Q1. What is Python, and why is it popular?
ANS. Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together.
Python's simplicity, versatility, extensive libraries, and strong community support make it a favorite for everything from web development to cutting-edge AI applications. It's this adaptability and ease of use that drive its immense popularity.

Q2. What is an interpreter in Python?
ANS. An interpreter in Python is a program that reads and executes Python code line by line. It converts the high-level Python code into a format that the computer can understand and execute immediatel i.e. machine code.

Q3. What are pre-defined keywords in Python?
ANS. Pre-defined keywords in Python are reserved words that have special meaning to the interpreter. They are part of the Python language syntax and cannot be used as identifiers (such as variable names, function names, or class names). These keywords help define the structure and flow of a Python program.
Example.False, None, True, and, as, assert, async, await, break, class, continue, def, del, elif, else, except, finally, for, from, global,
if, import, in, is, lambda, nonlocal, not, or, pass, raise, return,try, while, with, yield

Q4. Can keywords be used as variable names?
ANS. No, keywords cannot be used as variable names in Python. Keywords are reserved by the Python interpreter for specific purposes, so using them as variable names will result in a syntax error.

Q5. What is mutability in Python?
ANS. Mutability in Python refers to whether an object's value can be changed (mutated) after it is created. Based on mutability, Python objects are categorized as either mutable or immutable.
Example: Mutable - List, Dictiomary, Sets.
Immutable - Integer, String, Tuples.

Q6. Why are lists mutable, but tuples are immutable?
ANS. Lists are mutable as they are designed for dynamic changes like adding, removing, or modifying elements. Tuples, on the other hand, are b immutable, ensuring stability, performance optimization, and hashability, making them ideal for use as dictionary keys or set elements.

Q7. What is the difference between "==" and "is" operators in Python?
ANS == : Compares the values of two objects for equality.
is : Checks whether two objects refer to the same memory location.

Q8. What are logical operators in Python?
ANS In python logical operators are 'and', 'or' and 'not' which are used to combine two or more Boolean expressions and return a Boolean value (True or False) depending on the result of the combination.

Q9. What is type casting in Python?

ANS. Type Casting is the method to convert the variable datatype into a certain data type in order to perform the required operation.

Q10. What is the difference between implicit and explicit type casting?
ANS. Implicit type casting -> Automatically converts a smaller primitive type to a larger primitive type. It’s convenient for compatible conversions and can reduce errors, but can also lead to data loss and unexpected results.
Explicit type casting -> Manually converts a larger primitive type to a smaller primitive type. It offers more control and clarity.

Q11. What is the purpose of conditional statements in Python?
ANS. In Python, conditional statements are used to make decisions in our code. They allow our program to execute specific blocks of code based on n whether a condition evaluates to True or False.

Q12. How does the elif statement work?
ANS. The elif statement in Python is short for "else if" and is used to check multiple conditions in a sequence. It allows your program to execute a specific block of code if its associated condition evaluates to True, but only if the previous conditions were False.

Q13. What is the difference between for and while loops?
ANS. Both for loop and while loop is used to execute the statements repeatedly while the program runs. The major difference between for loop and while loop is that for loop is used when the number of iterations is known, whereas execution is done in a while loop until the statement in the program is proved wrong.

Q14. Describe a scenario where a while loop is more suitable than a for loop.
ANS. When the number of iterations is not known beforehand and depends on a condition.
For example ->> 1. reading input until a specific value is entered:
2. Running a game loop until the player decides to quit or a win/loss condition is met.

```
#1. Write a Python program to print "Hello, World!

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

```
Hello, World!
```

```
#2 Write a Python program that displays your name and age

name='Arpit Yadav'
age = 21
print(name)
print(age)
```

```
Arpit Yadav
21
```

```
#3 Write code to print all the pre-defined keywords in Python using
the keyword library.

import keyword
print("List of all keywords is: ")
print()
print(keyword.kwlist)
```

```
List of all keywords is:

['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await',
'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except',
'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is',
'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'return', 'try',
'while', 'with', 'yield']
```

```
#4 Write a program that checks if a given word is a Python keyword

import keyword
word = input("Enter a word: ")
if keyword.iskeyword(word):
    print(f"'{word}' is a Python keyword.")
else:
    print(f"'{word}' is not a Python keyword.")
```

```
Enter a word:  arpit

'arpit' is not a Python keyword.
```

```
#5 Create a list and tuple in Python, and demonstrate how attempting
to change an element works differently for each.

list_1 = [1,2,3]
tupple_1 = (1,2,3)
list_1[0] = 21
print("Modified List is: ",list_1)
```

```
Modified List is:  [21, 2, 3]

tupple_1[0] = 21
print("Modified tupple is: ",list_1)

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[35], line 1
----> 1 tupple_1[0] = 21
      2 print("Modified tupple is: ",list_1)

TypeError: 'tuple' object does not support item assignment

# tuple can not be modified....it will show error(TypeError)

#6 Write a function to demonstrate the behavior of mutable and
immutable arguments

def demonstrate():
    # Immutable example
    x = 5
    def modify_immutable(n):
        n += 1
    modify_immutable(x)
    print("Immutable after function:", x)

    # Mutable example
    lst = [1, 2, 3]
    def modify_mutable(l):
        l.append(4)
    modify_mutable(lst)
    print("Mutable after function:", lst)

demonstrate()

Immutable after function: 5
Mutable after function: [1, 2, 3, 4]

#7 Write a function to demonstrate the behavior of mutable and
immutable arguments

def demonstrate():
    # Immutable example
    x = 5
    def modify_immutable(n):
        n += 1
    modify_immutable(x)
    print("Immutable after function:", x)
    # Mutable example
    lst = [1, 2, 3]
    def modify_mutable(l):
        l.append(4)
    modify_mutable(lst)
    print("Mutable after function:", lst)

demonstrate()
```

```
Immutable after function: 5
Mutable after function: [1, 2, 3, 4]
```

```
#8 Write a program to demonstrate the use of logical operators.

a = 10
b = 20

# Using 'and', 'or', and 'not'
print(a < b and b > 15)  # True and True => True
print(a > b or b == 20)  # False or True => True
print(not (a > b))       # not(False) => True
```

```
True
True
True
```

```
#9. Write a Python program to convert user input from string to
integer, float, and boolean types

num = input("Enter a number: ")
type(num)
```

```
Enter a number:  4
```

```
str
```

```
#int
print(type(int(num)))
#float
print(type(float(num)))
#boolean
print(type(bool(num)))
```

```
<class 'int'>
<class 'float'>
<class 'bool'>
```

```
#10. Write code to demonstrate type casting with list elements

# Demonstrate type casting with list elements
lst = ["1", "2", "3"]
print("Original:", lst)
print("To int:", [int(x) for x in lst])
print("Back to str:", [str(x) for x in lst])
```

```
Original: ['1', '2', '3']
To int: [1, 2, 3]
Back to str: ['1', '2', '3']
```

```
#11. Write a program that checks if a number is positive, negative, or
zero

# Program to check if a number is positive, negative, or zero
num = int(input("Enter Your Number: "))
if num < 0:
    print(f"{num} is negative")
elif num > 0:
    print(f"{num} is positive")
else:
    print(f"{num} is zero")
```

```
Enter Your Number:  0

0 is zero
```

```
#12  Write a for loop to print numbers from 1 to 100
for i in range(1,101):
    print(i,end=" ")
```

```
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49
50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95
96 97 98 99 100
```

```
#13 Write a Python program to find the sum of all even numbers between
1 and 50

for i in range(1,51):
    if i%2==0:
        print(i,end=" ")
```

```
2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50
```

```
#14 Write a program to reverse a string using a while loop
str_1 = "String"
length = len(str_1)-1
while length>=0:
    print(str_1[length],end=" ")
    length-=1
```

```
g n i r t S
```

```python
#15 Write a Python program to calculate the factorial of a number
provided by the user using a while loop.
num = int(input("Enter your number: "))
factorial = 1
i = 1
while i <= num:
    factorial *= i
    i += 1
print("Factorial of", num, "is", factorial)
```

```
Enter your number:  6

Factorial of 6 is 720
```