**Assignment Part-1**

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

Python is called a "general purpose" language because it can be used to build almost any type of software, including web applications, desktop apps, network servers, machine learning models, etc. it has a wide range of libraries and frameworks that cater to various domains.

Python us called a "High-level" language because it abstracts aways the low-level details of computer system, such as memory management and provides a higher level of abstraction making it easier to write, read and maintain code.

**Q2. Why is Python called a dynamically typed language?**

Python is called “Dynamically Typed” language because the data type of a variable is determined at runtime, rather than being explicitly declared at the time of creation as in statically typed languages like java or C++. This means that you can reassign a variable to a different data type at any time during the execution of the program, without having to declare it beforehand. This flexibility can make development faster but also increases the risk of unintended type errors.

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

**Pros of Python:**

1. Easy to learn
2. Versatile
3. Large community
4. Rich library Support
5. High level language

**cons of Python:**

1. Performance
2. Dynamic Typing
3. Weak in mobile
4. Memory Management
5. Global Interpreter

**Q4. In what all domains can we use Python?**

Python is a versatile Programming language that can be used in many domains, including:

1. Web Development
2. Data science and Machine learning
3. Scientific Computing
4. Artificial intelligence
5. Finance Python
6. Game Development
7. Networking
8. Internet of Things
9. Education

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

A variable in Python is a named storage location that holds a value that can be changed or updated during the execution of a program. Variable are used to store data that can be used throughout a program , making it easier to manipulate and manage data .

To declare a variable in python, simply a value to it:

**i.e., # Declare a variable**

**X = 10**

**# Reassign a new value top the variable**

**X =20**

**# Declare a variable with an integer Value**

**X = 10**

**# Change the value of X to a string**

**X = “Hello World “**

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

In python, there are several ways to take input from the user:

Using the **‘Input’** function: The **‘input’** function is used to read from the user and return the input as a string:

**# Taking input from the user**

name = input (“Enter your name: “)

print (“Hello, “ + name + “!”)

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

The default data type of the value that has been taken as an input using the **‘input()’** function in Python is a string .

For example –

**Name = input (“Enter your name: “)**

**Print(type(name))**

In the above code , **‘name’** is a string that holds the value entered by the user . The **‘type ()’**

Function is used to check the data type of the **‘name’** variable, which will be **‘str’**

**Q8. What is type casting?**

Type casting is the process of converting a value from one data type to another in programming, values can be stored in different data type , such as integers, floats , strings etc.

* **‘int()’** – Converts a value to an integer
* **‘float()’** – Converts a value to a floating-point number
* **‘str()’** – Converts a value to a string
* **‘bool()**’ – Converts a value to a Boolean

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

No, we cannot take more than one input from the user using a single input () function. The input () function reads a single line of text from the user and returns it as a string If we want to take multiple inputs from the user in a single line, we can separate the inputs using a delimiter such as a comma or space and then split the string into separate values using the split () method.

**For example:**

**# Taking multiple inputs in a single line**

**Name, age = input(“Enter your name and age: “).split()**

**print(“Your name is:”, name)**

**print(“Your age is:”, age)**

In above code , the ‘**input()’** function is used to read a string from the user , which is then spilt into separate values using the **‘split()’** method . The resulting values are then stored in the **‘name’** and **‘age’** variables.

**Q10. What are keywords?**

In programming, keywords are reserved words that have a specific meaning and cannot be used as variable names, function names, or any other identifiers. They are used to define the syntax and structure of the language

In Python, there are 35 keywords that are used for various purposes such as controlling the flow of a program, defining functions, declaring variables, and more. Some of the most commonly used Python keywords are **if**, **else**, **for**, **while**, **def**, **return**, **import**, **in**, **not**, **and**, **or**, etc

Using keywords as identifiers will result in a syntax error, and it is recommended to choose meaningful names for variables, functions, and other identifiers to make the code more readable and maintainable.

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

No, we cannot use keywords as a variable in Python. Keywords are reserved words in Python that have a specific meaning and cannot be used as identifiers, including variable names. Using a keyword as a variable name will result in a syntax error

It is recommended to choose meaningful and descriptive names for variables, functions, and other identifiers to make the code more readable and maintainable. By using descriptive names, it becomes easier for others to understand the code and make modifications if needed

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

Indentation refers to the practice of starting a new line at a certain horizontal distance from the previous line. In Python, indentation is used to delimit blocks of code and indicate the structure of the program

The use of indentation in Python is to indicate the scope of code blocks, such as loop bodies, function definitions, class definitions, and conditional statements. A block of code is considered to belong to a certain scope if it is indented under a control statement or a function /class definition.

For example, in a **for loop**, the body of the loop is indented to indicate that it belongs to the loop. In a function definition, the function body is indented to indicate that it belongs to the function .

**Q13. How can we throw some output in Python?**

In Python, you can display output using the ‘Print()’ function. The ‘print()’ function takes one or more expression as input, evaluates them, and displays the result on the console .

for example – **name = “Rahul”**

**age = 30**

**print (“My name is %s and I am &d years old.” %(name, age))**

This code will display the string **‘”My name is John and I am 30 years old.”** On the console.

**Q14. What are operators in Python?**

Operators are symbols or special characters in Python that perform operations on variables and values. Python supports various types of operators, including:

* **Arithmetic operators**: These operators perform arithmetic operations like addition, subtraction, multiplication, division, modulo, etc. For example, +, -, \*, /, %, etc.
* **Comparison operators**: These operators compare two values and return a Boolean value (True or False) based on the result. For example, ==, !=, >, <, >=, <=, etc.
* Logical operators: These operators perform logical operations like and, or, and not. They are used to combine comparison operations and evaluate complex Expressions
* **Assignment operators**: These operators are used to assign values to variables. For example, =, +=, -=, \*=, /=, etc.
* **Bitwise operators**: These operators perform operations on binary representations of integers. For example, &, |, ^, ~, &#8203;oaicite:{"index":0,"invalid\_reason":"Malformed citation <<, >>"}&#8203;, etc.
* **Membership operators:** These operators test if a value is a member of a sequence (list, tuple, etc.). For example, in and not in it.
* **Identity operators:** These operators test if two values have the same identity. For example, is and is not

In summary, operators in Python are symbols or characters that perform operations on variables and values. Python supports various types of operators, including arithmetic, comparison, logical, assignment, bitwise, membership, and identity operators.

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

In Python, the **/** operator performs float division, meaning that the result of the division will be a floating-point number. For example, **10 / 3** would give 3.3333...

On the other hand, the **//** operator performs integer division, meaning that the result of the division will be rounded down to the nearest integer. For example, **10 // 3** would give 3.

So, in summary, **/** operator is used for float division and **//** operator is used for integer division.

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

**```**

**iNeuroniNeuroniNeuroniNeuron**

**```**

**Ans** – Here is one way to achieve this o/p in python

**Print(“iNeuron” \*4 + “iNeuron”)**

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

**Answer** - Here is the code to take input from the user and check if the number is odd or even:

**# Taking input from user**

**num = int(input(“Enter a number: “))**

**#Checking if the number is odd or even**

**If num % 2 == 0:**

**print(num, “is an even number”)**

**else:**

**print(num, “is an odd number”)**

**Q18. What are Boolean operator?**

**Answer :** Boolean operators are logical operators that return either True or False based on the evaluation of a condition. There are three Boolean operators in Python:

1. . and: Returns True if both the conditions being compared are True.
2. or: Returns True if at least one of the conditions being compared is True.
3. not: Returns True if the condition being compared is False and vice versa.

These operators are used to create more complex conditions for controlling the flow of a program.

**Q19. What will the output of the following?**

```

1 or 0

0 and 0

True and False and True

1 or 0 or 0

```

**Answer -** The outputs of the given statements are:

1. 1(Since **1** is considered as True in boolean context, it is the first True value, so **1 or 0** returns **1**)
2. 0 (Since **0** is considered as False in boolean context, **0 and 0** returns **0**)
3. False (Since all the conditions are False, **True and False and True** returns False)
4. 1 (Since **1** is the first True value, **1 or 0 or 0** returns **1)**

**Q20. What are conditional statements in Python?**

Conditional statements in python are used to perform different actions based on certain conditions. The most common conditional statements are **if-elif** statements. These statements allow you to check multiple conditions and execute different parts of code based on whether the conditions are met or not. The if statement checks if a condition is true and executes a block or code if it is.

The **elif** statement is used to check additional conditions if the previous if conditions are not met. The else statement is used to specify what should be done if all conditions are not met. These conditional statements are help in creating dynamic and interactive programs that respond to user input and other events.

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

**Answer -**

**if** statement is used to evaluate a condition and execute the code block inside it if the condition is **True**.

**elif** stands for "else if". It is used in a nested **if** statement to test multiple conditions. The **elif** block is executed if the previous conditions in the **if** statement are **False** and the **elif** condition is **True**.

**else** statement is used in a nested **if** statement to specify a code block to be executed if all the conditions in the **if** statement are **False**.

**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".**

**Answer** :

Here the code to do that:

**Age = int (input(“Enter your age: “))**

**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]

```

**Answer:**

Numbers = [12, 75, 150, 180, 145, 525, 50]

Sum = 0

For num in numbers:

If num % 2 == 0:

Sum += num

print(“The sum of all the even numbers is:”, sum)

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

**Answer:**

Here is a code that takes 3 numbers as input from the user and displays the greatest numbers as output:

**num1 = int(input(“Enter first number: “))**

**num1 = int(input(“Enter second number: “))**

**num1 = int(input(“Enter third number: “))**

**if num >= num2 and num1 >= num3:**

**greatest = num1**

**elif num2 >= num1 and num2 >= num:**

**greatest = num2**

**else:**

**greatest = num3**

**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]

```

**Answer –**

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

for num in numbers:

**if num > 500:**

**break**

**if num > 150:**

**continue**

**if num % 5 == 0:**

**print(num)**