Assignment Part-1

Q1. Why do we call Python as a general purpose and high-level programming language?

Sol)

General-purpose: Python is a versatile language that can be used for a wide range of applications. It supports various programming paradigms, including procedural, object-oriented, and functional programming. Python can be used for developing web applications, desktop applications, scientific computing, data analysis, artificial intelligence, machine learning, and more. Its flexibility and extensive libraries make it suitable for diverse tasks, making it a general-purpose language.

High-level: Python is considered a high-level language because it abstracts away many low-level details that are inherent in other programming languages. It provides constructs and syntax that are closer to human language, making it easier to read, write, and understand. Python allows programmers to focus more on solving problems rather than dealing with complex system-level details, memory management, or hardware interactions. This high-level nature enhances productivity and reduces the time required to develop software.

Q2. Why is Python called a dynamically typed language?

Sol)

Python is called a dynamically typed language because the type of a variable is determined and checked at runtime, rather than being explicitly declared or enforced at compile-time. In dynamically typed languages like Python, variables can hold values of different types during the execution of a program, and the type of a variable can be changed as needed.

Q3. List some pros and cons of Python programming language?

Sol)

Pros:

- Readability and simplicity
- Extensive standard library and third-party packages
- Cross-platform compatibility

Cons:

- Performance
- Mobile and browser limitations
- Memory consumption:

Q4. In what all domains can we use Python?

Sol)

- Web Development
- AI
- Data Analytics and Big Data
- Scientific Computing
- Internet of things
- Desktop applications
- Game Development

Q5. What are variable and how can we declare them?

Sol)

In Python, a variable is a named reference to a value stored in memory. It represents a location in memory that can hold data of different types, such as numbers, strings, lists, or objects. Variables allow us to store and manipulate data in our programs.

To declare or create a variable in Python, you simply assign a value to a name using the assignment operator (=). Here's the basic syntax for variable declaration

```
age = 25
```

Q6. How can we take an input from the user in Python?

Sol)

```
age = input("Enter your age: ")
print("Your age is " + age)
```

Q7. What is the default datatype of the value that has been taken as an input using input() function?

Sol)
The default data type of the value returned by the input() function in Python is a string.

```
age = input("Enter your age: ")
print(type(age))
```

```
<class 'str'>
```

Q8. What is type casting?

Sol)

Type casting, also known as type conversion, refers to the process of converting a value from one data type to another in a programming language. Type casting allows you to change the interpretation or representation of a value to suit the requirements of a specific operation or to store it in a different type of variable.

```
# Float to integer
a = 5.8
b = int(a)  # b is now 5 (integer)
```

Q9. Can we take more than one input from the user using single input() function? If yes, how? If no, why?

Sol)

No, the input() function in Python does not allow you to directly take multiple inputs using a single function call. The input() function is designed to prompt the user for a single value and return that value as a string.

Q10. What are keywords?

Sol)

In programming, keywords, also known as reserved words, are predefined words or identifiers that have special meanings and purposes within the programming language. These keywords are reserved by the language and cannot be used as variable names or other identifiers.

```
False      class      finally    is         return
None       continue   for        lambda     try
```

Q11. Can we use keywords as a variable? Support your answer with reason.

Sol)

No, we cannot use keywords as variables in Python or any programming language. Keywords are reserved by the language and have predefined meanings and purposes. They are specifically used to define the syntax and structure of the programming language.

```
if = 10  # SyntaxError: invalid syntax
```

In this case, you will encounter a SyntaxError because if is a keyword used for conditional statements, not for variable assignment.

Q12. What is indentation? What's the use of indentaion in Python?

Sol)

In Python, indentation refers to the whitespace at the beginning of a line of code. It is used to define the structure, grouping, and hierarchy of statements within the program. In most programming languages, indentation is primarily for readability purposes, but in Python, it serves a more significant role.

Q13. How can we throw some output in Python?

Sol)

To generate output in Python, you can use the print() function. The print() function is a built-in function that allows you to display text, variables, or other values on the console or output stream.

```
print("Hello, World!")
```

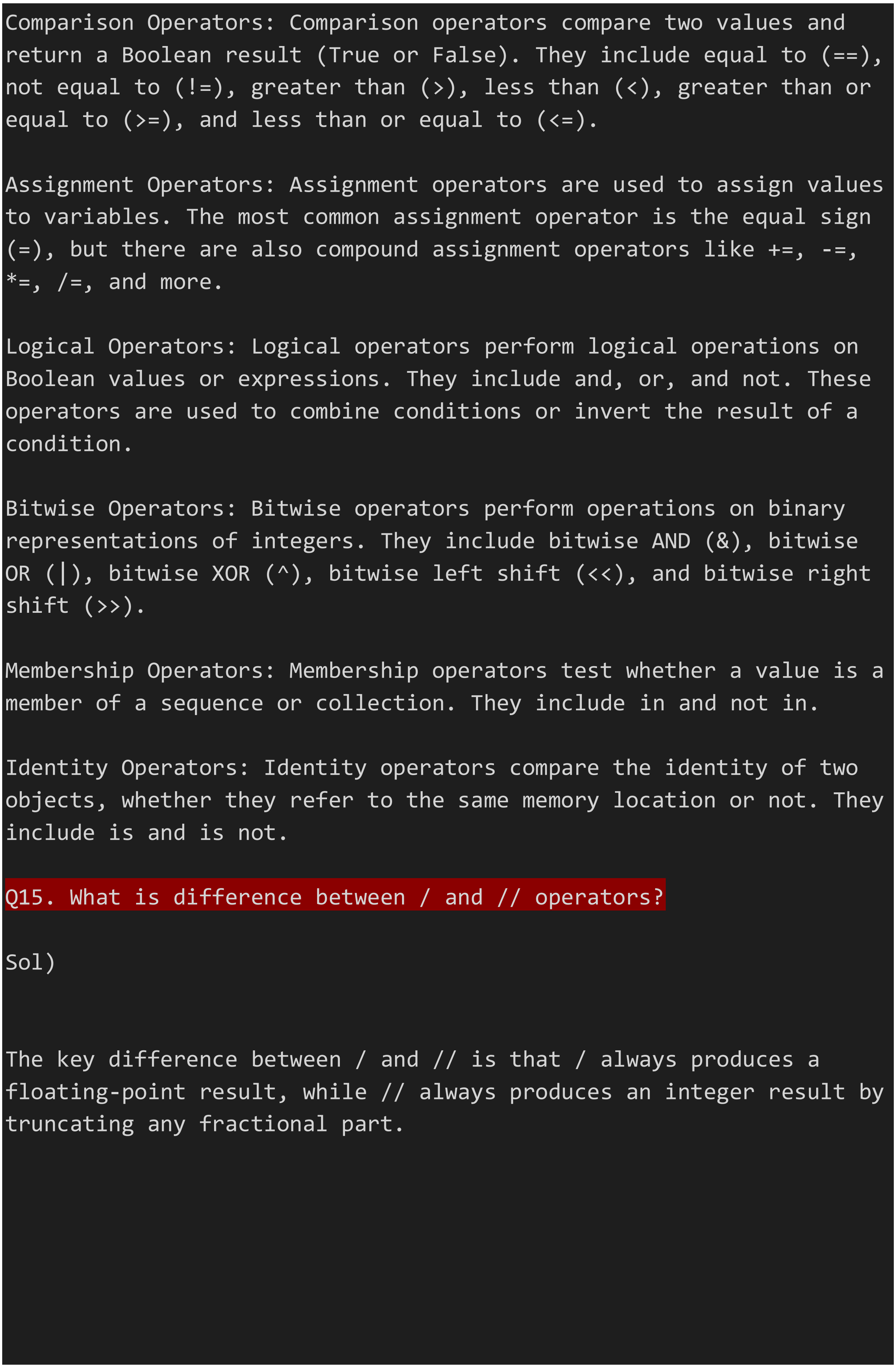
Q14. What are operators in Python?

Sol)

In Python, operators are special symbols or characters that perform specific operations on one or more operands (values or variables). Operators allow you to manipulate and perform computations on data, make comparisons, assign values, and more. They are an essential part of the Python language and are used extensively in programming.

Python supports a wide range of operators that can be classified into different categories based on their functionality. Here are the main categories of operators in Python:

Arithmetic Operators: Arithmetic operators perform mathematical operations on numeric operands and include addition (+), subtraction (-), multiplication (*), division (/), floor division (//), modulus (%), and exponentiation (**).



Comparison Operators: Comparison operators compare two values and return a Boolean result (True or False). They include equal to (==), not equal to (!=), greater than (>), less than (<), greater than or equal to (>=), and less than or equal to (<=).

Assignment Operators: Assignment operators are used to assign values to variables. The most common assignment operator is the equal sign (=), but there are also compound assignment operators like +=, -=, *=, /=, and more.

Logical Operators: Logical operators perform logical operations on Boolean values or expressions. They include and, or, and not. These operators are used to combine conditions or invert the result of a condition.

Bitwise Operators: Bitwise operators perform operations on binary representations of integers. They include bitwise AND (&), bitwise OR (|), bitwise XOR (^), bitwise left shift (<<), and bitwise right shift (>>).

Membership Operators: Membership operators test whether a value is a member of a sequence or collection. They include in and not in.

Identity Operators: Identity operators compare the identity of two objects, whether they refer to the same memory location or not. They include is and is not.

Q15. What is difference between / and // operators?

Sol)

The key difference between / and // is that / always produces a floating-point result, while // always produces an integer result by truncating any fractional part.

Q16. Write a code that gives following as an output.

Sol)

~~~~
```
iNeuroniNeuroniNeuroniNeuron
```
~~~~

~~~~
output = "iNeuroni" * 3 + "Neuron"
print(output)
~~~~

Q17. Write a code to take a number as an input from the user and check if the number is odd or even.

Sol)

~~~~
number = int(input("Enter a number: "))  # Take input from the user and convert it to an integer

if number % 2 == 0:
    print(number, "is even.")
else:
    print(number, "is odd.")
~~~~

Q18. What are boolean operator?

Sol)

Boolean operators in Python are used to perform logical operations on Boolean values or expressions. They take one or more Boolean operands and return a Boolean result (True or False) based on the outcome of the logical operation.

Q19. What will the output of the following?

Sol)

~~~~
```
1 or 0  -  1

0 and 0  - 0
~~~~

```
True and False and True    -    False

1 or 0 or 0    -    1
```

Q20. What are conditional statements in Python?

Sol)

Conditional statements in Python allow you to control the flow of your program based on certain conditions. They help you make decisions and execute specific blocks of code selectively, depending on whether a condition is true or false. The main conditional statements in Python are:

if statement:
if-else
if-elif-else

Q21. What is use of 'if', 'elif' and 'else' keywords?

Sol)

'if' keyword:

- The 'if' keyword is used to specify a condition that is evaluated as either true or false.
- If the condition is true, the code block following the 'if' statement is executed.
- If the condition is false, the code block is skipped, and the program continues to the next statement after the 'if' block.

'elif' keyword:

- The 'elif' keyword (short for "else if") is used to specify an alternative condition to be checked if the previous 'if' or 'elif' conditions are false.
- 'elif' allows you to test multiple conditions one by one until a condition is found to be true or until all conditions have been evaluated.
- If an 'elif' condition is true, the corresponding code block is executed, and the program skips the remaining conditions and blocks.

'else' keyword:

- The 'else' keyword is used as the final part of a conditional statement.
- It specifies a block of code to be executed if all the preceding conditions ('if' and 'elif') are false.
- The 'else' block is optional, but it provides a fallback option for cases when none of the preceding conditions are true.
- If none of the conditions preceding the 'else' statement are true, the code block following the 'else' statement is executed.

Q22. Write a code to take the age of person as an input and if age >= 18 display "I can vote". If age is < 18 display "I can't vote".

Sol)

```
age = int(input("Enter your age: "))  # Take input from the user and
convert it to an integer

if age >= 18:
    print("I can vote.")
else:
    print("I can't vote.")
```

Q23. Write a code that displays the sum of all the even numbers from the given list.

~~~
```
numbers = [12, 75, 150, 180, 145, 525, 50]
```
numbers = [12, 75, 150, 180, 145, 525, 50]

sum_even = 0



for number in numbers:

    if number % 2 == 0:

        sum_even += number



print("Sum of even numbers:", sum_even)
~~~

Sol)

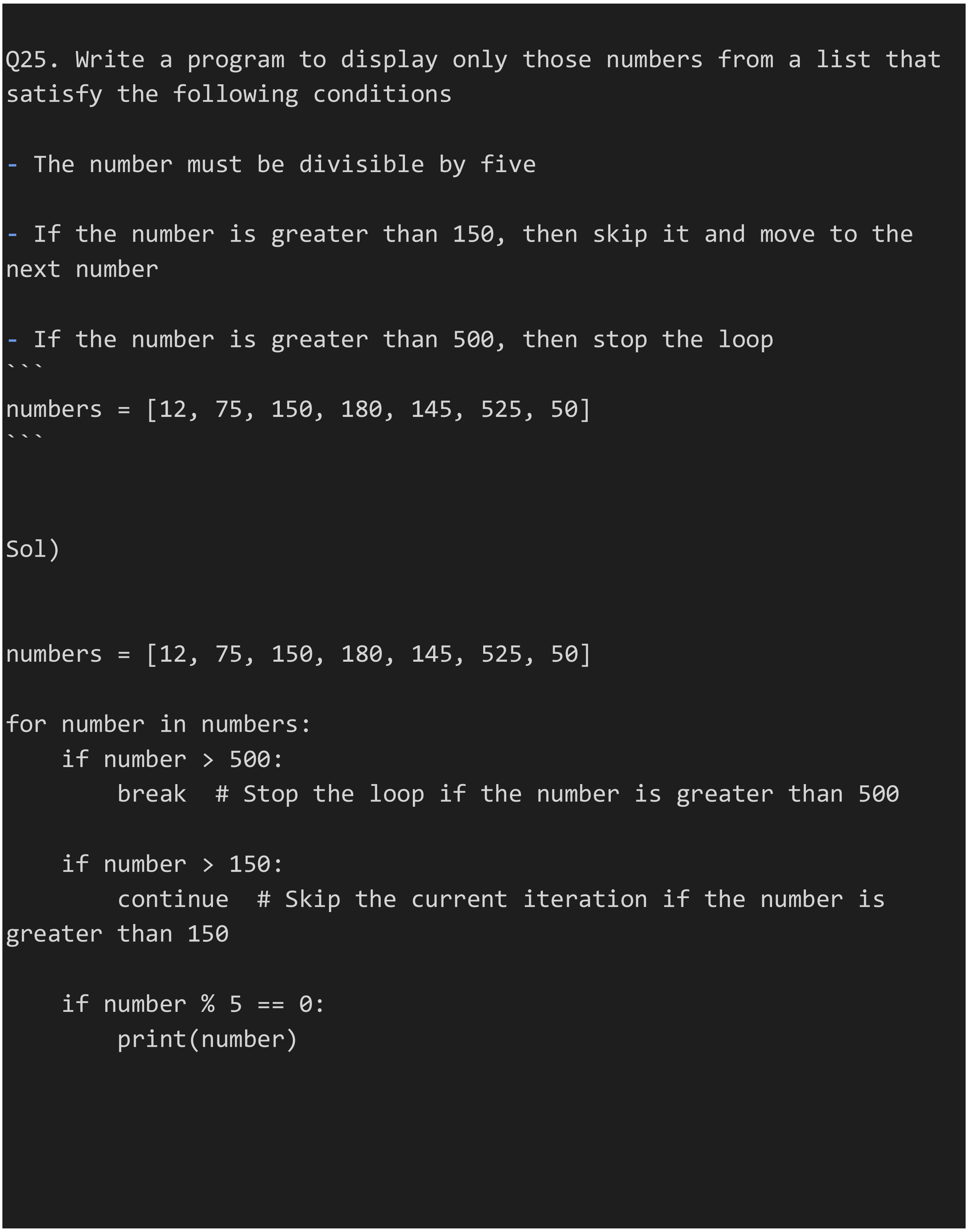
Q24. Write a code to take 3 numbers as an input from the user and display the greatest no as output.

Sol)

```
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))

if num1 >= num2 and num1 >= num3:
    greatest = num1
elif num2 >= num1 and num2 >= num3:
    greatest = num2
else:
    greatest = num3

print("The greatest number is:", greatest)
```



Q25. Write a program to display only those numbers from a list that satisfy the following conditions

- The number must be divisible by five

- If the number is greater than 150, then skip it and move to the next number

- If the number is greater than 500, then stop the loop

```
numbers = [12, 75, 150, 180, 145, 525, 50]
```

Sol)

```
numbers = [12, 75, 150, 180, 145, 525, 50]

for number in numbers:
    if number > 500:
        break  # Stop the loop if the number is greater than 500

    if number > 150:
        continue  # Skip the current iteration if the number is greater than 150

    if number % 5 == 0:
        print(number)
```