

**Student Learning Outcomes**

By the end of this chapter, you will be able to:

- Implement control structures such as decision-making statements and loops in Python.
- Work with Python modules, functions, and built-in data structures like lists.
- Apply modular programming techniques and object-oriented programming concepts in Python.
- Handle exceptions, perform file operations, and apply testing and debugging techniques in Python.

**Subjective Type Questions**

Long Questions prepared in the light of new Examination Techniques  
(Knowledge, Understanding, Application, Analytical & Conceptual)

**Introduction**

**Q. Describe the importance of control structure.**

**Ans:** In programming, we often need to control the flow of our program based on different conditions or repeat certain actions multiple times. This is where control structures come into play.

**Importance of Control structures;**

Control structures are essential because they help us manage the decision-making process and the repetition of tasks in our programs. There are two main types of control structures, Decision making and Looping:

**4.1****Decision Making**

**Q.1. Define decision making? Name the type of decision statements used in python?**

**Ans:**

Decision making in programming allows the program to choose different actions based on conditions. This is similar to how we make decisions in real life.

**Example:** If you try to quit an application (mobile app), by pressing back button application ask;

'Do you want to quit': Yes or No

This shows the decision-making structure followed by the software.

**Condition:**

Condition is an expression contain operands with relational operators or logical



operators or a combination of both which give results in boolean form, either it is true or either false. All decision making statements based on condition.

### Types of Decision statements:

- if statement
- if else statement

**Q.2. Define if statement with its syntax? Describe the limitation of if statement?**

**Ans:**

The if statement make decisions based on conditions. If the condition is true, it runs a block of code.

**Syntax:**

```
if condition:
    code to run if the condition is true
```

**Indentation:**

In Python indentation defines the structure block or scope of your code. In Python proper indentation is mandatory. Incorrect indentation may cause error in your code (Indentation Error).

**Example - 1:**

```
Age = 21
if Age > 18:
    print("You are a teenager!!!!!!")
```

**Example - 2:**

```
temperature = 20
if temperature > 25:
    print("It's a hot day")
```

**Limitation of if statement:**

if statement shows output if the certain condition becomes true if the condition becomes false nothing will displayed on the screen.

**Q.3. Define if - else statement with its syntax? Write an example.**

**Ans:**

The if-else statement allows us to execute one block of code if a condition is true and another block if the condition is false.

**Syntax of if-else statement:**

```
if condition:
    code to run if the condition is true
else:
    code to run if the condition is false
```

**Example:**

```
temperature = 15
if temperature > 30:
    print("It's a hot day")
```



else:

```
print("It's not a hot day")
```

**Output:** It's not a hot day

**Q.4. Define Nested condition with its syntax? Write an example.**

**Ans:**

Sometimes, we need to check multiple conditions inside another condition. This is called nesting.

So, if statement with the body of another if statement is called nested if. This structure is used to apply complex multiple conditions.

**Syntax of nested if statement:**

if condition1:

    if condition2:

        code to run if both condition1 and condition2 are true

    else:

        code to run if condition1 is true but condition2 is false

else:

    code to run if condition1 is false

**Example:** If the weather is rainy and the temperature is below 15 degrees, we wear a raincoat. If it is only rainy, we take an umbrella. If the weather is not rainy, we just enjoy the day.

**Code:**

```
weather = "rainy"
```

```
temperature = 10
```

```
if weather == "rainy":
```

```
    if temperature < 15:
```

```
        print("Wear a raincoat")
```

```
    else:
```

```
        print("Take an umbrella")
```

```
else:
```

```
    print("Enjoy your day!")
```

**Output:**

Wear a raincoat

**Explanation:**

In this example, the code checks if the weather is rainy. If it is, it then checks if the temperature is below 15 degrees. If both conditions are true, it prints "Wear a raincoat". If the weather is rainy but the temperature is not below 15 degrees, it prints "Take an umbrella". If the weather is not rainy, it prints "Enjoy your day!".

## 4.2

## Looping Constructs

**Q.5. Define Loop? Explain while loop with syntax and example.**

**Ans:**

If we want to repeat a statement or a set of statements for more than one times, we



use loops. Loop help us to generate maximum output with the help of minimum input. In python we use two type of loops:

- While loop
- For loop

### While loop:

A while loop runs as long as a condition is true. It checks the condition before each iteration and stops running when the condition is no longer true. Basically, while loop is used where we don't know in advance how many time our loop is repeats.

### Syntax of while loop:

while condition:

code to run while the condition is true

**Example:** Add 1 to a number until it reaches 10.

```
number = 1
```

```
while number < 10:
```

```
    print(number)
```

```
    number += 1
```

**Explanation:** In this example, the code starts with a number set to 1. The 'while' loop checks if the number is less than 10. If it is, the program prints the current value of the number and then adds 1 to it. This loop continues until the number reaches 10, at which point the loop stops running.

### Q.6. Define and explain for loop with syntax and example.

**Ans:**

A for loop repeats a block of code a specific number of times. It is commonly used to iterate over a sequence (like a list, tuple, or string).

### Syntax of for loop:

for variable in sequence:

code to run for each element in the sequence

**Example:** Say "As-Salaam-Alaikum" to each friend in a list of friends.

```
friends = ["Sami", "Raza", "Moosa"]
```

```
for friend in friends:
```

```
    print("Welcome to ", friend)
```

### Output:

```
Welcome to: Sami
```

```
Welcome to: Raza
```

```
Welcome to: Moosa
```

**Explanation:** In this example, the code goes through each friend in the list and prints a greeting message for each one.

### Q.7. Define range function and end statement. Also write examples

### Ans: Range Function:

We can use the range() function to generate a sequence of numbers, which is often used in for loops.



### Syntax of range function:

`range(start)`

`range(start, stop)`

`range(start, stop, step)`

**Example:** Print First 5 Whole Numbers.

```
for i in range(5):  
    print(i)
```

### Output:

0  
1  
2  
3  
4

### End Statement:

In Python, the end statement is not a separate command, but rather an optional parameter of the `print()` function. By default, every `print()` in Python ends with a newline character, but you can change what gets printed at the end using the `end` parameter.

**Example 1:** Print two Strings on same line using "end" statement.

### Code:

```
print("Azhar Shah", end=" ")  
print("Ahsan")
```

### Output:

Azhar Shah Ahsan

### Code:

```
print("Ayyan", end="***")  
print("Naveed")
```

### Output:

Ayyan\*\*\*Naveed

**Example 2:** Print Numbers from 1 to 5 on the same line using "end" statement.

### Code:

```
for i in range(1, 6):  
    print(i, end=" ")
```

### Output:

1 2 3 4 5

**Explanation:** The above code generates numbers from 0 to 4 and prints each number.

### DO YOU KNOW

Strings in Python are surrounded by either Single quotation ' marks or double quotation " marks. Like 'Ayyan' is the same as "Ayyan".



## ACTIVITY

1. Write a for loop using range() to print the even numbers from 2 to 10 on single line.
2. Write a Python program that prints the first 10 multiples of 3 using a for loop and the range() function.

### 1. Solution:

#### Code:

```
for i in range(2, 11, 2):  
    print(i, end=" ")
```

#### Output:

2 4 6 8 10

### 2. Solution:

#### Code:

```
for i in range(3, 31, 3):  
    print(i, end=" ")
```

#### Output:

3 6 9 12 15 18 21 24 27 30

## Q.8. Define nested loop? Also write examples.

### Ans:

Nested loops are loops inside other loop. For each iteration of the outer loop, the inner loop runs completely. We use nested loops when we need to perform repetitive tasks within tasks.

### Example 1: Nested loops for basic Number Pattern

#### Code:

```
print("=== Basic Number Pattern ===")  
for i in range(3): # Outer loop  
    for j in range(4): # Inner loop  
        print(f'i={i}, j={j}')  
    print("---") # Separator after each outer loop
```

#### Output:

=== Basic Number Pattern ===

i=0, j=0

i=0, j=1

i=0, j=2

i=0, j=3

---

i=1, j=0

i=1, j=1

i=1, j=2

i=1, j=3

---



i=2, j=0

i=2, j=1

i=2, j=2

i=2, j=3

---

**Example 2: Nested loops to print stars (asterisk) in Triangular Pattern**

**Code:**

```
print("=== Right-Angled Triangle ===")
```

```
n = 5
```

```
for i in range(1, n+1):
```

```
    for j in range(i): # Inner loop runs 'i' times
```

```
        print("*", end=" ")
```

```
    print() # New line after each row
```

**Output:**

```
*
```

```
**
```

```
***
```

```
****
```

```
*****
```

### 4.3

### Libraries in Python

**Q.9. What is a data structure and Library in python? Apply different libraries used in python?**

**Ans: Data Structure:**

A data structure refers to a particular format or method for organizing and storing data. In python, List, Tuple are popular data structures.

**List:**

In Python, libraries are like toolboxes full of useful tools that help you solve different problems without having to build everything from scratch. Libraries are like pre-built toolkits that you can use without having to write all the code yourself.

**Example:** Import the random library to generate random numbers

**Code:**

```
import random
```

```
number = random.randint(1, 10)
```

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

**Output:**

```
The random number is: 3
```

**Example:** Import the statistics library to perform statistical calculations.



**Code:**

```
import statistics
data = [23, 45, 67, 89, 12, 44, 56]
mean_value = statistics.mean(data)
print("The mean value is:", mean_value)
```

**Output:**

The mean value is: 48

**Explanation:** The statistics library is a great tool for performing basic statistical calculations, such as finding the mean, median, and mode of a set of data. This can be particularly useful in data analysis tasks.

In short, by using these libraries, you can save time and effort, allowing you to focus on solving the specific problem at hand rather than reinventing the wheel.

#### TOOL TIP

When importing libraries, make sure to only import what you need. You can explore more libraries at <https://docs.python.org/3/>

## 4.4

## Lists in Python

**Q.10. Define List? Describe creating, accessing and modifying list with example.**

**Ans:**

In Python, a list is a versatile data structure that can hold a collection of items. You can create, access, and modify lists easily.

**Creating Lists:** A list is created by placing items inside square brackets [ ], separated by commas. Lists can contain items of different types, such as numbers, strings, or even other lists.

**Index number:** Every list element have an index number. When a list is created by default an index number is assigned to each element. This index number is used to access all elements save in that list.

**Example:** Create a list of your favorite fruits.

**Code:**

```
fruits = ["Mango", "Apple", "Banana"]
print(fruits)
```

**Output:**

['Mango', 'Apple', 'Banana']

**Explanation:** This code creates a list named 'fruits' containing three elements and then prints the list.

**Accessing List Items:**

You can access items in a list by referring to their index, starting from 0.



**Example:** Access and print the second item from the list of fruits.

**Code:**

```
fruits = ["Mango", "Apple", "Banana"]  
print(fruits[1])
```

**Output:**

Apple

**Explanation:** The code initializes a list 'fruits' containing 'Mango', 'Apple', and 'Banana', then prints the second item, 'Apple', using the index '1'.

**Modifying a List:**

You can modify list items by accessing them via their index and assigning a new value.

**Example:** Change the first item in the list to "Orange" and add a new fruit "Pineapple".

**Code:**

```
fruits = ["Mango", "Apple", "Banana"]  
fruits[0] = "Orange"  
fruits.append("Pineapple")  
print(fruits)
```

**Output:**

['Orange', 'Apple', 'Banana', 'Pineapple']

**Explanation:** The code modifies the first element of the fruits list to 'Orange'. It then appends 'Pineapple' at the end, and prints the updated list.

### ACTIVITY

Write a Python program that imports the random and statistics libraries. Use random to generate a list of 10 random numbers between 1 and 50. Then, use statistics to calculate and print the mean (average) of those numbers.

**Solution:**

```
import random  
import statistics  
# Generate a list of 10 random numbers between 1 and 50  
random_numbers = []  
for i in range(10):  
    random_numbers.append(random.randint(1, 50))  
# Print the list of random numbers  
print("Random numbers:", random_numbers)  
# Calculate and print the mean (average)  
mean_value = statistics.mean(random_numbers)  
print("Mean (average):", mean_value)
```



**Output:**

Random numbers: [23, 45, 12, 38, 7, 42, 19, 31, 8, 27]

Mean (average): 25.2

**Q.11. Explain append(), remove(), sort() and reverse() operation applied in list using python code?**

**Ans:**

Python provides several built-in methods to work with lists. Here are a few useful ones:

- **append (item):** Adds an item to the end of the list.
- **remove(item):** Removes the first occurrence of an item from the list.
- **sort(item):** Sorts the list in ascending order.
- **reverse(item):** Reverses the order of the list.

**1. append() function:**

This operation add new item at the end of the list.

**Example:** Add a new student to the list of students and print the list.

**Code:**

```
students = ["Ahmed", "Sara", "Ali"]  
students.append("Hina")  
print(students)
```

**Output:**

```
['Ahmed', 'Ali', 'Sara', 'Hina']
```

**Explanation:** The code creates a list of students, adds 'Hina' to it, sorts the list alphabetically.

**2. remove () function:**

This operation is used to remove an item from the list

**Example:** Remove a student from the list of students and print the list.

**Code:**

```
students = ["Ahmed", "Sara", "Ali", "Hina", "Sara"]  
students.remove("Sara")  
print(students)
```

**Output:**

```
['Ahmed', 'Ali', 'Hina', 'Sara']
```

**Explanation:** The code creates a list of students, removes the first occurrence of "Sara" from the list, and prints the updated list. Note that only the first matching item is removed.

**3. Sort() function:**

This operation is used to arrange items in a specific order.



**Example:** Sort a list of numbers in ascending order.

**Code:**

```
numbers = [34, 12, 89, 5, 23]
numbers.sort()
print(numbers)
```

**Output:**

```
[5, 12, 23, 34, 89]
```

**Explanation:** The code creates a list of numbers, sorts them in ascending order using the `sort()` method, and prints the sorted list.

#### 4. `reverse()` function:

This operation is used to reverse the order of items of a list

**Example 4:** Reverse the order of fruits in a list.

**Code:**

```
fruits = ["Apple", "Banana", "Orange", "Mango"]
fruits.reverse()
print(fruits)
```

**Output:**

```
['Mango', 'Orange', 'Banana', 'Apple']
```

**Explanation:** The code creates a list of fruits, reverses the order of elements using the `reverse()` method, and prints the reversed list.

### ACTIVITY :

Imagine you are maintaining a list of your favorite books: ["To Kill a Mockingbird", "1984", "The Great Gatsby", "Pride and Prejudice"]. Perform the following tasks using Python:

- Add a new book "Alchemist" to the list.
- Replace "1984" with "Brave New World".
- Remove "The Great Gatsby" from the list.
- Merge this list with another list of books: ["War and Peace", "Hamlet"].
- Print the final list of books.

Write the Python code to execute these tasks and print the final list of books.

### Solution

**Code:**

**# Starting list**

```
books = ["To Kill a Mockingbird", "1984", "The Great Gatsby", "Pride and Prejudice"]
```

**# Perform all operations**

```
books.append("Alchemist")
```

```
books[books.index("1984")] = "Brave New World"
```



```
books.remove("The Great Gatsby")
books.extend(["War and Peace", "Hamlet"])
# Print final result
print("Final list of books:", favorite_books)
```

4.5

## Testing and Debugging in Python

### Q.12. Explain testing and its types?

Ans:

Testing is the process of running your code with various inputs to check if it behaves as expected. The goal is to find and fix any issues before the code is used in real-world applications.

#### Types of Testing:

- **Unit Testing:** Tests individual parts of the code (like functions or classes) in isolation. Python's unit test module is commonly used for this.
- **Integration Testing:** Checks how different parts of the code work together.
- **Functional Testing:** Validates that the software behaves as expected from the user's perspective.
- **Regression Testing:** Ensures that new changes don't affect the existing functionality.

### Q.13. Explain Debugging and its techniques?

Ans:

Debugging is the process of finding and fixing errors (bugs) in your code. It involves identifying the root cause of problems and making the necessary changes.

#### Common Debugging Techniques:

- **Print Statements:** Adding print statements to check the values of variables at different stages of the code.
- **Debugging Tools:** Using tools like pdb (Python Debugger) to step through the code, inspect variables, and understand the flow of execution.
- **Error Messages:** Reading and interpreting error messages to locate the source of the problem.

## Objective Type Questions

### Additional MCQ's

Multiple Choice Questions prepared in the light of new Examination Techniques  
(Knowledge, Understanding, Application, Analytical & Conceptual)

#### Introduction

#### 1. Control Structure is essential because:

- (A) Help us to manage Decision Making
- (B) Help us to manage Repetition task
- (C) Help us to manage complex programs
- (D) All above



2. Repetition structure is also called:  
(A) iteration (B) loop (C) Both a, b (D) None

#### 4.1

#### Decision Making

3. How many Types of conditional statement?  
(A) 1 (B) 2 (C) 3 (D) 4
4. Decision making allow programmer to choose different action based on:  
(A) Condition (B) Expression (C) Evaluation (D) Loop
5. Output of if (5 > 7): is using if statement structure:  
(A) True (B) False (C) No output (D) Yes
6. In Python \_\_\_\_\_ define structure or block of your code:  
(A) syntax (B) operator (C) Block (D) Indentation
7. If incorrect indentation exist it may cause?  
(A) Logical Error (B) Syntax error (C) Runtime Error (D) Logical Error
8. Syntax of if statement:  
(A) if condition then: Block (B) if condition: Block  
(C) if block: condition (D) if condition do: Block
9. The else keyword is used to display:  
(A) True statement (B) False statement  
(C) Parallel to it (D) Perpendicular to it
10. Output of if temp > 30 is:  
(A) True (B) False (C) Error (D) 30
11. In Python int() is a:  
(A) Data type (B) Keyword (C) Function (D) Syntax of integer
12. if statement within in if statement:  
(A) Multiple if (B) if-elif-else (C) if-else (D) Nested if
13. Inner if block is runs only when:  
(A) Both conditions are false (B) Both conditions are true  
(C) Both conditions are equal (D) none of them
14. the result of 10 < 5 in Python:  
(A) 5 (B) 10 (C) True (D) False
15. Block written in inner else is execute only when:  
(A) Cond-1 is true But Cond-2 is False  
(B) Cond-1 is False But Cond-2 is true  
(C) Both are true (D) Both are False
16. How many conditions are represent using nested conditions?  
(A) 1 (B) 2 (C) 3 (D) Any number

#### 4.2

#### Looping Constructs

17. How many Types of loop used in Python  
(A) 1 (B) 2 (C) 3 (D) 4



18. When total number of iteration are known \_\_\_\_\_ loop is used?  
 (A) while (B) do while (C) Nested (D) For
19. output of `n=1 while n < 4: Print(n)`  
 (A) 0 1 2 (B) 1 2 3 4 (C) 1 2 3 (D) 2 3 4
20. When total number of iterations are not known, we use \_\_\_\_\_ loop  
 (A) while (B) do-while (C) Nested (D) For
21. \_\_\_\_\_ loop uses sequence or range:  
 (A) while (B) do-while (C) For (D) All
22. \_\_\_\_\_ loop repeat until condition becomes False:  
 (A) while (B) For (C) Do-while (D) All
23. Which loop uses List for iteration  
 (A) while (B) Do-while (C) for (D) Nested
24. \_\_\_\_\_ function is used to get or generate sequence of numbers?  
 (A) input function (B) output function  
 (C) range function (D) Built-in function
25. In range function step performs:  
 (A) Addition (B) Subtraction  
 (C) Multiplication (D) Depends upon input
26. output of the following code: `for i in range(3): Print(i)`  
 (A) 0 1 2 3 (B) 0 1 2 (C) 1 2 3 (D) 3 2 1 0
27. Range function uses \_\_\_\_\_ syntaxes:  
 (A) 1 (B) 2 (C) 3 (D) 4
28. String in Python are marked with:  
 (A) Single quotes (B) Double quotes (C) Triple quotes (D) Both a, b
29. In python value 'ayan' and "Ayan" are:  
 (A) Same (B) Different (C) Both (D) None
30. Loop inside another loop is called:  
 (A) while (B) Do-while (C) Nested (D) For
31. If outer loop ends in 3 iteration and inner loop has range(5) then total number of iterations are:  
 (A) 3 (B) 5 (C) 15 (D) 12

### 4.3

### Libraries in Python

32. Libraries are \_\_\_\_\_ Modules:  
 (A) Built-in (B) user-define (C) compiler define (D) Function
33. Which keyword is used to add libraries in Python?  
 (A) Matplotlib (B) Random (C) import (D) Add
34. Random is a:  
 (A) Function (B) Library (C) Modulus (D) compiler



35. Randint() is a:  
(A) Function (B) Library (C) Modulus (D) compiler
36. Statistics.mean(data) in this line of code data is a:  
(A) Function (B) Library (C) List (D) Array

#### 4.4

#### Lists in Python

37. List is a:  
(A) Array (B) Function (C) Modulus (D) Data Structure
38. List can contain:  
(A) Same items (B) Single items (C) different items (D) None of them
39. In a List, every element have:  
(A) index (B) value (C) access (D) position
40. In a List starting index number is:  
(A) 1 (B) 2 (C) 0 (D) -1
41. We can access element in a by its:  
(A) value (B) Name (C) index (D) All
42. Which function is used delete elements from List?  
(A) append() (B) remove() (C) sort() (D) reverse()
43. Which function is used to arrange items in a specific order?  
(A) append() (B) remove() (C) sort() (D) reverse()
44. Which function is used to add element in List?  
(A) append() (B) remove() (C) sort() (D) reverse()
45. Which function is used to change the order of List?  
(A) append() (B) sort() (C) reverse() (D) both b, c

#### 4.5

#### Testing & Debugging in Python

46. Process of finding & removing errors:  
(A) Test (B) Debug (C) Find (D) Acknowledge
47. Types of Testing:  
(A) 1 (B) 2 (C) 3 (D) 4
48. which type of testing check single part separately?  
(A) unit (B) Integration (C) Functional (D) Regression
49. Which testing ensure that new changes don't affect the functionality?  
(A) unit (B) Integration (C) Functional (D) Regression
50. which type of testing check different parts works together?  
(A) unit (B) Integration (C) Functional (D) Regression
51. Types of Debugging Techniques:  
(A) 1 (B) 2 (C) 3 (D) 4
52. PDB is a:  
(A) Python Translator (B) Python Module  
(C) Python Debugger (D) None of them



## Answers

- |       |       |       |       |       |       |       |
|-------|-------|-------|-------|-------|-------|-------|
| 1. C  | 2. D  | 3. B  | 4. A  | 5. D  | 6. D  | 7. B  |
| 8. D  | 9. C  | 10. A | 11. C | 12. C | 13. B | 14. C |
| 15. A | 16. D | 17. B | 18. B | 19. A | 20. C | 21. B |
| 22. C | 23. B | 24. C | 25. B | 26. D | 27. C | 28. A |
| 29. C | 30. C | 31. D | 32. C | 33. A | 34. A | 35. C |
| 36. B | 37. A | 38. B | 39. C | 40. D | 41. A | 42. A |
| 43. B | 44. C | 45. C | 46. C | 47. B | 48. C | 49. D |
| 50. D | 51. C | 52. C |       |       |       |       |

## Additional Short Question's

Short Answer Questions prepared in the light of new Examination Techniques  
(Knowledge, Understanding, Application, Analytical & Conceptual)

### Introduction

#### 1. What is the importance of control Structure?

**Ans:** Control structures are essential because they help us manage the decision-making process and the repetition of tasks in our programs. There are two main types of control structures, Decision making and Looping.

#### 4.1

### Decision Making

#### 2. Define decision making.

**Ans:** Decision making in programming allows the program to choose different actions based on conditions. This is similar to how we make decisions in real life.

**Example:** If you try to quit an application (mobile app), by pressing back button application ask;

'Do you want to quit': Yes or No

This shows the decision-making structure followed by the software.

#### 3. Define Condition with Example.

**Ans:** Condition is an expression contain operands with relational operators or logical operators or a combination of both which give results in boolean form, either it is true or either false.

**Example:**

- $5 > 7$  False
- $5 == 5$  True

#### 4. What is the importance of indentation in python programming?

**Ans:** In Python indentation defines the structure block or scope of your code. In Python proper indentation is mandatory. Incorrect indentation may cause error in your code (Indentation Error).



5. What is the limitation of if statement?

Ans: if statement shows output if the certain condition becomes true if the condition becomes false nothing will displayed on the screen.

6. Write the syntax of if-else statement.

Ans:

if condition:

code to run if the condition is true

else:

code to run if the condition is false

7. Define Nested if.

Ans: if statement with the body of another if statement is called nested if. This structure is used to apply complex multiple conditions.

Syntax of nested if statement:

if condition1:

if condition2:

code to run if both condition1 and condition2 are true

else:

code to run if condition1 is true but condition2 is false

else:

code to run if condition1 is false

## 4.2

## Looping Constructs

8. Define loop and named types of loop used in python?

Ans: If we want to repeat a statement or a set of statements for more than one times, we use loops. Loop help us to generate maximum output with the help of minimum input. In python we use two type of loops:

- While loop
- For loop

9. Differentiate between for loop and while loop

Ans:

| For Loop                                                                    | While Loop                                                                    |
|-----------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| Used where when total number of iterations are known in advance             | Used where when total number of iterations are not known in advance           |
| Also called counter control loop                                            | Also called sential control loop                                              |
| <b>Syntax:</b><br>for variable in sequence:<br>code to run for each element | <b>Syntax:</b><br>while condition:<br>code to run while the condition is true |

10. Define range function.

Ans: We can use the range() function to generate a sequence of numbers, which is often used in for loops.



### Syntax of range function:

range(start)

range(start, stop)

range(start, stop, step)

**Example:** Print First 5 Whole Numbers.

```
for i in range(5):
```

```
    print(i)
```

**Output:**

0

1

2

3

4

### 11. Define nested loop.

**Ans:** Nested loops are loops inside other loop. For each iteration of the outer loop, the inner loop runs completely. We use nested loops when we need to perform repetitive tasks within tasks.

## 4.3

## Libraries in Python

### 12. Define data structure.

**Ans:** A data structure refers to a particular format or method for organizing and storing data. In python, List, Tuple are popular data structures.

### 13. Define List.

**Ans:** In Python, libraries are like toolboxes full of useful tools that help you solve different problems without having to build everything from scratch. Libraries are like pre-built toolkits that you can use without having to write all the code yourself.

### 14. Describe an example to import a library in python.

**Ans: Code:**

```
import random
```

```
number = random.randint(1, 10)
```

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

**Output:**

The random number is: 3

### 15. Write an example to import statistics library in python.

**Ans:** import statistics

```
data = [23, 45, 67, 89, 12, 44, 56]
```

```
mean_value = statistics.mean(data)
```

```
print("The mean value is:", mean_value)
```

**Output:**

The mean value is: 48



**16. Explain creating a list.**

**Ans:** A list is created by placing items inside square brackets [ ], separated by commas. Lists can contain items of different types, such as numbers, strings, or even other lists.

**Example:** Create a list of your favorite fruits.

**Code:**

```
fruits = ["Mango", "Apple", "Banana"]
print(fruits)
```

**Output:**

```
['Mango', 'Apple', 'Banana']
```

**17. Describe modifying a list element.**

**Ans:** You can modify list items by accessing them via their index and assigning a new value.

**Example:** Change the first item in the list to "Orange" and add a new fruit "Pineapple".

**Code:**

```
fruits = ["Mango", "Apple", "Banana"]
fruits[0] = "Orange"
fruits.append("Pineapple")
print(fruits)
```

**Output:**

```
['Orange', 'Apple', 'Banana', 'Pineapple']
```

**18. Describe append methods.**

**Ans:** `append()` function:

This operation add new item at the end of the list.

**Example:** Add a new student to the list of students and print the list.

**Code:**

```
students = ["Ahmed", "Sara", "Ali"]
students.append("Hina")
print(students)
```

**Output:**

```
['Ahmed', 'Ali', 'Sara', 'Hina']
```

**19. Define testing and named its types.**

**Ans:** Testing is the process of running your code with various inputs to check if it behaves as expected. The goal is to find and fix any issues before the code is used in real-world applications.

**Types of Testing:**

Unit Testing

Integration Testing



Functional Testing

Regression Testing

20. Define debugging and named its techniques.

Ans: Debugging is the process of finding and fixing errors (bugs) in your code. It involves identifying the root cause of problems and making the necessary changes.

Common Debugging Techniques:

Print Statements

Debugging Tools

Error Messages

## Exercise Questions (Solved)



### Multiple Choice Questions

1. What does the range() function do in Python?

- (a) Generates a list of numbers
- (b) Creates a sequence of numbers
- (c) Calculates the sum of numbers
- (d) Prints a range of numbers

2. What is the output of the below code?

```
count = 0
```

```
for i in range(2):
```

```
    for j in range(3):
```

```
        count += 1
```

```
print(count)
```

- (a) 4
- (b) 5
- (c) 6
- (d) 7

3. Which method is used to add an item at the end of the list in Python?

- (a) insert()
- (b) append()
- (c) add()
- (d) extend()

4. What is the purpose of control structures in programming?

- (a) Only to make code longer
- (b) To manage decision-making and repetition of tasks
- (c) To create variables
- (d) To import libraries

5. Which statement is used to execute code when a condition is false?

- (a) if
- (b) elif
- (c) else
- (d) while

6. How do you prevent print() from adding a newline?

- (a) Use end=""
- (b) Use newline=False
- (c) Use break
- (d) Use continue

7. Which library is used to generate random numbers?

- (a) Datetime
- (b) statistics
- (c) random
- (d) math

8. What does list.append() do?

- (a) Removes the first item
- (b) Adds an item to the end
- (c) Sorts the list
- (d) Reverses the list



9. The type of testing which checks how different parts of the code work together is:
- (a) Unit testing
  - (b) Integration testing
  - (c) Functional testing
  - (d) Regression testing
10. Which debugging technique involves adding output statements?
- (a) Unit testing
  - (b) Print statements
  - (c) Error messages
  - (d) Integration testing

### Answers

- |        |        |        |        |         |
|--------|--------|--------|--------|---------|
| 1. (b) | 2. (c) | 3. (b) | 4. (b) | 5. (c)  |
| 6. (a) | 7. (c) | 8. (b) | 9. (b) | 10. (b) |

### Short Questions

1. Explain the use of `range()` function in for loop?

**Ans:** We can use the `range()` function to generate a sequence of numbers, which is often used in for loops.

**Example:** Print First 5 Whole Numbers.

```
for i in range(5):  
    print(i)
```

2. What is the difference between `append` and `remove` methods in Python list?

**Ans:** `append()` function:

This operation add new item at the end of the list.

**Example:** Add a new student to the list of students and print the list.

**Code:**

```
students = ["Ahmed", "Sara", "Ali"]  
students.append("Hina")  
print(students)
```

**remove () function:**

This operation is used to remove an item from the list

**Example:** Remove a student from the list of students and print the list.

**Code:**

```
students = ["Ahmed", "Sara", "Ali", "Hina", "Sara"]  
students.remove("Sara")  
print(students)
```

3. Define debugging.

**Ans:** Debugging is the process of finding and fixing errors (bugs) in your code. It involves identifying the root cause of problems and making the necessary changes.

4. What are the three parameters of the `range()` function?

**Ans:** Syntax of range function:

```
range(start, stop, step)
```

In this syntax, start parameter tells the starting of sequence of number. Stop shows



the final limit for generate the sequence of number and step show the increment between numbers.

**5. Name three list methods and their purpose.**

**Ans:** **append (item):** Adds an item to the end of the list.

**remove(item):** Removes the first occurrence of an item from the list.

**sort(item):** Sorts the list in ascending order.

**6. How do you import a library in Python?**

**Ans:** A library can be imported in python using a **"import"** keyword:

**Example:**

```
import random
number = random.randint(1, 10)
print("The random number is:", number)
```

**7. What is unit testing and why it is important?.**

**Ans:** Tests individual parts of the code (like functions or classes) in isolation. Python's unit test module is commonly used for this. It is important because programmer can check each individual component at the time of creation, it will increase the efficacy of working.

**8. How do you access the third element in a list?**

**Ans:** A list contains index number starting from 0 by default. If we have a list of 4 elements we can access third element with the help of index number 2.

**Example:**

```
marks = [76, 75, 77, 78]
print(marks[2])
```

**Output:**

77

**9. What is the purpose of the end parameter in print()?.**

**Ans:** In Python, the end statement is not a separate command, but rather an optional parameter of the print() function. By default, every print() in Python ends with a newline character, but you can change what gets printed at the end using the end parameter.

**Example:** Print two Strings on same line using "end" statement.

**Code:**

```
print("Azhar Shah", end=" ")
print("Ahsan")
```

**Output:**

Azhar Shah Ahsan

**10. How can we access an element from the list?**

**Ans:** A list contains index number starting from 0 by default. If we have access to elements from the list we can access element with the help of index number.

**Example:**

```
marks = [76, 75, 77, 78]
print(marks[2])
```



**Output:**

77

**11. How do you modify an existing item in a list?**

**Ans:** You can modify list items by accessing them via their index and assigning a new value.

**Example:** Change the first item in the list to "Orange" and add a new fruit "Pineapple".

**Code:**

```
fruits = ["Mango", "Apple", "Banana"]  
fruits[0] = "Orange"  
fruits.append("Pineapple")  
print(fruits)
```

**Output:**

```
['Orange', 'Apple', 'Banana', 'Pineapple']
```

### Long Questions

**1. Explain the different types of decision-making statements in Python with examples for each.**

**Ans:** Decision making in programming allows the program to choose different actions based on conditions. This is similar to how we make decisions in real life.

**Types of Decision statements:**

- if statement
- if else statement

**if Statement:**

The if statement make decisions based on conditions. If the condition is true, it runs a block of code.

**Syntax:**

if condition:

code to run if the condition is true

**Indentation:**

In Python indentation defines the structure block or scope of your code. In Python proper indentation is mandatory. Incorrect indentation may cause error in your code (Indentation Error).

**Example:**

```
Age = 21  
if Age > 18:  
    print("You are a teenager!!!!!!")
```

**if - else:**

The if-else statement allows us to execute one block of code if a condition is true and another block if the condition is false.

**Syntax of if-else statement:**

if condition:

code to run if the condition is true

else:

code to run if the condition is false



**Example:**

```
temperature = 15
```

```
if temperature > 30:
```

```
    print("It's a hot day")
```

```
else:
```

```
    print("It's not a hot day")
```

**Output:** It's not a hot day

## 2. Compare while loops and for loops.

**Ans: While loop:**

A **while** loop runs as long as a condition is true. It checks the condition before each iteration and stops running when the condition is no longer true. Basically, while loop is used where we don't know in advance how many times our loop repeats.

**Syntax of while loop:**

```
while condition:
```

```
    code to run while the condition is true
```

**Example:** Add 1 to a number until it reaches 10.

```
number = 1
```

```
while number < 10:
```

```
    print(number)
```

```
    number += 1
```

**Explanation:** In this example, the code starts with a number set to 1. The 'while' loop checks if the number is less than 10. If it is, the program prints the current value of the number and then adds 1 to it. This loop continues until the number reaches 10, at which point the loop stops running.

**For Loop:**

A **for** loop repeats a block of code a specific number of times. It is commonly used to iterate over a sequence (like a list, tuple, or string).

**Syntax of for loop:**

```
for variable in sequence:
```

```
    code to run for each element in the sequence
```

**Example:** Say "As-Salaam-Alaikum" to each friend in a list of friends.

```
friends = ["Sami", "Raza", "Moosa"]
```

```
for friend in friends:
```

```
    print("Welcome to ", friend)
```

**Output:**

```
Welcome to: Sami
```

```
Welcome to: Raza
```

```
Welcome to: Moosa
```

**Explanation:** In this example, the code goes through each friend in the list and prints a greeting message for each one.

## 3. Describe how to work with Python lists, including creation and modification.

**Ans:** For Answer See Q No. 10.



**4. Explain the importance of testing and debugging in programming. Discuss different debugging techniques.**

**Ans:** Testing is the process of running your code with various inputs to check if it behaves as expected. The goal is to find and fix any issues before the code is used in real-world applications.

Debugging is the process of finding and fixing errors (bugs) in your code. It involves identifying the root cause of problems and making the necessary changes.

**Common Debugging Techniques:**

- **Print Statements:** Adding print statements to check the values of variables at different stages of the code.
- **Debugging Tools:** Using tools like pdb (Python Debugger) to step through the code, inspect variables, and understand the flow of execution.
- **Error Messages:** Reading and interpreting error messages to locate the source of the problem.

**5. How do Python libraries enhance programming efficiency? Provide an example.**

**Ans:** Code:

For Answer See Q No. 9.

**6. Write a Python program using a loop that prints all the odd numbers between 1 and 20.**

**Ans:** Code:

```
# Print odd numbers using step parameter
print("Odd numbers from 1 to 20:")
for i in range(1, 21, 2):
    print(i, end=" ")
```

**7. Write a Python program that performs the following tasks:**

- Generates a list of 5 random numbers between 1 and 20.**
- Uses the statistics library to calculate and print the mean of the generated numbers.**

**Ans:** Code:

```
import random
import statistics
```

```
random_numbers = []
```

```
for i in range(5):
```

```
    random_numbers.append(random.randint(1, 20))
```

```
print("Generated random numbers:", random_numbers)
```

```
mean_value = statistics.mean(random_numbers)
```

```
print("Mean (average) of the numbers:", mean_value)
```



8. Write a Python program that performs the following operations:
- (a) Create a list of popular Pakistani dishes: ["Biryani", "Nihari", "Karahi", "Seekh Kebabs"].
  - (b) Add a new dish "Quorma" to the list.
  - (c) Change "Karahi" to "Chicken Karahi".
  - (d) Remove "Nihari" from the list.

Ans: Code:

```
pakistani_dishes = ["Biryani", "Nihari", "Karahi", "Seekh Kebabs"]
print("Initial list:", pakistani_dishes)
```

```
pakistani_dishes.append("Quorma")
pakistani_dishes[pakistani_dishes.index("Karahi")] = "Chicken Karahi"
pakistani_dishes.remove("Nihari")
print("Final list:", pakistani_dishes)
```

9. Write a code to print stars (asterisk) in rectangular pattern using nested loops with three rows and five columns.

Ans: Code:

```
for row in range(3):
    for col in range(5):
        print("*", end=" ")
    print()
```

output:

```
*****
*****
*****
```

10. Write the output of the following code:

```
print("=== Multiplication Table (1-5) ===")
for i in range(2, 4):      # Outer: 2 to 3
    for j in range(1, 11):  # Inner: 1 to 10
        print(f"{i} x {j} = {i*j:2d}", end=" ")
    print()                # New Line after each table
```

Ans:

=== Multiplication Table (1-5) ===

```
2 x 1 = 2 2 x 2 = 4 2 x 3 = 6 2 x 4 = 8 2 x 5 = 10 2 x 6 = 12 2 x 7 = 14 2 x 8 = 16
2 x 9 = 18 2 x 10 = 20
3 x 1 = 3 3 x 2 = 6 3 x 3 = 9 3 x 4 = 12 3 x 5 = 15 3 x 6 = 18 3 x 7 = 21 3 x 8 = 24
3 x 9 = 27 3 x 10 = 30
```