Enjoy Learning!

FINAL SUBMISSION GUIDELINES

SUBMIT A ZIP ARCHIVE, INCLUDING

- The full source code of your Flutter project
- A video to showcase your demo app (in either Android or iOS emulator/real devices). Please upload your video elsewhere (like in Youtube, Google Drive, Dropbox etc) and share the link in your submission.
- A simple README about how to deploy and run your code

8. WRITING THE CODE – STEPS

STEP 1: CHOOSE THE RIGHT LANGUAGES

Different types of programming require different skills. For example, if you want to learn how to code a website, you should learn languages like HTML, CSS, Java, and JavaScript. Likewise, mobile app development requires Java (for Android) or Swift (for iOS). Choosing the right language is extremely important, which is why we have outlined some of the most popular among beginners below:

JAVA

Java is arguably the king of programming languages in the modern world. It is prevalent, versatile, and it is used for things like Android app creation and back-end web development. Java is quite a functional language for learning how to code for beginners. It is relatively simple to learn, it reads like English, and there is a lot of work out there for entry-level Java programmers.

PYTHON

If Java is the king of programming languages, then Python is the queen. It is also super versatile, and it has a lot of different uses. A lot of academics and researchers use Python to create data analysis programs to help their studies. It is also behind the rise of new technologies like artificial intelligence and machine learning. It is easy to learn, and it is the most popular language taught in schools. That's why it's perfect for those who are looking for how to code for beginners.

RUBY

Ruby is a language that is becoming increasingly popular for full-stack web development. It allows for the smooth integration of both front and back-end components of a new website through the Ruby on Rails framework.

SOLIDITY

If you are interested in cryptocurrencies and blockchain development, then Solidity is one of the best programming languages to learn. Although not well known or widespread by any stretch of the imagination, Solidity is the primary language for smart contract development on the Ethereum blockchain.

HTML

If you want to get into front-end web development, then you have to learn HTML. Used for marking up text so that computers can understand it, HTML (or Hypertext Markup Language) is one of the simplest languages to learn and is where a lot of people start when learning how to code for beginners.

HTML is very simple, and the best way to learn it is arguably to first familiarize yourself with the main components of the language and then to get out there and practice how to write a code for beginners.

CSS

CSS is used almost exclusively alongside HTML to style and format content on webpages. It allows you to do things like change the color and size of elements, change the position of elements on a webpage, and change the entire layout of a webpage.

Like HTML, CSS is essential if you plan on becoming a front-end web developer. In my opinion, it's a good idea to learn HTML and CSS alongside each other

JAVASCRIPT

JavaScript is another language that is quite versatile and extremely widespread – much like Java and Python. Although it has historically been a front-end web development language that allowed the creation of interactive elements, JavaScript is becoming increasingly used for other things – including back-end web development.

If you want to become a decent front-end or back-end web developer, we recommend learning JavaScript.

STEP 2: CHOOSE THE RIGHT RESOURCES TO HELP YOU LEARN

Now that we've looked at some of the most popular languages to help you discover how to code for beginners, it's time to look at the best ways to learn them. Some of these include:

Online coding courses: Online coding courses, such as those recommended above, are one of the best ways to get a feel for a new programming language. When done right, they will teach you some of the essential syntaxes of a language and how it is used. They are usually taught in a fun, engaging manner, and many of them allow you to write code directly in your browser as you learn.

Textbooks: Code reference books are a great resource that every single beginner should make use of. They contain all of the syntax and conventions of a language, allowing you to look things up when you're confused or unsure about how to proceed with your code.

YouTube videos: Online videos and video tutorials are great resources that can help you get your head around complex concepts. You should use them alongside an online course or another method of learning how to code for beginners.

STEP 3: DOWNLOAD A CODE EDITOR

Although a lot of online courses let you write code in a code editor directly in your browser, you need to realize that this isn't how it works in the real world. Once you graduate from your online course, you will need to start writing your code directly into a code editor, so it's a good idea to start practicing doing this now.

Most programming languages have their code editors that you can download. They usually come complete with interactive development environments (IDEs) that will help you build clear, error-free code. Simple languages like HTML and CSS can be written straight into a simple text document if you find that easier.

STEP 4: PRACTICE WRITING YOUR PROGRAMS

Once you have a basic working knowledge of your chosen language, you will need to start creating your programs and projects. Although you might feel that this is a long way away right now, you will be surprised how fast you progress once you start learning.

The type of program or project you start building will depend on the language you learn. For example, if you are learning how to code a website using HTML/CSS, you might choose to start building your website. You can even begin with drag & drop website builders to make the learning process easier. However, a knowledge of Java will allow you to begin programming your Android app or game.

Remember, practice makes perfect, and practice is the key to becoming a successful programmer. Practice, practice, and more practice!

STEP 5: JOIN AN ONLINE COMMUNITY

Navigating the intricacies of learning how to code for beginners on their own can be extremely difficult. There will be times when you need support. For example, let's say that you're learning a language, and you've come across a piece of syntax that you can't get your head around. Joining an online chat board or coding forum will give you access to people who will be able to answer your questions and help you get past any problems you're having.

STEP 6: HACK SOMEONE ELSE'S CODE

Once you have a pretty good idea of how to code in a particular language, it's time to delve deeper. One of the best ways to keep learning and to reinforce your knowledge of a programming language is to unpack someone else's code.

Find a program or code file somewhere – GitHub is a great place to start – and open it in your code editor. Work through the code and make sure that you understand every line and what its function is. Add comments if you need to. If you see something which you think could be done in a more efficient way or which you think you can improve, change it.

Once you have finished, save your edited code and re-upload it so that other people can critique your effort.

STEP 7: KEEP LEARNING

In a way, programming languages are like any other language. They require regular practice if you want to become fluent in them, and you will forget things if you don't practice using them. It is, therefore essential to make sure that you keep learning new things every day.

Don't fall into the habit of using the same coding patterns and syntax for everything. Instead, keep striving to learn new ways to do things and to improve your method. I try and make a habit of learning one thing every single day. This could be a new piece of syntax, a new way to write a line of code, or simply a better way of debugging a program.

9. TESTING AND DEPLOYMENT

WHAT IS DEPLOYMENT IN SDLC?

Finally, a time has come where you will deliver the product to the client. This could be considered the last phase of the SDLC (Software Development Life Cycle) in many of the cases, off course if you are not responsible for the maintenance. If you are a part of the **maintenance phase** then there is one more phase for you call maintenance, which we do not discuss here.

The objective of the deployment phase is to make the developed software operational in a live environment. A deployment in the operational environment comes only after the product is fully tested and accepted by the business in the acceptance stage of the testing

phase. Here we may need to provide training to the real time users and post-deployment review is done in this phase.

During the training of the real time users, the following points should be covered:

- Deployment schedule and phases in which deployment is planned.
- Discuss all the benefits user will get by using the new system.
- If new software is the remake of old software then the difference should be conveyed to the existing users.
- If we are responsible for the support after sales then we should communicate the proper communication channel for support.

Usually, a project manager should be responsible for a successful execution of the software deployment. Reviewing the deliverable, the status of deployment and the reporting to management should be done by a manager. The managers are accountable for filling in the deployment checklist and inform the change management that deployment is successfully done.

THE THREE PHASES OF THE DEPLOYMENT TESTING CYCLE

Testing a software project before and after deploying it on production is not that difficult. But too often, major bugs appear on production server after the deployment process. To avoid situations in which your production environment is threatened by these bugs, you should use a streamlined deployment and testing flow.

Generally speaking, this flow consists of three phases: *pre-deploy*, *deploy* and *post-deploy*.

During the **Pre-Deploy** testing phase, both the development team and the QA engineer should be tasked with the following items:

1. Ask developers to make Production and Stage environment backups.
2. Ask developers to copy database from Production to Stage.
3. Announce a Code Freeze for the entire development team.
4. Retest new features and bug fixes.
5. Perform general smoke testing using checklists.
6. Report all bugs, categorizing them with a consistent metric for urgency.
7. Test urgent bug fixes and engage in regression testing of this functionality.
8. When Stage is tested and has no major and/or minor bugs (depending on the project),
— announce deployment.

When the QA engineer announces the Code Freeze, developers cannot push, commit, or deploy anything. But they can still continue working on their local environment, so no time is wasted. When the Code Freeze is cancelled, developers can push, commit, and deploy their work to Stage.

After finishing the pre-deploy testing cycle, the manager has to approve **the deploy process** before the developer who is responsible for this process puts it into motion. Having one developer be responsible for deploys is always beneficial, as it avoids misunderstandings like incorrectly merged branches.

After the deployment is done, QA begins the **Post-Deploy phase**, which consists of the following tasks:

1. Retesting new features and bug fixes.
2. Perform general smoke testing using checklists.
3. Report all bugs, categorizing them with a consistent metric for urgency.
4. Urgent bug fixes should first be deployed on the Stage environment. Here, QA can test them, and engage in regression testing before deploying them on production. Then, the testing cycle — including bug fix testing and regression testing — should be repeated.
5. When production is tested and has no issues, — deployment is complete.
6. Cancel Code Freeze, or wait for client feedback as needed before cancelling Code Freeze.

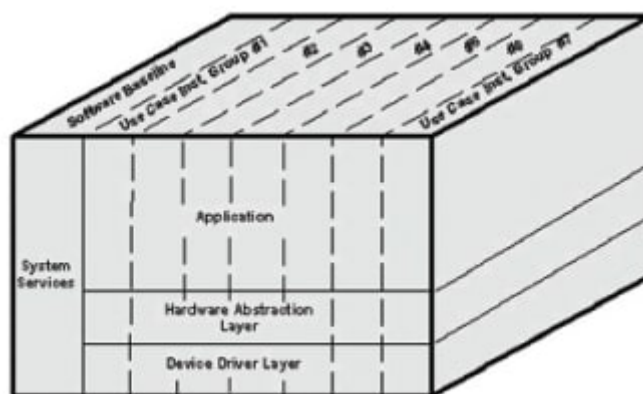
If for any reason issues appear during the deployment process, you should quickly troubleshoot them. Here are some examples of possible problems and their solutions:

1. If production is severely broken after deploy, — use a backup to quickly restore it.
2. If developers violated Code Freeze, — they need to roll back all the changes.
3. To quickly roll back any new features, developers should use separate branches for different features.
4. To avoid timing troubles, — never make deploys late in on Friday evenings.

Testing your software is a crucial part of the deployment cycle. By running through tests before, during, and after deployment, you can improve the quality of your software and ensure a better product.

10. DEVELOPMENT OF SMALL SYSTEMS

Developing software for small-scale embedded applications is different from developing large-scale software applications. Large-scale applications use commercially available ‘one fits all’ software development solutions that are difficult to scale downward and usually miss



the desired process goals. In many cases, developing a small-scale software application development process within an existing corporate environment is quicker, less expensive, and results in superior developer productivity and product quality.

Figure 1: A selected group of use-case instances describe an iteration cycle in the implementation and test

and verification phases of the development process.

Software pioneer Grady Booch famously commented that, “Building quality software in a repeatable and predictive fashion is hard.” This statement not only describes the difficulty of the software development process, but also describes the primary goal of any software

development process — software products should be defect-free, maintainable, and have veracity requirements to guarantee successful operation.

Software development processes can be fully described by four orthogonal views: methodology, process artifacts, process procedures, and quality assurance. This article will focus on the methodology view.

ANALYSIS PHASE

The small-scale application development methodology is best described as a use-case-driven “hybrid spiral” (part waterfall, part iterative). The analysis phase is the waterfall portion of the hybrid. The ability to perform this upfront analysis in detail is a unique advantage of small-scale development.

The cornerstone artifact of the analysis phase is a software requirements document (SRD) that details the requirements in textual use-case sequences, top-level state description, and a hardware interface list. This is an artifact that is maintained throughout the product lifecycle. The product SRD largely consists of textual use-case sequences, which drive the rest of the development (i.e. object identification via sequence diagrams, test cases, and schedule).

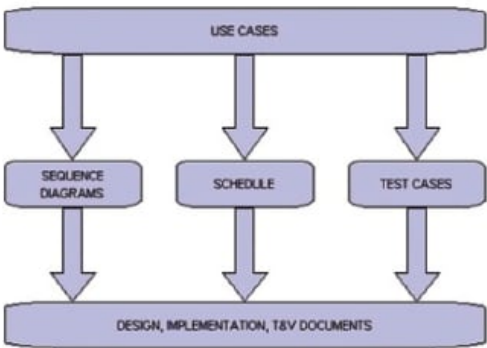


Figure 2: Iterative development is use-case-driven; they are the information baseline source for all other documents.

Avoid any process of requirements discovery that includes partial design and implementation of the software product; this is very inefficient. Changes in the requirements will occur, prompting updates to the SRD, but an efficient development process means that the designers should make every attempt to unambiguously define the

requirements of the customer up front. The process should also accommodate existing processes within the design company, including manufacturing, quality assurance, marketing, and engineering requirements.

Earlier, small-scale application development was defined as following a waterfall and iterative process. An iteration is the software development of one or more use-case scenarios. A schedule, developed as part of the analysis, defines the order of iterations for the implementation and test phases of the process. This order is decided based on customer needs, risk identification, and the importance of each use to the end product. Designers should schedule uses that contain risk items first, using functionality as the variable, and commit to delivering product on time.

For schedule documentation, spreadsheets work fine: one worksheet describes use case delivery, while a second details design, implementation, and verification efforts for each object as they are identified. Once the initial schedule is developed, modifications take less than five minutes per week.

DESIGN PHASE

The design is documented in a System Architecture Specification (SAS). In addition to the architecture design, this document details public-member definition of all objects; the development environment, including tool list and installation instructions; design notes; diagnostics for quality assurance procedures; and performance monitoring. The SAS and SRD documents are the only two documents that need to be maintained throughout the product lifecycle. It is the go-to document for a maintenance programmer.

Design documents are notorious for being out of date with the actual implementation. This is avoided by using the SAS for a high-level view of the design, which includes the architecture and a description of each object. Detailed designs of complex individual objects and sequence diagrams for object identification should be documented separately. Designers then can discard these documents after peer reviews and implementation are complete.

The architecture also should include a state diagram that details a brief description of operations visible to the user, including each associated object. When multiple programmers are involved with object assignments, create public members for the object interfaces for use by other team members.

The software baseline (or system services) portion of the application architecture and an object diagram should be the focus of the first SAS iteration. This design provides the software skeleton from which all object modules will derive services and execution. As changes to the SAS occur, each use-case-driven iteration starts with additions and modifications to the SAS document.

IMPLEMENTATION PHASE

Implementation encompasses coding, unit testing, and test-case definition activities. As each iteration passes through the implementation phase, only code each object to support the current iteration. In this way, the designer can develop the use-case iterations and object refactoring as needed and keep complexity to a minimum.

During the test-case phase, developers should map each test case to the SRD use-case scenarios. Because the group may have to develop several test cases per iteration, it is critical that the test-case definitions be generated from analysis of the SRD use-case scenarios and the coding implementation. This has to be done during the implementation of the objects while all facets of the objects function are in the minds of the developers. Additional support or hooks may have to be added to appropriate modules and/or software baseline to support the execution of the use cases. At the end of the implementation phase, the complete test case portfolio forms the basis of the regression test used in test and verification.

TEST AND VERIFICATION (T&V) PHASE

While T&V can include the hardware platform, that activity is out of scope from the software development itself and, therefore, is beyond the scope of this article. T&V includes regression testing, performance testing, and alpha and beta field trials.

A regression test is the execution of the test cases generated during the implementation phase. An iteration is not considered complete until the associated test case has been added to the regression test suite and the regression test suite is executed successfully. A communication port on the target is used to facilitate the regression testing. A scripted, automated test suite implemented on a port controller (i.e. PC utility) is preferable. The regression test is critical to product quality throughout the software products lifecycle.

The performance of the code should be measured prior to each software release to make sure the latest code development has not negatively impacted the application. At a minimum, the performance measurements should be:

- Memory Usage. A minimum of 10 percent of code and data memory for each memory type should be available at all times to enable maintenance programming.
- Stack usage. Designers should measure the size of worst-case stack allotment. A minimum of 20 percent of stack allocation should be available for maintenance programming.
- Execution time for each interrupt service routine.
- Overhead usage. Developers should determine the worst-case total interrupt execution time as a percentage of a real-time clock interval.
- Buffer Usage. Determine the worst case “high water” mark for each buffer used in the system to make sure 20 percent is available for maintenance programming.
- Timer service overruns.

An alpha field trial is an engineering field trial, not a customer or field support field trial, which includes all elements of the system. It usually is performed prior to software completion, but after the high priority and risk use cases are implemented. Its purpose is to test the product to uncover issues relatively early in the development cycle.

In its best form, alpha field trial is implemented by providing company personnel with product and user instructions. A weekly meeting is held to discuss any questions they have. This provides not only defect detection, but also a usability test of the product and its user documentation.

A beta field trial is a customer (or a field support department) trial conducted in a customer domain. This is performed after all software development is completed. The trial is extended with additional releases if failures should occur. A beta field trial is complete when the customer (or field support department) signs for acceptance of the product.

MAINTENANCE PHASE

Maintenance covers the software product support from initial production release to the end of the product lifecycle. The SRD and SAS artifacts from the analysis and design process phases are critical to the cost and quality of maintenance modifications made to the software during this period. They are documents that a maintenance programmer can rely on being accurate and up to date at all times. The same software development phases used in the initial development apply to the maintenance activity.

11. DATABASE DEVELOPMENT LIFE CYCLE

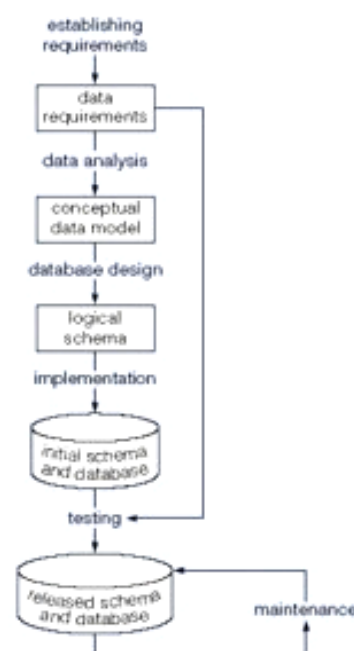
We can use the waterfall cycle as the basis for a model of database development that incorporates three assumptions:

1. We can separate the development of a database – that is, specification and creation of a schema to define data in a database – from the user processes that make use of the database.
2. We can use the three-schema architecture as a basis for distinguishing the activities associated with a schema.
3. We can represent the constraints to enforce the semantics of the data once within a database, rather than within every user process that uses the data.

Figure on the left: A waterfall model of the activities and their outputs for database development.

Using these assumptions and Figure 13.2, we can see that this diagram represents a model of the activities and their outputs for database development. It is applicable to any class of DBMS, not just a relational approach.

Database application development is the process of obtaining real-world requirements, analyzing requirements, designing the data and functions of the system, and then implementing the operations in the system.



REQUIREMENTS GATHERING

The first step is *requirements gathering*. During this step, the database designers have to interview the customers (database users) to understand the proposed system and obtain and document the data and functional requirements. The result of this step is a document that includes the detailed requirements provided by the users.

Establishing requirements involves consultation with, and agreement among, all the users as to what persistent data they want to store along with an agreement as to the meaning and interpretation of the data elements. The data administrator plays a key role in this process as they overview the business, legal and ethical issues within the organization that impact on the data requirements.

The *data requirements document* is used to confirm the understanding of requirements with users. To make sure that it is easily understood, it should not be overly formal or highly encoded. The document should give a concise summary of all users' requirements – not just a collection of individuals' requirements – as the intention is to develop a single shared database.

The requirements should not describe how the data is to be processed, but rather what the data items are, what attributes they have, what constraints apply and the relationships that hold between the data items.

ANALYSIS

Data analysis begins with the statement of data requirements and then produces a conceptual data model. The aim of analysis is to obtain a detailed description of the data that will suit user requirements so that both high and low level properties of data and their use are dealt with. These include properties such as the possible range of values that can be permitted for attributes (e.g., in the school database example, the student course code, course title and credit points).

The conceptual data model provides a shared, formal representation of what is being communicated between clients and developers during database development – it is focused on the data in a database, irrespective of the eventual use of that data in user processes or implementation of the data in specific computer environments. Therefore, a conceptual data model is concerned with the meaning and structure of data, but not with the details affecting how they are implemented.

The conceptual data model then is a formal representation of what data a database should contain and the constraints the data must satisfy. This should be expressed in terms that are independent of how the model may be implemented. As a result, analysis focuses on the questions, "What is required?" not "How is it achieved?"

LOGICAL DESIGN

Database design starts with a conceptual data model and produces a specification of a logical schema; this will determine the specific type of database system (network, relational, object-oriented) that is required. The relational representation is still independent of any specific DBMS; it is another conceptual data model.

We can use a relational representation of the conceptual data model as input to the logical design process. The output of this stage is a detailed relational specification, the logical schema, of all the tables and constraints needed to satisfy the description of the data in the conceptual data model. It is during this design activity that choices are made as to which tables are most appropriate for representing the data in a database. These choices must take into account various design criteria including, for example, flexibility for change, control of duplication and how best to represent the constraints. It is the tables defined by the

logical schema that determine what data are stored and how they may be manipulated in the database.

Database designers familiar with relational databases and SQL might be tempted to go directly to implementation after they have produced a conceptual data model. However, such a direct transformation of the relational representation to SQL tables does not necessarily result in a database that has all the desirable properties: completeness, integrity, flexibility, efficiency and usability. A good conceptual data model is an essential first step towards a database with these properties, but that does not mean that the direct transformation to SQL tables automatically produces a good database. This first step will accurately represent the tables and constraints needed to satisfy the conceptual data model description, and so will satisfy the completeness and integrity requirements, but it may be inflexible or offer poor usability. The first design is then flexed to improve the quality of the database design. *Flexing* is a term that is intended to capture the simultaneous ideas of bending something for a different purpose and weakening aspects of it as it is bent.

First, for a given conceptual data model, it is not necessary that all the user requirements it represents be satisfied by a single database. There can be various reasons for the development of more than one database, such as the need for independent operation in different locations or departmental control over “their” data. However, if the collection of databases contains duplicated data and users need to access data in more than one database, then there are possible reasons that one database can satisfy multiple requirements, or issues related to data replication and distribution need to be examined.

Second, one of the assumptions about database development is that we can separate the development of a database from the development of user processes that make use of it. This is based on the expectation that, once a database has been implemented, all data required by currently identified user processes have been defined and can be accessed; but we also require flexibility to allow us to meet future requirements changes. In developing a database for some applications, it may be possible to predict the common requests that will be presented to the database and so we can optimize our design for the most common requests.

Third, at a detailed level, many aspects of database design and implementation depend on the particular DBMS being used. If the choice of DBMS is fixed or made prior to the design task, that choice can be used to determine design criteria rather than waiting until implementation. That is, it is possible to incorporate design decisions for a specific DBMS rather than produce a generic design and then tailor it to the DBMS during implementation.

It is not uncommon to find that a single design cannot simultaneously satisfy all the properties of a good database. So it is important that the designer has prioritized these properties (usually using information from the requirements specification); for example, to decide if integrity is more important than efficiency and whether usability is more important than flexibility in a given development.

At the end of our design stage, the logical schema will be specified by SQL data definition language (DDL) statements, which describe the database that needs to be implemented to meet the user requirements.

IMPLEMENTATION

Implementation involves the construction of a database according to the specification of a logical schema. This will include the specification of an appropriate storage schema, security enforcement, external schema and so on. Implementation is heavily influenced by the choice of available DBMSs, database tools and operating environment. There are additional tasks beyond simply creating a database schema and implementing the constraints – data must be entered into the tables, issues relating to the users and user processes need to be addressed, and the management activities associated with wider aspects of corporate data management need to be supported. In keeping with the DBMS approach, we want as many of these concerns as possible to be addressed within the DBMS. We look at some of these concerns briefly now.

In practice, implementation of the logical schema in a given DBMS requires a very detailed knowledge of the specific features and facilities that the DBMS has to offer. In an ideal world, and in keeping with good software engineering practice, the first stage of implementation would involve matching the design requirements with the best available implementing tools and then using those tools for the implementation. In database terms, this might involve choosing vendor products with DBMS and SQL variants most suited to the database we need to implement. However, we don't live in an ideal world and more often than not, hardware choice and decisions regarding the DBMS will have been made well in advance of consideration of the database design. Consequently, implementation can involve additional flexing of the design to overcome any software or hardware limitations.

REALIZING THE DESIGN

After the logical design has been created, we need our database to be created according to the definitions we have produced. For an implementation with a relational DBMS, this will probably involve the use of SQL to create tables and constraints that satisfy the logical schema description and the choice of appropriate storage schema (if the DBMS permits that level of control).

One way to achieve this is to write the appropriate SQL DDL statements into a file that can be executed by a DBMS so that there is an independent record, a text file, of the SQL statements defining the database. Another method is to work interactively using a database tool like SQL Server Management Studio or Microsoft Access. Whatever mechanism is used to implement the logical schema, the result is that a database, with tables and constraints, is defined but will contain no data for the user processes.

POPULATING THE DATABASE

After a database has been created, there are two ways of populating the tables – either from existing data or through the use of the user applications developed for the database.

For some tables, there may be existing data from another database or data files. For example, in establishing a database for a hospital, you would expect that there are already

some records of all the staff that have to be included in the database. Data might also be brought in from an outside agency (address lists are frequently brought in from external companies) or produced during a large data entry task (converting hard-copy manual records into computer files can be done by a data entry agency). In such situations, the simplest approach to populate the database is to use the import and export facilities found in the DBMS.

Facilities to import and export data in various standard formats are usually available (these functions are also known in some systems as loading and unloading data). Importing enables a file of data to be copied directly into a table. When data are held in a file format that is not appropriate for using the import function, then it is necessary to prepare an application program that reads in the old data, transforms them as necessary and then inserts them into the database using SQL code specifically produced for that purpose. The transfer of large quantities of existing data into a database is referred to as a *bulk load*. Bulk loading of data may involve very large quantities of data being loaded, one table at a time so you may find that there are DBMS facilities to postpone constraint checking until the end of the bulk loading.

12. SYSTEM INTEGRATION

DEFINITION - WHAT DOES *SYSTEM INTEGRATION (SI)* MEAN?

System integration (SI) is an IT or engineering process or phase concerned with joining different subsystems or components as one large system. It ensures that each integrated subsystem functions as required.

SI is also used to add value to a system through new functionalities provided by connecting functions of different systems.

For the last decade, the aggregation of different component systems or subsystems that cooperate to deliver a whole functionality has been the focus of industries that use technology. This is known as the modular approach to systems building, and the SI process has always been at the near-end of the development cycle. Because systems or subsystems to be integrated may span different fields in software and hardware engineering, a SI engineer must have a broad range of skills and breadth of knowledge.

METHODS OF INTEGRATION

- **VERTICAL INTEGRATION** (as opposed to "horizontal integration") is the process of integrating subsystems according to their functionality by creating functional entities also referred to as **silos**.¹ The benefit of this method is that the integration is performed quickly and involves only the necessary vendors, therefore, this method is cheaper in the short term. On the other hand, cost-of-ownership can be substantially higher than seen in other methods, since in case of new or enhanced functionality, the only possible way to implement (scale the system) would be by implementing another silo. Reusing subsystems to create another functionality is not possible.
- **STAR INTEGRATION**, also known as **spaghetti integration**, is a process of systems integration where each system is interconnected to each of the remaining subsystems. When observed from the perspective of the subsystem which is being

integrated, the connections are reminiscent of a star, but when the overall diagram of the system is presented, the connections look like spaghetti, hence the name of this method. The cost varies because of the interfaces that subsystems are exporting. In a case where the subsystems are exporting heterogeneous or proprietary interfaces, the integration cost can substantially rise. Time and costs needed to integrate the systems increase exponentially when adding additional subsystems. From the feature perspective, this method often seems preferable, due to the extreme flexibility of the reuse of functionality.

- **HORIZONTAL INTEGRATION** or **Enterprise Service Bus (ESB)** is an integration method in which a specialized subsystem is dedicated to communication between other subsystems. This allows cutting the number of connections (interfaces) to only one per subsystem which will connect directly to the ESB. The ESB is capable of translating the interface into another interface. This allows cutting the costs of integration and provides extreme flexibility. With systems integrated using this method, it is possible to completely replace one subsystem with another subsystem which provides similar functionality but exports different interfaces, all this completely transparent for the rest of the subsystems. The only action required is to implement the new interface between the ESB and the new subsystem.

The horizontal scheme can be misleading, however, if it is thought that the cost of intermediate data transformation or the cost of shifting responsibility over business logic can be avoided.

- **A COMMON DATA FORMAT** is an integration method to avoid every adapter having to **convert data** to/from every other applications' formats, **Enterprise application integration (EAI)** systems usually stipulate an application-independent (or common) data format. The EAI system usually provides a data transformation service as well to help convert between application-specific and common formats. This is done in two steps: the adapter converts information from the application's format to the bus' common format. Then, semantic transformations are applied on this (converting zip codes to city names, splitting/merging objects from one application into objects in the other applications, and so on).

CHALLENGES OF INTEGRATION

System integration can be challenging for organizations and these challenges can diminish their overall return on investment after implementing new software solutions. Some of these challenges include lack of trust and willing to share data with other companies, unwillingness to outsource various operations to a third party, lack of clear communication and responsibilities, disagreement from partners on where functionality should reside, high cost of integration, difficulty finding good talents, and common **API** standards. These challenges result in creating hurdles that “prevent or slow down business systems integration within and among companies”. Clear communication and simplified information exchange are key elements in building long term system integrations that can support business requirements.

13. ADDITIONAL EXERCISES:

JAVA EXERCISES:

1) Insert the missing part of the code below to output "Hello World".

```
public class MyClass {  
    public static void main(String[] args) {  
        ..("Hello World");  
    }  
}
```

Answer:

```
public class MyClass {  
    public static void main(String[] args) {  
        System.out.println ("Hello World");  
    }  
}
```

2) Comments in Java are written with special characters. Insert the missing parts:

```
 This is a single-line comment  
 This is a multi-line comment 
```

Answer:

```
//This is a single-line comment  
/* This is a multi-line comment*/
```

3) Create a variable named `CARNAME` and assign the value `VOLVO` to it.

```
  = ;
```

Answer:

```
String carName = "Volvo";
```

4) Create a variable named `MAXSPEED` and assign the value `120` to it.

```
  = ;
```

Answer: `int maxSpeed = 120;`

5) Display the sum of 5 + 10, using two variables: x and y.

```
  =  ;
```

```
int y = 10;
```

```
System.out.println(x + y);
```

Answer: `int x = 5;`

6) Create a variable called z, assign x + y to it, and display the result.

```
int x = 5;
```

```
int y = 10;
```

```
  = x + y;
```

```
System.out.println();
```

Answer: `int z = x + y;`

```
System.out.println(z);
```

7) Fill in the missing parts to create three variables of the same type, using a **comma-separated list**:

```
 x = 5  y = 6  z = 50;
```

```
System.out.println(x + y + z);
```

Answer: `int x = 5, y = 6, z = 50;`

```
System.out.println(x + y + z);
```

8) Add the correct data type for the following variables:

```
 myNum = 9;
```

```
 myFloatNum = 8.99f;
```

```
 myLetter = 'A';
```

```
 myBool = false;
```

```
 myText = "Hello World";
```


Answer: int myNum = 9;
 float myFloatNum = 8.99f;
 char myLetter = 'A';
 Boolean myBool = false;
 String myText = "Hello World";

9) byte, short, int, long, float, double, boolean and char are called:

data types.

Answer: primitive data types.

10) Use the correct method to find the **highest value** of x and y.

```
int x = 5;  
int y = 10;  
Math. (x, y);
```

Answer: int x = 5;
 int y = 10;
 Math. Max(x, y);

HTML EXERCISES:

1) Add a "tooltip" to the paragraph below with the text "About me".

<p ="About me">George is a web developer.</p>

ANSWER: <p title ="About me">George is a web developer.</p>

2) Set the size of the image to 250 pixels wide and 400 pixels tall.

```

```

Answer: ``

3) Make the element below into a link that goes to "https://www.mywebpage.com".

```
<a "https://www.mywebpage.com">This is a link</a>
```

Answer: `This is a link`

4) Specify an alternate text for the image.

Alternate text is useful when the image cannot be displayed, like when the page is read by a screen reader.

```

```

Answer: ``

5) Add six headings to the document with the text "Hello".

Start with the most important heading (the largest) and end with the least important heading (the smallest).

```
<html>
```

```
<body>
```

```
</body>
```

```
</html>
```

```
\
```

Answer:

```
<h1>Hello</h1>
```

```
<h2>Hello</h2>
```

```
<h3>Hello</h3>
```

```
<h4>Hello</h4>
```

```
<h5>Hello</h5>
```

```
<h6>Hello</h6>
```

6) Use the correct HTML to make the text below into a link to "default.html".

```
  >Visit our HTML tutorial. 
```

Answer: `Visit our HTML tutorial.`

7) Use CSS to remove the underline from the link.

```
<a href="html_images.asp" style=">HTML Images</a>
```

Answer: `HTML Images`

8) Use the correct HTML attribute to make the link open in a new window.

```
<a href="html_images.asp" >HTML Images</a>
```

Answer: `HTML Images`

9) Use the correct HTML to make the image become a link to "default.html".

```
   
  

```

Answer: `

`

10) Add a "tooltip" to the link. The "tooltip" should say "Home".

```
<a href="default.html" >Back to Home</a>
```

Answer: `Back to Home`

C# EXERCISES

1) Insert the missing part of the code below to output "Hello World!".

```
static void  (string[] args)
{
      ("Hello World!");
}
```

Answer: static void Main(string[] args)

```
{
    Console.WriteLine("Hello World!");
}
```

2) Create a variable named `myNum` and assign the value 50 to it.

```
  =  ;
```

Answer: `int myNum = 50;`

3) Display the sum of `5 + 10`, using two variables: `x` and `y`.

***Notice the difference between C# and Java

```
  =  ;
int y = 10;
Console.WriteLine(x + y);
```

Answer: `int x = 5;`
 `int y = 10;`
 `Console.WriteLine(x + y);`

***Notice the difference between C# and Java

4) Fill in the missing parts to get user input, stored in the variable `userName`:

```
Console.WriteLine("Enter username:");
 userName = Console. ;
Console.WriteLine("Username is: " + userName);
```

Answer: `Console.WriteLine("Enter username:");`
 `string userName = Console.ReadLine();`
 `Console.WriteLine("Username is: " + userName);`

5) Fill in the missing parts to print a number put in by the user:

```
Console.WriteLine("Think of a number:");  
int myNum = Convert. (Console.  
Console.WriteLine("Your number is: " + myNum);
```

Answer: Console.WriteLine("Think of a number:");
 int myNum = Convert.ToInt32(Console.ReadLine());
 Console.WriteLine("Your number is: " + myNum);

6) Print `i` as long as `i` is less than 6.

```
int i = 1;  
 (i  6)  
{  
    Console.WriteLine(i);  
      
}
```

Answer: int i = 1;
 while (i < 6)
 {
 Console.WriteLine(i);
 i++;
 }

7) Use the `do/while` loop to print `i` as long as `i` is less than 6.

```
int i = 1;  
  
{  
    Console.WriteLine(i);  
      
}  
 (i  6);
```

Answer: int i = 1;
 do
 {
 Console.WriteLine(i);
 i++;
 }
 while (i < 6);

8) Use a for loop to print "Yes" 5 times:

```
 (int i = 0; i < 5; )  
{  
  Console.WriteLine("Yes");  
}
```

Answer: for (int i = 0; i < 5; i++)
 {
 Console.WriteLine("Yes");
 }

9) Stop the loop if i is 5.

```
for (int i = 0; i < 10; i++)  
{  
  if (i == 5)  
  {  
    ;  
  }  
  Console.WriteLine(i);  
}
```

Answer: for (int i = 0; i < 10; i++)
 {
 if (i == 5)
 {
 break;
 }
 Console.WriteLine(i);
 }

PYTHON EXERCISES

- 1) Insert the missing part of the code below to output "Hello World".

```
("Hello World")
```

Answer: `print("Hello World")`

- 2) Comments in Python are written with a special character, which one?

```
This is a comment
```

Answer: `#`This is a comment

- 3) Create a variable named `carname` and assign the value `Volvo` to it.

```
 = ""
```

Answer: `carname = "Volvo"`

- 4) Which characters are illegal in the variable name:

```
2my-first_name = "John"
```

Answer: `2my-first_name = "John"`

- 5) Insert the correct syntax to assign the same value to all three variables in one code line.

```
x  y  z  "Orange"
```

Answer: `x = y = z = "Orange"`

- 6) The following code example would print the data type of x, what data type would that be?

```
x = 5  
print(type(x))
```

Answer: `int` (for `integer`)

- 7) The following code example would print the data type of x, what data type would that be?

```
x = "Hello World"  
print(type(x))
```

Answer: `str` (for `string`)

- 8) Use the correct membership operator to check if "apple" is present in the fruits object.

```
fruits = ["apple", "banana"]  
if "apple"  fruits:  
    print("Yes, apple is a fruit!")
```



```
Answer:      fruits = ["apple", "banana"]
             if "apple" in fruits:
                 print("Yes, apple is a fruit!")
```

9) Use the correct comparison operator to check if 5 is **not equal** to 10.

```
if 5  10:
    print("5 and 10 is not equal")
```

```
Answer: if 5 != 10:
        print("5 and 10 is not equal")
```

10) Use the correct logical operator to check if at least one of two statements is **True**.

```
if 5 == 10  4 == 4:
    print("At least one of the statements is true")
```

```
Answer:      if 5 == 10 or 4 == 4:
                 print("At least one of the statements is true")
```