Module – 1 (SDLC)

1. What is Software?

- Software is a set of computer programs and associated documentation and data. This is in contrast to hardware, from which the system is built and which actually performs the work. At the lowest programming level, executable code consists of machine language instructions supported by an individual processor—typically a central processing unit (CPU) or a graphics processing unit (GPU).
- The two main categories of software are application software and system software. An application is software that fulfills a specific need or performs tasks. System software is designed to run a computer's hardware and provides a platform for applications to run on top of.
- Software is typically stored on an external long-term memory device, such as a hard drive or magnetic diskette. When the program is in use, the computer reads it from the storage device and temporarily places the instructions in random access memory (RAM). The process of storing and then performing the instructions is called "running," or "executing," a program.

- The two major types of computer software are:
    - **System Software**
        - System software helps the user, hardware, and application software interact and function with each other. System software acts as a mediator or middle layer between the user and the hardware.
    - **Application software**
        - Application software is software that helps an end user complete tasks such as doing research, taking notes, setting an alarm, designing graphics, or keeping an account log.

2. What are the types of Applications?

- Application software is software that helps an end user complete task such as doing research, taking notes, setting an alarm, designing graphics, or keeping an account log.
- Application software lies above the system software and is different from system software in that it's designed for the end use and is specific in its functionality. This type of software is sometimes referred to as non-essential software because it's installed and operated based on the user's needs.
- Application software can also be classified depending on how much it costs and how easily it can be accessed. Here are some examples of application software:

- Word Processing Software
- Spreadsheet Software
- Presentation Software
- Multimedia Software
- Web Browsers
- Educational Software
- Graphics Software
- Freeware
- Shareware
- Simulation Software
- Open Source
- Closed Source

3. What is programming?

- Programming is, quite literally, all around us. From the take-out we order, to the movies we stream, code enables everyday actions in our lives. Tech companies are no longer recognizable as just software companies — instead, they bring food to our door, help us get a taxi, influence outcomes in presidential elections, or act as a personal trainer.
- programming is giving a set of instructions to a computer to execute. If you've ever cooked using a recipe before, you can think of yourself as the computer and the recipe's author as a programmer.

- **Computer programming** is the process of performing particular computations (or more generally, accomplishing specific computing results), usually by designing and building executable computer programs. Programming involves tasks such as analysis, generating algorithms, profiling algorithms' accuracy and resource consumption, and the implementation of algorithms (usually in a particular programming language, commonly referred to as **coding**).
- Programming, in the context of computer science and technology, refers to the process of designing, writing, and testing instructions for a computer to follow. These instructions are typically written in a programming language, which is a set of precise and structured commands that can be understood by a computer.

  - **Problem Solving**: Programming involves breaking down complex problems into smaller, more manageable tasks and then devising algorithms (step-by-step instructions) to solve those tasks. Programmers use their problem-solving skills to design efficient and effective solutions.
  - **Programming Languages**: Programming languages are used to write code that computers can understand. These languages vary in terms of syntax (grammar rules) and semantics (meaning of commands).

- Example of programming languages include Python, Java, C++, and JavaScript.
- **Writing Code**: Programmers write code by typing out instructions in a text editor or an integrated development environment (IDE). Code is typically organized into functions, classes, or modules to create structured and maintainable programs.
- **Testing and Debugging**: After writing code, programmers test it to ensure it works as intended. Debugging involves identifying and fixing errors (bugs) in the code that cause unexpected behavior or errors.
- **Algorithms**: Algorithms are step-by-step procedures or sets of rules that define how a task is to be performed. Developing efficient algorithms is a crucial part of programming to ensure that tasks are completed as quickly and accurately as possible.
- **Data Structures**: Programming often involves working with data. Data structures, such as arrays, lists, and trees, are used to organize and manipulate data effectively.

- Programming is a versatile skill with applications in a wide range of fields, including web development, mobile app development, artificial intelligence, data analysis, scientific research, and much more. It plays a central role in the creation of software and computer programs that power the modern digital world.

4. What is python?

- Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It is high level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for rapi application development, as well as for use as a scripting or glue language to connect existing components together.
- Python is simple, easy to learn syntax emphasizes readability and therefor reduces the cost of program maintenance. Python supports moduels and packages, which encourages program modularity and code reuse.
- The python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.
- Often, programmers fall in love with python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging python program is easy a bug or a bad input will never cause a segmentation fault.
- A souce level debugger allows inspection of local and global variables, evaluation of arbitrary expression, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in python itself, testifying to python's introspective power.
- On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test□debug cycle makes this simple approach very effective.