Module – 1 (SDLC)

Q-1. What is software ?

Ans. Software refers to a collection of programs, data, and instructions that enable a computer system to perform specific tasks or functions. It is a set of logical and organized instructions that tell a computer how to carry out various operations. Software can be categorized into two main types: system software and application software.

1. System Software: This type of software provides a foundation for the computer's operation and manages its resources. It includes the operating system, device drivers, firmware, and other low-level programs. The operating system (such as Windows, macOS, or Linux) controls the hardware and provides a user interface for interaction with the computer. System software ensures that different hardware components and software applications can work together efficiently.

2. Application Software: Application software is designed to perform specific tasks or provide functionality for users. It includes a wide range of programs such as word processors, spreadsheets, web browsers, image editors, games, and more. Application software allows users to accomplish diverse activities, from creating documents and managing finances to playing multimedia files and accessing the internet.

Software is created using programming languages, which are sets of instructions that are understood by computers. Programmers or software developers write the code that makes up software, and it is then compiled or interpreted to be executed by the computer. Software can be distributed in various forms, including physical media such as CDs or downloaded from the internet.

Q-2. What are the types of application ?

Ans. There are many types of applications, but they can generally be divided into two categories:

- General-purpose applications are designed to perform a wide range of tasks, such as word processing, spreadsheet analysis, and presentation creation. Examples of general-purpose applications include Microsoft Office, Google Docs, and Adobe Photoshop.
- Specialized applications are designed for a specific purpose, such as accounting, customer relationship management (CRM), or project management. Examples of specialized applications include QuickBooks, Salesforce, and Microsoft Project.

In addition to these two categories, there are also a number of other types of applications, such as:

- Web applications are hosted on a server and accessed over the internet. Examples of web applications include email, social media, and online banking.
- Mobile applications are designed for mobile devices, such as smartphones and tablets. Examples of mobile applications include games, productivity tools, and navigation apps.
- Enterprise applications are large, complex applications that are used by businesses. Examples of enterprise applications include ERP systems, CRM systems, and supply chain management systems.

Q-3. What is programing ?

Ans. Programming, also known as coding, refers to the process of writing and creating sets of instructions for a computer to execute. It involves using programming languages to communicate with a computer and provide step-by-step instructions for it to follow.

Programmers, or software developers, write code using programming languages such as Python, Java, C++, JavaScript, or Ruby. These languages have their own syntax and rules that dictate how the instructions are written.

The process of programming involves several steps:

1. Problem Solving: Programmers first identify a problem or task that they want the computer to solve or perform. They analyze the problem, break it down into smaller components, and design a plan or algorithm to solve it.

2. Writing Code: Programmers then write the code using the chosen programming language. They use the syntax and rules of the language to express the desired instructions for the computer to execute.

3. Testing and Debugging: After writing the code, programmers test it to ensure it functions as intended. They run the program, input data, and analyze the output to identify and fix any errors or bugs. This process is called debugging.

4. Compiling or Interpreting: Depending on the programming language, the code is either compiled or interpreted. Compilers convert the entire code into machine-readable instructions, creating an executable file. Interpreters, on the other hand, read and execute the code line by line.

5. Execution: Once the code is compiled or interpreted, the program is ready to be executed by the computer. The computer follows the instructions provided by the programmer and performs the desired tasks.

Programmers may work on various types of software, ranging from simple scripts and small applications to complex software systems or websites. They often collaborate with other programmers, use development tools, and follow coding best practices to write efficient, maintainable, and scalable code.

Programming is a creative and problem-solving field that empowers individuals to build software, automate tasks, develop new technologies, and solve real-world challenges using the power of computers.

Q-4. What is Python ?

Ans. Python is a high-level, versatile, and interpreted programming language. It was created by Guido van Rossum and first released in 1991. Python emphasizes readability and simplicity, making it a popular choice for beginners and experienced programmers alike.

Here are some key features and characteristics of Python:

1. Readability: Python uses a clean and easy-to-understand syntax, which makes the code highly readable. It emphasizes code readability through the use of indentation and a minimalistic approach.

2. Versatility: Python is a multipurpose language that can be used for a wide range of applications, such as web development, scientific computing, data analysis, artificial intelligence, machine learning, automation, and more.

3. Expressive and Concise: Python allows programmers to express concepts and ideas in fewer lines of code compared to other languages. This promotes a more efficient and concise coding style.

4. Extensive Standard Library: Python comes with a large standard library, providing a rich set of modules and functions that can be readily used to perform various tasks without needing external dependencies.

5. Third-Party Libraries: Python has a vast ecosystem of third-party libraries and frameworks, such as NumPy, Pandas, Django, Flask, TensorFlow, and many more. These libraries extend the functionality of Python and enable developers to build complex applications efficiently.

6. Interpreted Language: Python code is executed by an interpreter, which translates the code into machine-readable instructions at runtime. This allows for dynamic and interactive programming, where code changes can be made on the fly without the need for compilation.

7. Cross-Platform Compatibility: Python is available on various operating systems, including Windows, macOS, Linux, and others. This cross-platform compatibility ensures that Python programs can run on different environments without major modifications.

8. Strong Community and Support: Python has a large and active community of developers who contribute to its growth, create libraries, provide support, and share knowledge through forums, online communities, and extensive documentation.

Q5. How memory is managed in Python?

Ans. In Python, memory management is handled automatically by the Python runtime environment using a combination of techniques, including reference counting, garbage collection, and memory pooling. Let's explore each of these techniques:

1. Reference Counting: Python uses reference counting to keep track of the number of references to an object. Each object in memory has a reference count associated with it. When a reference to an object

is created, the reference count is incremented, and when a reference is deleted or goes out of scope, the reference count is decremented. When the reference count reaches zero, meaning there are no more references to the object, the memory occupied by the object is freed.

2. Garbage Collection: In addition to reference counting, Python also employs a garbage collector to handle cyclic references. Cyclic references occur when objects reference each other in a way that forms a cycle, making it impossible to reach them from the rest of the program. The garbage collector periodically identifies and collects such cyclically referenced objects that are no longer reachable. It frees up memory occupied by these objects, even though their reference count may not be zero.

3. Memory Pooling: Python utilizes memory pooling for small objects, such as integers and strings, to improve performance. Instead of allocating and deallocating memory for each small object individually, Python preallocates fixed-size blocks of memory, known as memory pools. When an object of a specific size is needed, Python assigns it from the corresponding memory pool. This reduces the overhead of memory allocation and deallocation, resulting in faster execution.

Python's memory management is transparent to developers, who generally don't need to manually allocate or deallocate memory. The automatic memory management in Python simplifies memory handling and helps prevent common memory-related errors, such as memory leaks and invalid memory accesses. However, understanding the underlying mechanisms can be useful for optimizing memory usage and performance in specific cases or when dealing with large-scale applications.

Q.6 What is the purpose continue statement in python?

Ans. In Python, the “continue” statement is used within loops to skip the current iteration and move to the next iteration without executing the remaining code within the loop for that particular iteration. It allows you to control the flow of the loop and skip certain iterations based on a specific condition.

The main purpose of the “continue” statement is to selectively bypass the execution of certain code within a loop while continuing with the next iteration. This is particularly useful when you want to skip some iterations based on certain conditions but continue looping over the remaining items.

Q.7 What are negative indexes and why are they used ?

Ans. Negative indexes in Python are used to access elements from the end of a sequence, such as string, list or tuple. Instead of starting from the beginning of the sequence with index 0, negative indexes start counting from the end of the sequence, with index -1 representing the last element.