Experiment No. 4
Creating functions, classes and objects using python
Date of Performance:
Date of Submission:

Experiment No. 4

Title: Creating functions, classes and objects using python

Aim: To study and create functions, classes and objects using python

Objective: To introduce functions, classes and objects in python

Theory:

A function is a block of code which only runs when it is called.

You can pass data, known as parameters, into a function.

A function can return data as a result.

A class is a user-defined blueprint or prototype from which objects are created. Classes provide a means of bundling data and functionality together. Creating a new class creates a new type of object, allowing new instances of that type to be made. Each class instance can have attributes attached to it for maintaining its state. Class instances can also have methods (defined by their class) for modifying their state.

To understand the need for creating a class let's consider an example, let's say you wanted to track the number of dogs that may have different attributes like breed, age. If a list is used, the first element could be the dog's breed while the second element could represent its age. Let's suppose there are 100 different dogs, then how would you know which element is supposed to be which? What if you wanted to add other properties to these dogs? This lacks organization and it's the exact need for classes.

Class creates a user-defined data structure, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A class is like a blueprint for an object.

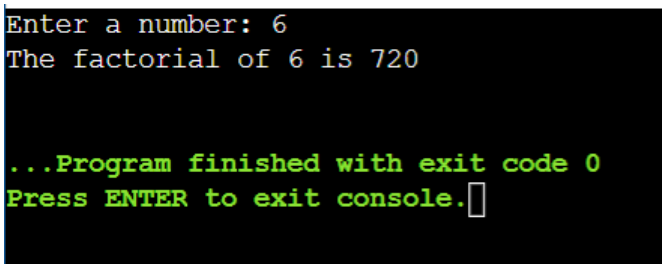
Code: Write a python program to find factorial of a number using function

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)


num = int(input("Enter a number: "))


if num < 0:
    print("Factorial does not exist for negative numbers")
elif num == 0:
    print("The factorial of 0 is 1")
else:
    print("The factorial of", num, "is", factorial(num))
```

Output:



Code: Write a python program to find the number is prime or not using function

```
def is_prime(n):
    if n <= 1:
        return False
    elif n <= 3:
        return True
    elif n % 2 == 0 or n % 3 == 0:
        return False
    i = 5
    while i * i <= n:
        if n % i == 0 or n % (i + 2) == 0:
            return False
        i += 6
    return True


num = int(input("Enter a number: "))


if is_prime(num):
    print(num, "is a prime number")
else:
    print(num, "is not a prime number")
```

Output:

```
Enter a number: 2
2 is a prime number


...Program finished with exit code 0
Press ENTER to exit console.
```

Code: Write a python program for student details using class

```
class Student:
    def __init__(self, name, age, grade):
        self.name = name
        self.age = age
        self.grade = grade

    def display_details(self):
        print("Name:", self.name)
        print("Age:", self.age)
        print("Grade:", self.grade)


student1 = Student("John", 15, "10th")
student2 = Student("Alice", 16, "11th")


print("Details of Student 1:")
```

```
student1.display_details()

print("\nDetails of Student 2:")

student2.display_details()
```

Output:

```
Details of Student 1:
Name: John
Age: 15
Grade: 10th

Details of Student 2:
Name: Alice
Age: 16
Grade: 11th
```

Conclusion:

Functions, classes, and objects are essential concepts in Python programming, enabling code organization, modularity, and reusability. Functions allow encapsulation of code blocks, classes provide blueprints for creating objects with attributes and methods, and objects represent instances of classes with specific data and behavior. Understanding how to define and use functions, classes, and objects is crucial for developing maintainable and scalable Python applications.