**Day-4 Morning Assessment**

**Functions**

1. A function in Python is a block of code that performs a specific task and can be reused.

2. Use the def keyword:  
def my\_function():  
   print("Amitha!")

3. Just use its name followed by parentheses:  
my\_function()

4. The return statement sends a value from a function back to the caller.

5. def add(a, b):  
   return a + b

6. Parameters are variables listed in a function definition.  
Arguments are actual values passed when calling the function.  
Example:  
python  
Copy code  
def greet(name):  # 'name' is a parameter  
   print("Hello", name)  
  
greet("Amitha")   # "Amitha" is an argument

7. The code inside the function won't run until it’s called.

8. def check\_even\_odd(num):  
   if num % 2 == 0:  
       print("Even")  
   else:  
       print("Odd")

9. A default argument is a parameter that has a default value.  
Example:  
def greet(name="Guest"):  
   print("Hello", name)

10. A keyword argument is when you pass arguments using the parameter name

11. def greet(name="User"):  
   print("Hello", name)

12. def square(n):  
   return n \* n

13. Yes, it can return multiple values using tuples:  
  
def operations(a, b):  
   return a + b, a - b  
  
sum\_, diff = operations(10, 5)

14. A function in Python is a block of code that performs a specific task and can be reused.  
  
   return a + b, a - b  
  
sum\_, diff = operations(10, 5)  
  
  
15. A variable declared outside all functions. It can be accessed from anywhere in the code.  
  
x = 10  # global  
def show():  
   print(x)  
  
16.A variable declared inside a function. It's only accessible within that function.  
def func():  
   y = 5  # local variable  
   print(y)  
  
17. def show\_info(name, age):  
   print("Name:", name)  
   print("Age:", age)  
  
18. The pass statement is used as a placeholder for code that hasn’t been written yet.  
def future\_function():  
   pass  
  
19. multiply = lambda a, b: a \* b  
  
20. def factorial(n):  
   if n == 0 or n == 1:  
       return 1

**While Loop**

21.A while loop keeps running as long as a given condition is True.  
  
22.i = 1  
while i <= 5:  
   print(i)  
   i += 1  
  
23. A loop that never ends because the condition is always true.  
  
24. Use the break statement:  
while True:  
   value = input("Type 'stop' to exit: ")  
   if value == "stop":  
       break  
  
25. i = 2  
while i <= 10:  
   print(i)  
   i += 2

o/p:

2

4

6

8

10  
  
26. for loop is When the number of iterations is known. while loop is When the number of iterations is unknown  
  
27. i = 1  
while i <= 10:  
   if i == 5:  
       break  
   print(i)  
   i += 1

o/p:

1

2

3

4  
  
28. i = 0  
while i < 5:  
   i += 1  
   if i == 3:  
       continue  
   print(i)

o/p:

1

2

4

5  
  
29. i = 1  
total = 0  
while i <= 100:  
   total += i  
   i += 1  
print("Sum:", total)

o/p: Sum: 5050  
  
30. while True:  
   text = input("Enter something (type 'exit' to quit): ")  
   if text.lower() == "exit":  
       break  
  
31. So the condition can be checked and updated properly. Without initialization, it may raise an error.  
  
32. i = 1  
while i <= 10:  
   print("5 x", i, "=", 5 \* i)  
   i += 1

o/p:

5 x 1 = 5

5 x 2 = 10

5 x 3 = 15

5 x 4 = 20

5 x 5 = 25

5 x 6 = 30

5 x 7 = 35

5 x 8 = 40

5 x 9 = 45

5 x 10 = 50  
  
33. It runs forever (infinite loop) unless broken with break.  
  
34. i = 10  
while i >= 1:  
   print(i)  
   i -= 1

o/p:

10

9

8

7

6

5

4

3

2

1  
  
35. i = 1  
while i <= 20:  
   print(i)  
   i += 2

o/p:

1

3

5

7

9

11

13

15

17

19  
  
36. Yes, the else block runs if the loop was not terminated by break.  
Example:  
i = 1  
while i <= 3:  
   print(i)  
   i += 1  
else:  
   print("Loop finished normally")

o/p: 3

Loop finished normally  
  
37. num = 123  
rev = 0  
while num > 0:  
   digit = num % 10  
   rev = rev \* 10 + digit  
   num = num // 10  
print("Reversed:", rev)

o/p: Reversed: 321  
  
38. To eventually make the condition false and avoid infinite loops.

39. num = 5  
fact = 1  
while num > 0:  
   fact \*= num  
   num -= 1  
print("Factorial:", fact)

o/p: Factorial: 120  
  
40. num = 121  
original = num  
rev = 0  
while num > 0:  
   digit = num % 10  
   rev = rev \* 10 + digit  
   num //= 10  
  
if original == rev:  
   print("Palindrome")  
else:  
   print("Not a palindrome")

o/p: Palindrome