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

Ans: The Python language is called general purpose language because it can be used in the different domains such as Artificial Intelligence, Machine Learning, Data Science, Data Analysis, Web Development, Automation, etc. In the other words it is not specified for a special domain. That's why it is called general purpose programming language.

Secondly, Python is a high-level programing language because of its readability. Its syntax is readable to the humans. Aa interpreter interprets the code in the machine language (Low level language).

Q2. Why is Python called a dynamically typed language?

Ans: The Python is called the Dynamic Language because here the user does not have to declare the type of a variable like other languages where there is a strict declaration of variables before assigning the value. In the simple words we can say that variables in python are not bound to types, instead, values have types. Furthermore, we can change the values on runtime.

Example:

```
x=6
x="Apple"
```

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

Ans: Some pros and cons of the Python Language are as follows:

Advantages	Disadvantages
Vast Libraries Support	Slow Execution
Beginner-Friendly	Weak in Mobile Computing
High-Level Language	Runtime Errors
Dynamically Typed	Not Memory Efficient

Q4. In what all domains can we use Python?

Ans: There are a lot of domains where we can use the Python such as Data Science, Data Analysis, Machine Learning, Artificial Intelligence, Wed development, Automation etc.

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

Ans: The variable is a named memory location. We use them to store the values. The python has no command to declare a variable. A variable is declared when a values assigned to it. The value assigned to it determines the data type of that variable.

So, declaring a variable in python in two steps

1 Name the variable

2 Assign a value

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

Ans: We can take input from the user in python just as follows:

variable name = data type (input ("any specific instruction if necessary"))

Example: a= int (input("Please enter a value"))

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

Ans: The default datatype of the value that has been taken as an input using input () function in python is "String".

Q8. What is type casting?

Ans: The type casting is a way to change the data type of a variable into another type to do a specific operation. There are two types of type casting.

1: Implicit Type Casting

In the implicit type casting, python converts data type into another data type automatically. The user does not have to involve in the process.

Example:

```
# Python automatically converts a to int
b=2.2
print(type(b))

# Python automatically converts b to float
b = 3.0
print(type(b))
```

```
<class 'float'>
<class 'float'>
```

2: Explicit Type Casting

In the explicit type casting, a user does a process to change a data type of a variable into another specific data type. Mainly the type casting done with int(), float(0), and str() data type functions.

Int() it takes float() or str() as an argument and return int type object.

Float() it takes int() or str() as an argument and return float type object.

Str() it takes int() or float() as an argument and return string type object.

Example:

```
#integer type variable
a=2
# now type cast to float
f=float(a)
print(type(f))
print(f)

#now typecast folat to integer
c=2.0
i=int(c)
print(type(i))
print(i)

#now typecast int to str
o=5
t=str(o)
print(type(t))
print(t)

#now typecast string to int
v="123445"
k=int(v)
print(type(k))
print(k)
```

```
<class 'float'>
2.0
<class 'int'>
2
<class 'str'>
5
<class 'int'>
123445
```

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

Ans: Yes, we can take more than one inputs from the user using single input function. To do so, there are two different methos which are follows.

1: Split() Method: With this method we can take more than one inputs from the. This method breaks the inputs based on provided separator. If the separator is not given, then white space will be used to separate the inputs. This method is usually used to split the strings, but we can use it to take multiple inputs. The syntax of split() methos is as follows:

Input().split(separator, maxsplit)

Example: Without Separator

```
#taking two inputs using split method without using separator
a, b = input('Enter two integer numbers: ').split()
print("a = ",a, ",b = ",b)

#taking three inputs using split method without using separator
a, b, c = input('Enter two integer numbers: ').split()
print("a = ",a, ",b = ",b, "c = ",c)

#taking multiple inputs using split method without using separator
x=list(map(int, input("Enter multiple numbers! ").split()))
print("The values are ",x)
```

```
Enter two integer numbers: 22 33
a =  22 ,b =  33
Enter two integer numbers: 11 55 77
a =  11 ,b =  55 c =  77
Enter multiple numbers! 333 55 6 764 356 83 2 3 4 5 6 7
The values are  [333, 55, 6, 764, 356, 83, 2, 3, 4, 5, 6, 7]
```

Example: With Separator

```
#taking two inputs using split method without using separator
a, b = input('Enter two integer numbers: ').split(",")
print("a=",a, ",b=",b)

#taking three inputs using split method without using separator
a, b, c = input('Enter two integer numbers: ').split(",")

#taking multiple inputs using split method without using separator
x=list(map(int, input("Enter multiple numbers! ").split(",")))
print("The values are ",x)
```

```
Enter two integer numbers: 33,55
a= 33 ,b= 55
Enter two integer numbers: 00,67,12
Enter multiple numbers! 123,876,454,888,000,333
The values are  [123, 876, 454, 888, 0, 333]
```

2: List Comprehension: The list comprehension is a most elegant way to take more than one or multiple inputs using a single input function.

```
# Taking multiple inputs in a single line

x=list(map(int, input("Enter the multiple numbers").split()))
print("The numbers are",x)

# above i also done the type casting using list() function
```

```
Enter the multiple numbers123 456 7 8 99 334 55 66 33 3 33 3 32335 6 6
The numbers are [123, 456, 7, 8, 99, 334, 55, 66, 33, 3, 33, 3, 32335, 6, 6]
```

Same as in split() method, here, we also can use separator to break the inputs.

Q10. What are keywords?

Ans: The keywords in python are predefined and reserved words use to perform a specific operation. The keywords can not be used as identifier, function or variable name. All the keywords in python are written in lower case except True and False. Some python keywords are: if, elif, else, and, or, not, for, while, break etc.

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

Ans: No, we can not use keywords as variable because keywords are the reserved words to perform some specific operations in python. If we take a keyword as variable, it will create ambiguity and the interpreter will get confused whether to take is as variable or keywords.

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

Ans: The indentation in python is nothing but a white space (or tab) before a statement (line of code). Like in C, C++ and many other programming languages we use curly braces to create a block of code but here in python we use indentation to make our code structured, easily readable and beautiful. The statements with the same indentations belong to the same block or block of code called **suite**. In most of the IDEs, the indentation is automatically performed.

Example:

```
if a==1:
    print(a)
    if b==2:
        print(b)
print("End!")
```

In the above example the first and last lines of code belongs to the same suite because there is no indentation before them. So, after executing first if statement the interpreter will go the line two. If the condition is not true, it will execute the last line.

At the next stage the following statements are typed with the four spaces (or a tab). So, they are belonging to the same suite.

```
print(a)
if b==2:
```

The third statement “b==2” will be executed only if the first statement “a==1” is true.

In the next stage (below), there are eight typed spaces (or two tabs). So, it belongs to a separate suite. And it will be executed only if the thirds statement “b==2” will be true.

```
print(b)
```

Q13. How can we throw some output in Python?

Ans: To get some output in python we use the print() function.

Example: print(“iNeuron”)

Q14. What are operators in Python?

Ans: The operators are used to perform different operations on variables and values. There is different type of operators in python.

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Identity operators
- Membership operators
- Bitwise operators

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

Ans: / and // both are the arithmetic operators. But the difference is that / is used for division while // is used for the floor division. To elaborate it, when we use / operator the result comes in float value but if we want to get the result in single value of without floating point value then we use the floor division.

Example:

```
a=12
b=13
print("The result of / is ",a/b)
print("The result of // is ",a//b)
```

```
The result of / is  0.9230769230769231
The result of // is  0
```

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

iNeuroniNeuroniNeuroniNeuron

Ans:

```
print("iNeuroniNeuroniNeuroniNeuron")
```

```
iNeuroniNeuroniNeuroniNeuron
```

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

Ans:

```
a=int(input("Please enter a number! "))
if a%2==0:
    print (a,"Is a Even Number")
else:
    print (a,"Is a Odd Number")
```

```
Please enter a number! 67
67 Is a Odd Number
```

Q18. What are Boolean operator?

Ans: The logical operators in python are called Boolean operators. It includes and, or, & not operators. They give the result in True or False. Here, and & or require two operands to provide result in True or False while not operator require only one operand to provide the result.

Example: And Operator

```
#using 'and' operator to get the result
a=100
b=90
a>100 and b>90
```

```
False
```

Example: Or Operator

```
a=80
b=50
a>70 or b<50
```

```
True
```

Example: Not Operator

```
a=1000
b=2000
not (a>b)
```

```
True
```

Q19. What will the output of the following?

```

```

(1 or 0)

0 and 0

True and False and True

1 or 0 or 0

```
```

Ans:

```
print(1 or 0)

print(0 and 0)

print(True and False and True)

print(1 or 0 or 0)
```

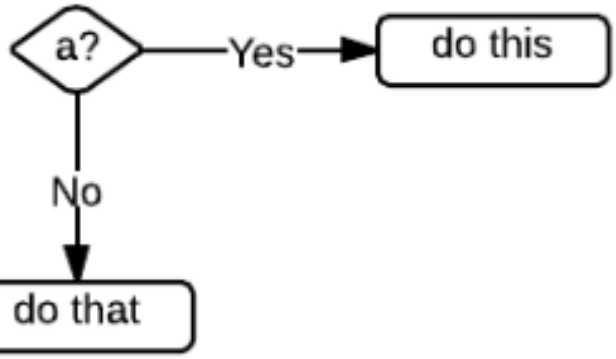
```
1
0
False
1
```

Q20. What are conditional statements in Python?

Ans: The conditional statements in python are used to handle the conditions. The conditions are evaluated as true or false. Like if a specific condition is true then the program will run as needed but if the given condition becomes false then the statement which is following the “if condition” will be executed.

The basic structure of the conditional statement is as follows:

```
if a:
    do this
else:
    do that
```



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

Ans: These are all conditional statements in python. These are used when we write a program where we must handle the conditions.

If Condition: This is the most used conditional statement in python. It decides whether the certain statements need to be executed or not. If the given condition is true, then the code inside the if block will be executed otherwise not.

Example:

```
a=20
b=30
if b>a:
    print("The higher value is ",b)
else:
    Print("b is not a higher value!")
```

```
The higher value is  30
```

Elif Condition: This condition is used when the if condition is not true, but we want to check some more conditions.

Example:

```
a=int(input("Enter the marks of student "))
if a>=90:
    print("The grade is A")
elif a>=70:
    print("The grade is B")
elif a>=50:
    print("The grade is C")
elif a<50:
    print("You are fail")
else:
    print("Invalid command")
```

```
Enter the marks of student 33
You are fail
```

Else Condition: The else condition is executed when other condition like if and elif gets False. This condition is always comes in the last.

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".

Ans:

```
 #if age >= 18 display "I can vote". If age is < 18 display "I can't vote".
a=int(input("Please enter the age! "))
if a>=18:
    print("I can vote ")
if a<18:
    print("I can not vote")
```

```
Please enter the age! 12
I can not 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]**

```

Ans:

```
numbers = [12, 75, 150, 180, 145, 525, 50]
print("The even numbers from the list are as follows:")
for a in numbers:
    if a%2==0:
        print(a)
```

```
The even numbers from the list are as follows:
12
150
180
50
```

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

Ans:

```
a=int(input("Please enter the first number! "))
b=int(input("Please enter the second number! "))
c=int(input("Please enter the third number! "))
if a>b and a>c:
    print("The greater number is ",a )
elif b>a and b>c:
    print("The greater number is ",b)
else:
    print("The greate numbe is ",c)
```

```
Please enter the first number! 777
Please enter the second number! 3455
Please enter the third number! 222445
The greate numbe is  222445
```

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]**

```
numbers = [12, 75, 150, 180, 145, 525, 50]
print("The numbers divisible by five are as follows!")
for a in numbers:
    if a%5==0:
        print(a)

    elif a>500:
         break;
```

```
The numbers divisible by five are as follows!
75
150
180
145
525
50
```
```