CORE 2: PROBLEM SOLVING AND PYTHON PROGRAMMING

SUBJECT CODE: 18BCS14C

UNIT I

CHAPTER I

PROBLEM SOLVING

1.1Introduction to Problem Solving

1.1.1. Problem Analysis

The key dimensions basis on which an appropriate method is applied are as follows

- Decomposable/non decomposable :

  Analyze whether the problem is decomposable, i.e, whether it can be broken into sub-problems.

  If cannot be decomposed then such problem is non-decomposable.

- Solution steps- can/cannot be ignored:

It is important to determine if we can or cannot ignore a few steps while solving a problem. On the basis of the key dimensions, a problem can be categorized into there types. They are

a) Ignorable

b) Recoverable

c) Irrecoverable

a) Ignorable:

a few steps can be ignored since they are not extremely important

b) Recoverable:

The problems are those in which previously executed steps can be easily backtracked.

c) Irrecoverable:

we cannot backtrack previously executed steps.

- Predictable/unpredictable:

It is important to know whether there will be expected/unexpected output after applying a particular input.

- Good Solution:

Absolute/Relative: It is important to determine whether we are trying to achieve an optimum solution or idea to identify the best solution. On the basis of this, appropriate techniques can be applied to achieve the best solution.

1.1.2 FORMAL DEFINITION OF PROBLEM

A problem can be defined as a gap between the actual and desired conditions.

It is an unfulfilled customer need.

Example:

There will be problem when a standard is not achieved or customer's requirements are not fulfilled.

If goal is 100% but attained only 70% , this implies that we have not met the actual standard.

1.2 PROGRAM DEVELOPMENT

a. PLANNING

- The Planning stage is the stage where programmers decide and define the tasks that should be performed by the program.

- This involves identifying the problem, defining the problem in simpler terms, identifying the program requirements (inputs, outputs the formulas needed), and finally, planning for the solution.

b. PROGRAM DESIGN

- *This stage deals with the flow of the program*, the way it should produce output and gather inputs, the interface or the working environment of the program, and how it should be used.
- In planning and designing the program, you need to have an understanding of the logic behind the programs performance.
- You also need to be knowledgeable in the theory, *algorithms*, *pseudo codes*, and *flowcharts*.

c. CODING

- It is the process of expressing the fully detailed algorithm to a standard high-level programming language.
- This stage of program development requires following the rules of format and syntax (vocabulary, grammar and punctuation.
- This stage requires an extensive knowledge in the *programming language user* will use.
- *Coding is the act of actual writing of the computer program.*
- It means generating the *"source code"* in the *language* of the programmer's choice.
- A computer program is generated through the *source code*.

d. TESTING & DEBUGGING

- The *Testing Stage* is where the programmer makes sure that the program performs the way it is intended.
- Programs are tested by running them and observing the output they produce.
- The process of correcting mistakes in a program is called *debugging*.

PROGRAM TESTING

- PROGRAM TESTING is the stage in which the programmer run their programs to discover if it is free from syntactical and logical errors thus making it capable of doing what it was intended for.
- testing a program requires three separate activities: desk checking, translation and debugging.

i. Desk Checking

- Desk Checking

Process of sitting in a desk and checking the source code, proofreading it for obvious syntax errors as well as looking for logical errors which are not immediately detectable.

- Types of errors

Syntax Errors

Violations of the rules of a specific programming language.

Logical Errors

Mistakes in the algorithm or program design which are not easily detected.

ii. Translation

- Translation is the conversion of the source code to the internal instructions that the computer required.
- During translation, another computer program called the translator

  checks for and finds all syntax errors which may be corrected once the computer issues diagnostic messages which inform the user of the errors.

iii. Debugging

- Debugging is the process of detecting, locating and correcting logic errors (bugs) by submitting a translated program to the computer for execution and seeing what happens.

- For this purpose, valid test data must be administered to see how the program will fare in a similar situation.

f. DOCUMENTATION

- It is the detailed description of a program's algorithm, design, coding method, testing and proper usage.
- Documentation usually includes the necessary information about the requirements of the program – the operating system, and hardware requirements needed for the program to run.
- *Documentation* also contains technical information such as where and who created the program, who contact when there's a problem with the program and instructions on the use and maintenance of the program.

g. MAINTENANCE

The final stage in programming is in maintaining or updating the program.

This is the stage where the programmer is tasked to keep the program running smoothly and updated with the developments and changes in the field where it is used.

1.3. MODULAR DESIGN

Modular programming (also called "top-down **design**" and "stepwise refinement") is a software **design** technique that emphasizes separating the functionality of a program into independent, interchangeable **modules**, such that each contains everything necessary to execute only one aspect of the desired functionality.

It Eliminates Effects Between Unrelated Things.

Design components that are: self-contained, independent, and have a single, well-defined purpose.

ADVANTAGES

- Minimize Complexity
- Reusability
- Extensibility
- Portability
- Maintainability

1.4. METHODOLOGY OF PROBLEM SOLVING

.

- The process of developing a solution consists of development of a structure chart, a pseudo code and flow chart.
- A structure chart is used to develop the whole program.
- A flow chart or a pseudo code is used to develop individual parts of a program. These parts are known as modules.

1.4.1 Structure Chart

- A chart is a hierarchy that shows the functional flow of a program.
- Large programs are complex structures comprising interrelated parts.
- A structure chart shows a logical breakdown of a program into different steps.
- Each step has separate modules that are related to different modules.

Pseudo code

Pseudo code is used to state an algorithm in English-like sentence.

A pseudo code is used by the professional programmers.

- It is easy to understand.
- These are used to depict the design of an algorithm.

Example:

The pseudo code for adding two numbers

i)Read a,b

ii)Add two numbers

Sum=a+b

iii)Result sum “the sum is”

iv)end

Flow Chart

- A flow chart is a graphical representation of the logical flow of data.

- It uses the standard graphical symbols to narrate the sequential process of a specific module.

- These steps must be followed while designing the whole program.

1.5. ALGORITHMS

- Algorithm is a set of instructions used for solving problems in a step-by-step manner.
- This step-by-step explanation of doing something is known as a algorithm.
- Algorithm is a finite and ordered sequence of steps.
- It is a description of a process independent of any programming language.
- It can be implemented in many different language by using different methods and programs.

Example 1

Sum of two nos

Step 1: Start

Step 2: read number n1 and n2

Step 3: sum= n1+n2

Step 4: write sum

Step 5: stop

Example 2

put a book in the box

Step 1: open the box

Step 2: pick up the book

Step 3: put the book in side the box

Step 4: close the box

An algorithm can be written in two ways:

- Pseudo code
- Flow chart

Example:

Pseudo code for counting people in a room

Step 1 : Let p=0

Step 2 : For each person in a room, set p = p+1

Step 3 : will repeat until every person in the room has been counted.

1.6. FLOW CHARTS

- A flow chart is a simple diagram that illustrates a sequence of operations to be performed for obtaining a solution.
- It allows you to identify the actual sequence of events in a process that any product or service follows.
- Flow charts are very effective in understanding how a process works.
- In a flow chart, the flow of data is represented by arrows.
- It is a graphical representation of the algorithmic solution of a problem.
- Flow charts are also known as process maps that can be used to identify:

I. Flow of information
II. Number of steps in a process
III. Branches in a process
IV. Inter-dependent operations.

1.6.1. FLOW CHARTS SYMBOLS

There are six basic symbols in flow charts. They are

- Start/Stop
- Input/Output
- Process
- Decision symbol
- Flow line
- Connector

i. Start/Stop

- Every flow chart has starting and a terminating point.
- The symbol that is used for both the starting and terminating points is a rounded rectangle with round corners.
- It is called a “terminal”.

ii. Input/Output

- Every time you take input from a user and return an output to the user.
- An input /output symbol is used in the flow chart.
- The symbol that is used for both input and output related actions is a parallelogram.

iii. Process

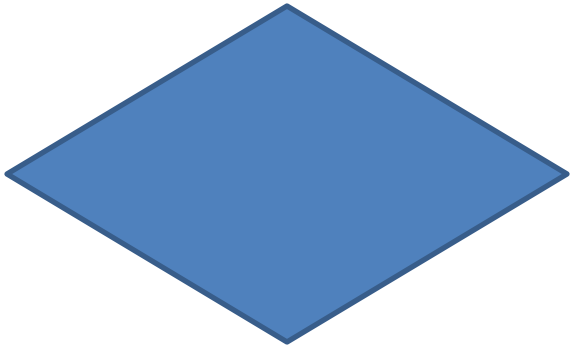
- If you are running a processing instructions, you need to use a rectangular box in the flow chart.

iv. Decision symbol

- In a flow chart , a decision symbol is used for answering questions in the form of either

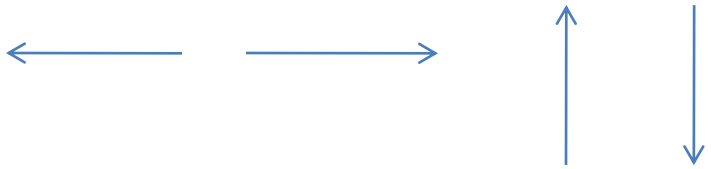
  yes/no or true/false.

- Each answer can lead to a different path in the flow chart.



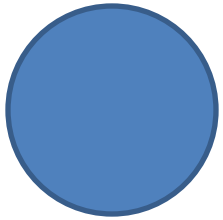
v. Flow lines

- Flow lines depict the direction of a flow in a flow chart.
- There are four types of flow lines.
- Flow lines can depict a left, right, top and bottom directions.



vi. Connector

- A connector connects different steps in a flow chart that are on different pages and gives a sense of continuation.
- Generally, it is used in extremely complex flow charts and it is denoted by a small circle.

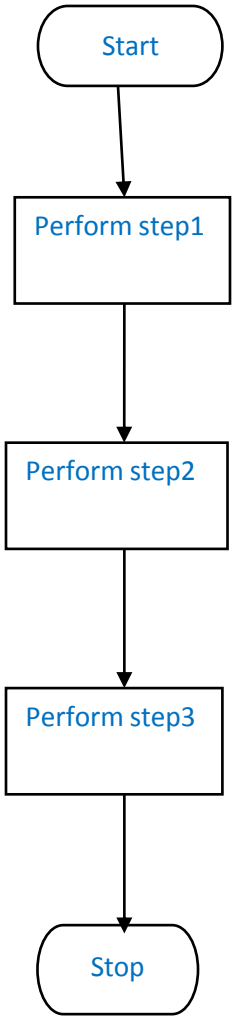


1.6.2. *FLOW CHARTS CONVENTIONS*

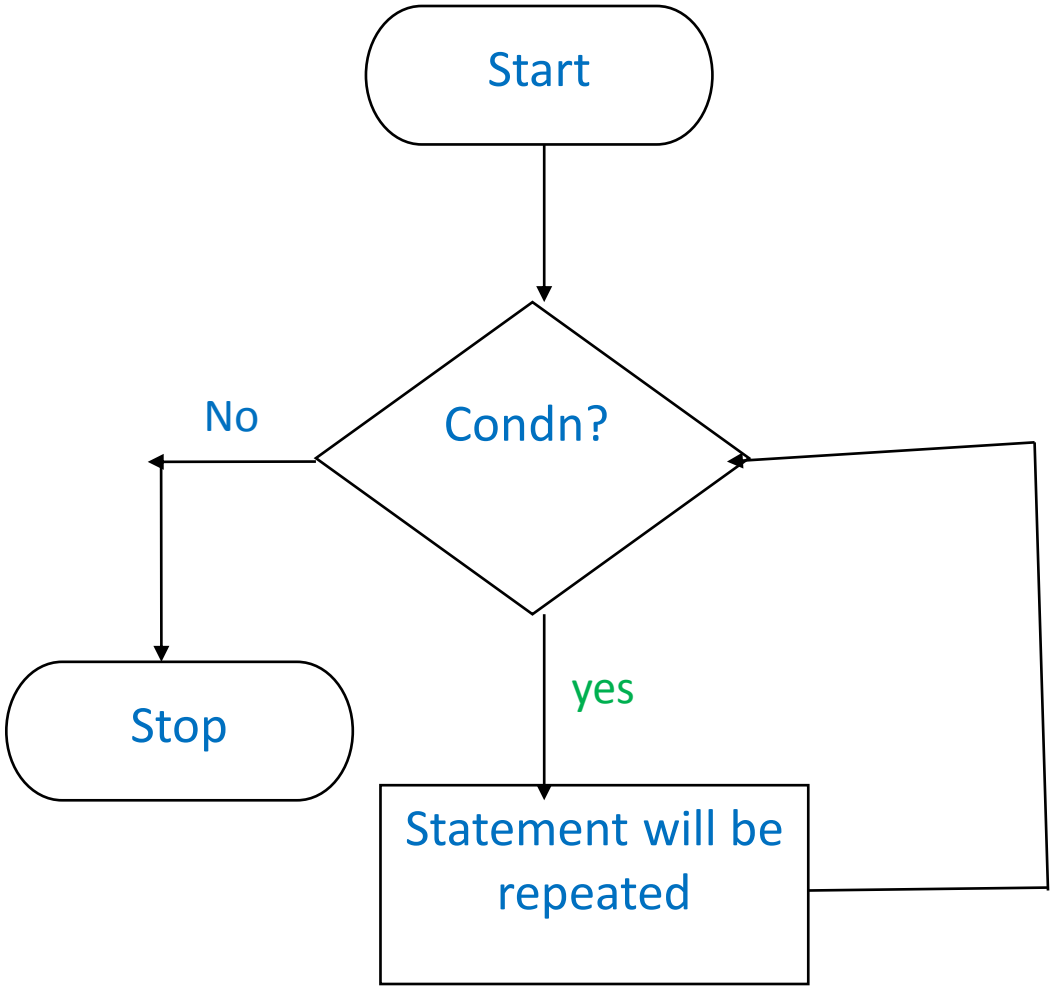
There are three different types of flow chart structures.

- Sequential structure
- Repetition structure
- Selection structure

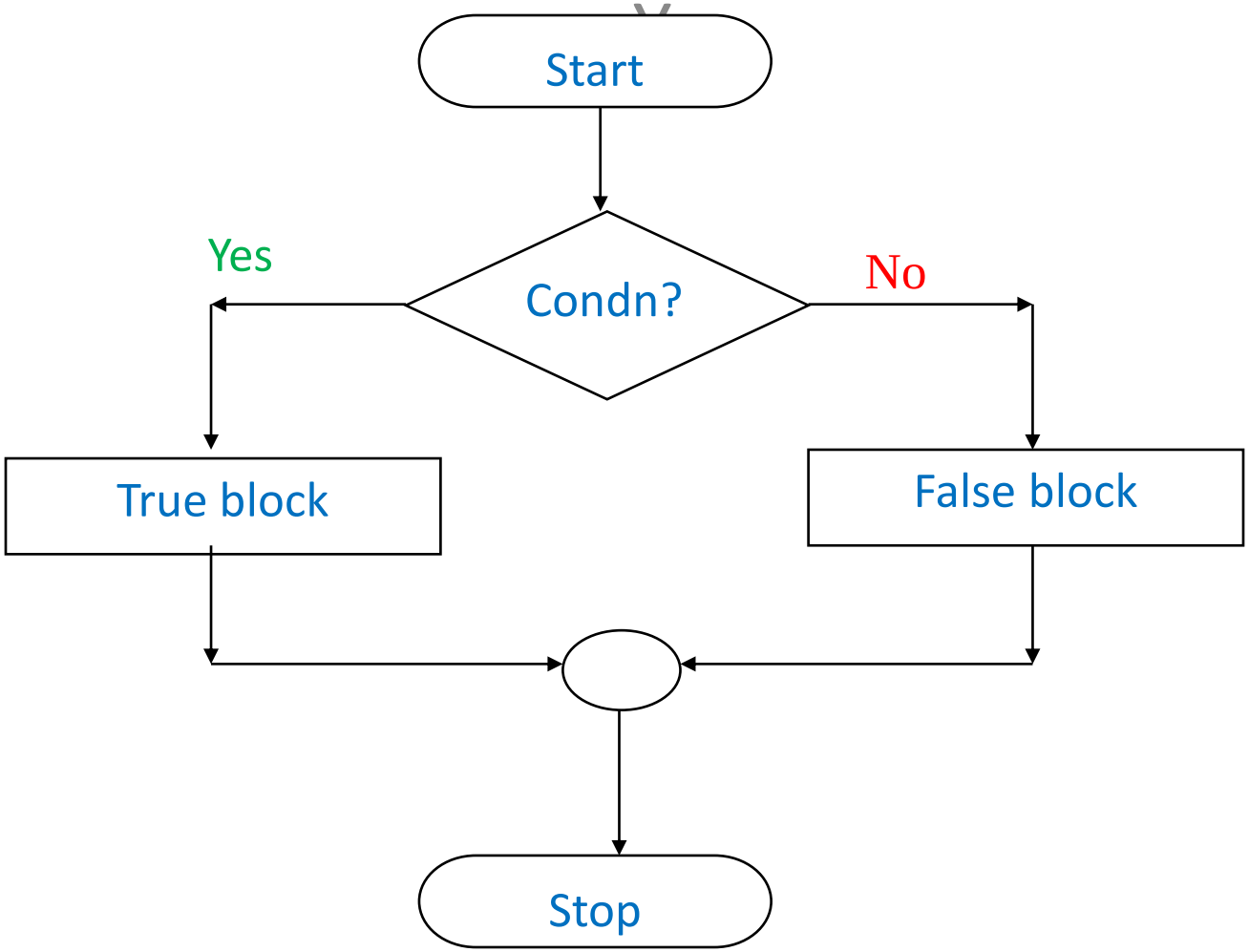
1. Sequential structure



2. Repetition structure



3. Selection structure



1.7. **PROGRAMMING LANGUAGES**

A computer language is used to make computer understand what the user wants to say.

A program is a set of instructions by which the computer comes to know what is to be done.

A programming language is a coding language used by the programmer to write the instructions that a computer can understand and act on.

There are three types of programming language.

- High-level language
- Assembly language
- Machine language

a. High-level language

High-level languages use English-like languages allowing the programmer to focus on application problem.

High-level languages are converted into machine level language using converting software called compiler.

It is a computer programming language that does not requires great efforts from the programmer.

It is called High-level language because it is closed to the user.

The first High-level language used was FORTRAN, which was followed by COBOL.

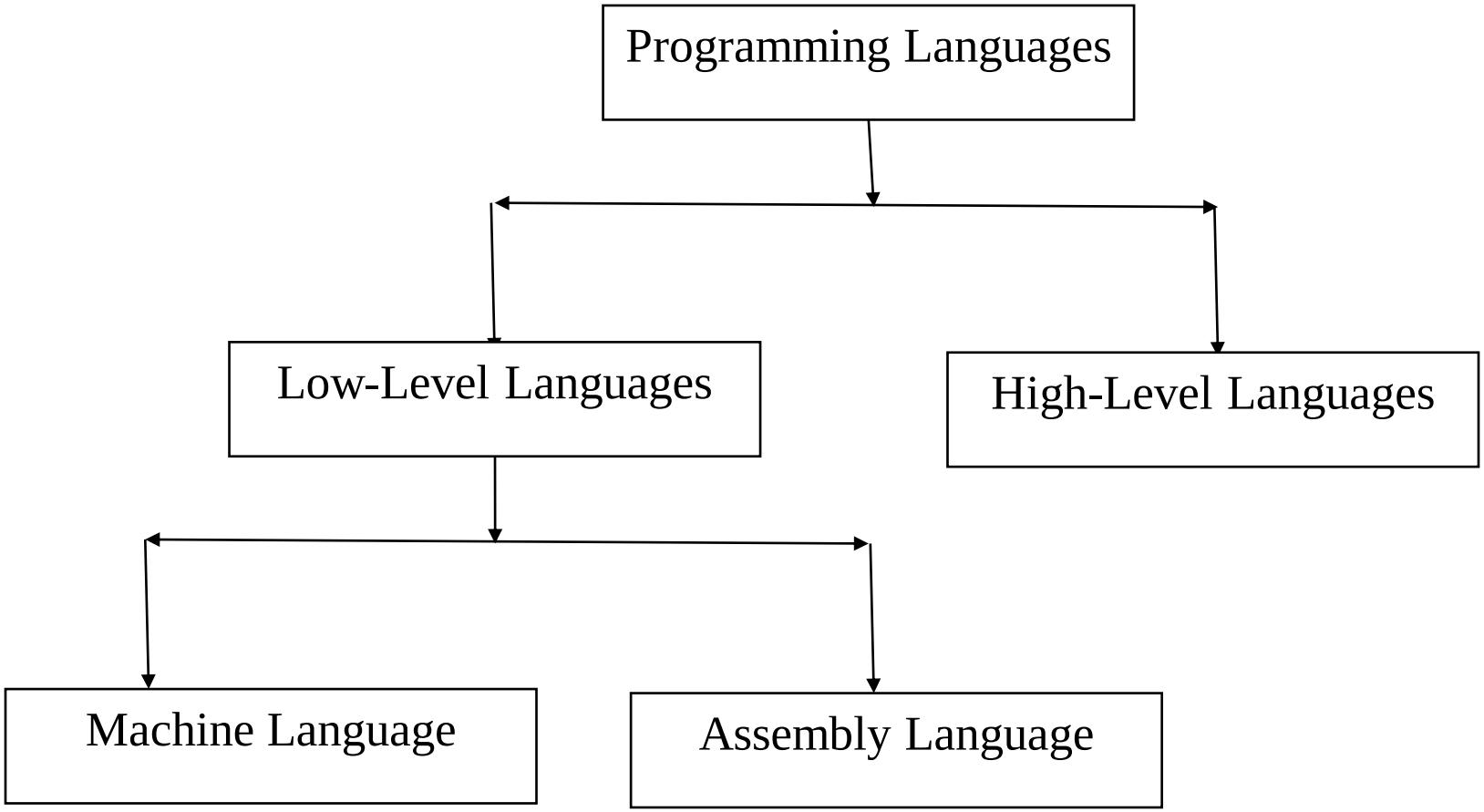
b. Assembly language

- It is a low-level language.
- It is more machine friendly and requires more efforts from the programmer.
- Assembly language closely resembles machine language.
- Symbols and mnemonics are used in this language to represent various machine language instructions.
- Assembly languages are converted into machine level language using converting software called assembler.

c. Machine language

- Machine language consisting of 0s and 1s.
- The computer can understands only 0s and 1s because it is made of switches, transistors, and other electronic devices which can only be in the state of either on or off.
- The on state is represented by 1 and off set 0.
- Machine language is low-level language and is more machine friendly.
- The language is known as Machine language because it is close to the machine.

Programming Languages



1.8. **PROGRAM DEVELOPMENT ENVIRONMENT**

In computer **program** and **software** product **development**, the **development environment** is the set of processes and **programming** tools used to create the **program** or **software** product.

The term may sometimes also imply the physical **environment**.

An integrated development environment is one in which the processes and tools are coordinated to provide developers an orderly interface to and convenient view of the development process (or at least the processes of writing code, testing it, and packaging it for use).

What is Integrated Development Environment (IDE)?

IDE stands for Integrated Development Environment.

IDE is basically a software pack that consist of equipment's which are used for developing and testing the software.

A developer throughout SDLC uses many tools like editors, libraries, compiling and testing platforms.

IDE helps to automate the task of a developer by reducing manual efforts and combines all the equipment's in a common framework.

If IDE is not present, then the developer has to manually do the selections, integrations and deployment process.

IDE was basically developed to simplify the SDLC(Software Development Life Cycle) process, by reducing coding and avoiding typing errors.

In contrast to the IDE, some developers also prefer Code editors.

Code Editor is basically a text editor where a developer can write the code for developing any software.

Code editor also allows the developer to save small text files for the code.

In comparison to IDE, code editors are fast in operating and have a small size.

In fact code editors possess the capability of executing and debugging code.

1.8.1. IDLE

IDLE (Integrated Development and Learning Environment) is a default editor that comes with Python.

This software helps a beginner to learn Python easily.

IDLE software package is optional for many Linux distributions. The tool can be used on Windows, macOS, and Unix.

Features:

- Search multiple files
- It has an interactive interpreter.
- Supports smart indent, undo, call tips, and auto-completion.
- The user can search and replace within any window.

Download Link: https://docs.python.org/3/library/idle.html

What is the difference between IDE and TEXT EDITOR?

IDE and Text Editor can be used in the place of each other for developing any software.

Text editor helps the programmer for writing scripts, modifying code or text etc.

But with IDE a programmer can perform several other functions as well like

running and executing the code,
controlling the version,
debug,
interpreting,
compiling,
auto-complete feature,
auto linting function,
pre-defined functions
and in build terminal etc.

IDE can be considered as a development environment where a programmer can write the script, compile and debug the completing process.

IDE also has an integrated file management system and deployment tool.

IDE provides support to

- SVN(***Subversion)***,
- CVS(client-***server*** system),
- FTP,
- SFTP,
- framework etc.

Basically, a Text editor is a simple editor to edit the source code and it does not possess any integrated tools or packages.

Advantage of Text editor

It allows modifying all types of files rather than specifying any particular language or types.

Both play an important role in their respective situations when used.

Top Python IDEs and Code Editors Comparison

IDE	User Rating	Size in MB	Developed in
PyCharm	4.5/5	BIG	JAVA, PYTHON
Spyder	May 4, 2018	BIG	PYTHON
PyDev	4.6/5	MEDIUM	JAVA, PYTHON
Idle	4.2/5	MEDIUM	PYTHON
Wing	May 4, 2018	BIG	C, C++, PYTHON

CHAPTER 2

INTRODUCTION TO PYTHON

2.1. INTRODUCTION

- **Python** is an interpreted, high-level, general-purpose programming language.
- Created by Guido van Rossum and first released in 1991.
- Python's design philosophy emphasizes code readability .
- Python is dynamically typed and garbage-collected.
- It supports multiple programming paradigms , including structured (particularly, procedural), object-oriented, and functional programming.
- Python is often described as a "batteries included" language due to its comprehensive standard library.

- The syntax of the python programs can express concepts in fewer lines as compared to programs in C,C++ and JAVA.

- It also supports automatic memory management and large standard library and numerous set of third party libraries.

- Python can be used on almost all operating system because its interpreter is available for many operating system.

- Python is free and open-source software.

- It generally has a community-based development.

2.2. HISTORY

- Python was conceived in the late 1980s as a successor to the ABC language.

- Python 2.0, released in 2000, introduced features like list comprehensions and a garbage collection system with reference counting.

- Python 3.0, released in 2008, Python 2 code does not run unmodified on Python 3.

- Python interpreters are available for many operating systems.

- A global community of programmers develops and maintains CPython, a free and open-source reference implementation.

- A non-profit organization, the Python Software Foundation, manages and directs resources for Python and CPython development.

2.3. PYTHON OVERVIEW

Python is an

Interpreted

High-level language and

General purpose programming language

open source software.

FEATURES:

The code written in python is automatically compiled to byte code and executed.

Extending python with C or C++.

Python supports lot of features such as nested code blocks, functions, classes, modules and packages.

Python is an object-oriented programming language.

It has many built-in data types: strings, lists, tuples, dictionaries, etc.,

It supports control statements such as if, if-else, if-eif-else, while, iterative for, etc.,

It has functions, classes, modules and packages.

2.3. GETTING STARTED WITH PYTHON

There are three ways of staring python:

- Running a script written in python. → text editor requires

- Using Graphical user interface (GUI) from an Integrated Development environment (IDE).→ GUI application

- Employing an interactive approach.→ command line interpreter

2.3.1. INSTALLING PYTHON INTERPRETER

Use the following link to download the python installer.

https://www.Python.org/downloads/

2.3.1.1. INSTALLING ON WINDOWS OS

Python is now installed in our computer's directory.

Steps.

Step 1: Click start button.

Step 2: Go to all programs.

Step 3:Search for python folder.

Step 4:Click python (command line).

This will start the python command line.

Once python starts, you can see the interpreter startup message, indication version and platform.

You can also be given the python interpreter prompt,

ie., “>>>” which is also known as python chevron prompt.

The “>>>” indicates that python interpreter is waiting for an expression or command.

The interacting environment where we are interaction with the Python interpreter is called the console or command shell.

2.4. PYTHON INTERPRETER, INTERACTIVE PROMPT AND PROGRAM EXECUTION

- The interactive shell is between the user and the operating system (e.g. Linux, Unix, Windows or others).
- Instead of an operating system an interpreter can be used for a programming language like Python as well.
- The Python interpreter can be used from an interactive shell.
- The interactive shell is also interactive in the way that it stands between the commands or actions and their execution.
- In other words, the shell waits for commands from the user, which it executes and returns the result of the execution.
- Afterwards, the shell waits for the next input.

- A shell in operating systems lies between the kernel of the operating system and the user.
- It's a "protection" in both directions.
- The user doesn't have to use the complicated basic functions of the OS but is capable of using the interactive shell which is relatively simple and easy to understand.
- The kernel is protected from unintentional and incorrect usages of system functions.
- Python offers a comfortable command line interface with the Python Shell, which is also known as the "Python Interactive Shell".
- . The Python interpreter can be invoked by typing the command "python" without any parameter followed by the "return" key at the shell prompt.

2.4.1. PYTHON INTERPRETER

- Python interpreter is a kind of program that executes other program.
- Python interpreter reads a python source code and carries the instructions it contains.
- The interpreter is a layer of software logic between code and the computer hardware on machine.
- When the python package is installed in a computer, it generates a number of components-minimally, on interpreter and support library.
- Depending on how you use it, the python interpreter makes the form of an executable program, or a set of libraries linked into another program.

- First, the python interpreter should install in your computer to execute the python code.
- Python installation may vary per platform.

1. Window users fetch and run a self-installing executable file, which puts python on their machine. Simply double click and say yes to next all prompt.

2. Linux and UNIX users typically either install python from RPM file or compile it from its full source-code distribution package.

3. Other platforms have installation techniques relevant to that platform. For instance, files are synched on palm pilots.

2.4.2. **PROGRAM EXECUTION**

There are can be two different views in program execution.

1. The programmers view
2. Python's view

The programmers view

- Python programs are just a text file containing python statements.
- Python programs files are given names that have extension as ".py", technically, this naming scheme is required only for filters that are imported.
- Python program files may be launched by command lines, by clicking those icons, and with other standard techniques.

Python’s view

❖The code converted to byte code and then routed to virtual machine.

1. Byte code conversion

➢Python compiler compiled source code into byte code.

➢Byte code can be much quicker than the original source code.

➢Byte code file contain the extension as “.pyc”.

➢Python saves the byte code file for future reference.

➢ If the python code is executed next time then the python will load .pyc file and skip the compilation step, as long as source code is modified.

- Python automatically checks the time stamps of source code and byte code files to know when it must recompile.

- Python cannot write the byte code file to your machine then the program still works.

- I.e., the byte code is generated in memory and simply discarded on program exit.

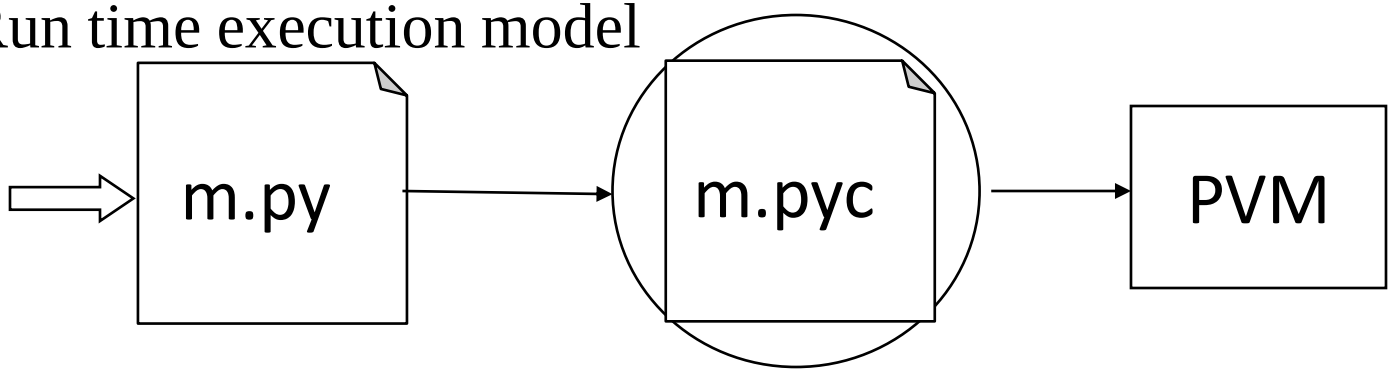
- Byte code files are also one way to skip python programs.

- Python will execute the .pyc file without the presence of source file .py.

2. Python Virtual Machine

- Once the python program compiled to byte code, it is shipped off for an execution to something known as Python Virtual Machine (PVM).
- The PVM is a big loop that iterate iterates through byte code instructions, one by one, to carry out their operations.
- The PVM is run time engine of python.
- It is always available in python system.
- The components are truly run scripts.
- Technically, it's just the last step of what is called the python interpreter.

Run time execution model



2.4.3. INTERACTIVE PROMPT

The interactive prompt runs code and echoes results but it does not save code in a file.

We cannot do the group of statements together.

The interactive prompt runs out to be a great place to both experiment with the language and text program files.

a. Experimenting

The interactive prompt is perfect place to experiment with the language

Example:

Interactive Python Console

>>> 5

5

>>> 5.5

5.5

>>> ""hello”

hello

b. Testing

- The interactive interpreter is a place to test code written in files.
- We can import module files interactively and run test on the tools they define by typing calls at the interactive prompt.
- The interactive prompt is a place to test program components regardless of their source.
- We can import and test function and classes in python files, type calls to link in c functions, exercise Java classes under Jython and more.
- Partly because of its interactive nature, python supports an experiments and exploratory programming style we will find convenient when getting started.

c. Using the interactive prompt

i. Type the python commands only.

Type the python code at the python prompt, not system commands.

ii. Print statements are required only in files.

Because, the interactive interpreter is automatically prints the results of expressions. You do not need to type complete print statement interactively.

iii. Don’t indent at the interactive prompt (yet).

When typing python programs, either interactively a into a text file, be sure to start all un nested statements.

2.5. **IDLE USER INTERFACE**

- IDLE is a graphical user interface for doing Python development, and is a standard and free part of the Python system.

- It is usually referred to as an Integrated Development Environment (IDE), because it binds together various development tasks into a single view.

- IDLE is a corruption of IDE, named in honor of Monty Python member Eric Idle.

- In short, IDLE is a GUI that lets you edit, run, browse, and debug Python programs, all from a single interface.

- Moreover, because IDLE is a Python program that uses the Tkinter GUI toolkit,
- it runs portably on most Python platforms: MS Windows, X Windows (Unix, Linux), and Macs.
- For many, IDLE is an easy-to-use alternative to typing command lines, and a less problem-prone alternative to clicking on icons.

References

- Allen B Downey. “Think Python: How to Think like a Computer Scientist”, Green Tea Press,version 2.0.17.
- E. Balagurusamy , “Introduction to Computing and Problem Solving Using Python”,Mc Graw hill education.
- Guido van Rossum and Fred L. Drake Jr, “An Introduction to Python – Revised and updated for Python 3.2”, Network Theory Ltd., 2011.
- John V Guttag, “Introduction to Computation and Programming Using Python”, Revised and expanded Edition, MIT Press , 2013
- Mark Lutz, “Learning Python :Powerful object oriented programming”, o’Reilly,4Th edition.
- www.w3school.com
- https://www.programiz.com/python-programming.
- https://www.tutorialspoint.com
- https://hzltenedero.files.wordpress.com/2015/07/program-development-process.png
- Wikipedia
- https://searchcio.techtarget.com/definiton -posted by Margaret Rouse