Class XII
Computer Science
Short Questions & Answers

Topic: Functions in Python

Short Questions & Answers

1. Q: What are the three types of functions in Python? A: Built-in functions, functions defined in modules, and user-defined functions.
2. Q: How do you create a user-defined function in Python? A: By using the **def** keyword followed by the function name and a pair of parentheses containing the parameters.
3. Q: What are the two types of parameters in Python functions? A: Positional parameters and default parameters.
4. Q: How do you define a default parameter in a Python function? A: By assigning a default value to a parameter using the equal sign (e.g., **def my_function(x, y=5):**).
5. Q: Can a Python function return multiple values? A: Yes, by returning them as a tuple or another collection type (e.g., list or dictionary).
6. Q: What is the flow of execution in a Python program? A: The flow of execution refers to the order in which statements are executed during the program's run.
7. Q: What is the difference between global scope and local scope in Python? A: Global scope refers to variables that are accessible throughout the entire program, while local scope refers to variables that are only accessible within a specific function or block of code.
8. Q: How do you create a global variable in Python? A: By declaring a variable outside of any function or block of code.
9. Q: How do you access a global variable from within a function? A: By using the **global** keyword followed by the variable name.
10. Q: Can a local variable with the same name as a global variable be used within a function? A: Yes, but it will shadow the global variable, making it inaccessible within the function.
11. Q: What is a built-in function in Python? A: A function that is provided by the Python interpreter and is always available without needing to import any modules.
12. Q: Give an example of a built-in function in Python. A: **len()**, which returns the number of elements in a list or other iterable.
13. Q: What is a module in Python? A: A module is a file containing Python code, usually consisting of functions, classes, and variables, that can be imported into other Python programs.
14. Q: How do you import a function from a module in Python? A: By using the **from** keyword followed by the module name, the **import** keyword, and the function name.

kvcoders.in

15. Q: What is a positional parameter in a Python function? A: A positional parameter is a parameter that must be passed in a specific order when calling the function.
16. Q: What is a default parameter? A: A default parameter is a parameter that has a default value specified when defining the function, allowing the function to be called without providing a value for that parameter.
17. Q: What happens if you provide a value for a default parameter when calling a function? A: The provided value will override the default value specified in the function definition.
18. Q: What is the purpose of the **return** statement in a Python function? A: The **return** statement is used to exit a function and return a value to the caller.
19. Q: Can you use a function without a **return** statement? A: Yes, but the function will return **None** by default.
20. Q: What is the difference between arguments and parameters in Python? A: Parameters are the variables defined in the function definition, while arguments are the values passed to the function when it is called.

kvcoders.in

Programs:

1. Program to demonstrate a built-in function:

```
# Calculate the length of a list using the built-in len() function

my_list = [1, 2, 3, 4, 5]
length = len(my_list)
print("Length of the list:", length)
```

2. Program to demonstrate importing and using a function from a module:

```
# Calculate the square root of a number using the math module
import math

number = 16
sqrt_number = math.sqrt(number)
print("Square root of", number, "is", sqrt_number)
```

3. Program to demonstrate a user-defined function:

```
# Function to find the sum of two numbers
def add_numbers(a, b):
    return a + b

x = 5
y = 3
```

```
result = add_numbers(x, y)
print("Sum of", x, "and", y, "is", result)
```

4. Program to demonstrate a function with default parameters:

```
# Function to find the power of a number with default exponent 2 (square)
def power(base, exponent=2):
    return base ** exponent

number = 4
square = power(number)
print("Square of", number, "is", square)
```

5. Program to demonstrate a function with positional and default parameters:

```
# Function to find the area of a rectangle with optional height (default height = 1)
def area_of_rectangle(width, height=1):
    return width * height

width = 5
height = 3
area = area_of_rectangle(width, height)
print("Area of the rectangle is", area)
```

6. Program to demonstrate the use of global and local variables:

```
global_var = 10

def example_function():
    local_var = 5
    print("Global variable inside function:", global_var)
    print("Local variable inside function:", local_var)

example_function()
print("Global variable outside function:", global_var)
```

7. Program to demonstrate the use of the **global** keyword:

```
count = 0

def increment_count():
    global count
    count += 1

increment_count()
print("Count after increment:", count)
```

8. Program to demonstrate a function returning multiple values:

```
def min_max(numbers):
    return min(numbers), max(numbers)

my_list = [1, 5, 3, 8, 2]
```

kvcoders.in

```
min_val, max_val = min_max(my_list)
print("Minimum and maximum values are:", min_val, max_val)
```

9. Program to demonstrate the use of a function as a parameter:

```
def square(x):
    return x * x

def cube(x):
    return x * x * x

def apply_function(func, x):
    return func(x)

number = 4
result_square = apply_function(square, number)
result_cube = apply_function(cube, number)
print("Square of", number, "is", result_square)
print("Cube of", number, "is", result_cube)
```

10. Program to demonstrate a function with a variable number of arguments:

```
def sum_numbers(*args):
    return sum(args)

result = sum_numbers(1, 2, 3, 4, 5)
print("Sum of the numbers is:", result)
```

Practice Questions

1. Write a program that calculates the factorial of a given number using a user-defined function.
2. Write a program that determines whether a given number is a prime number using a user-defined function.
3. Write a program that finds the greatest common divisor (GCD) of two numbers using a user-defined function.
4. Write a program that calculates the sum of a series of numbers (e.g., 1+2+3+...+n) using a user-defined function.
5. Create a user-defined function that returns the length of a given string without using the built-in **len()** function.
6. Write a program that accepts a list of numbers and returns a new list containing only the even numbers using a user-defined function.
7. Create a user-defined function that accepts two strings and returns a concatenated string without using the **+** operator.
8. Write a program that calculates the sum of the diagonal elements of a square matrix using a user-defined function.

kvcoders.in

9. Write a program that accepts a list of numbers and returns the average of those numbers using a user-defined function.
10. Create a user-defined function that accepts a list of strings and returns the longest string in the list.
11. Write a program that sorts a list of numbers in ascending order using a user-defined function.
12. Write a program that reverses a given string without using the built-in **reverse()** function or string slicing.
13. Create a user-defined function that checks if a given word is a palindrome (reads the same forward and backward).
14. Write a program that calculates the Fibonacci sequence up to a given number using a user-defined function.
15. Create a user-defined function that accepts a list of numbers and returns the maximum and minimum values in the list.
16. Write a program that converts a given temperature from Celsius to Fahrenheit and vice versa using user-defined functions.
17. Write a program that finds the common elements between two given lists using a user-defined function.
18. Create a user-defined function that calculates the area and circumference of a circle, given its radius.
19. Write a program that counts the occurrences of a specific character in a given string using a user-defined function.
20. Write a program that accepts a list of numbers and returns a list of unique elements from the original list using a user-defined function.

kvcoders.in