

# Vidyavardhini's College of Engineering & Technology

## Department of Computer Engineering

	•		3 T	4
Live	arim	ant	NIA	/
LEXL	erim	CIII	INO.	. 4
				-

Creating functions, classes and objects using python

Date of Performance:

Date of Submission:





### Department of Computer Engineering

### **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 calculate factorial(number):

factorial = 1



## Vidyavardhini's College of Engineering & Technology

### Department of Computer Engineering

```
if number < 0:
    return "Factorial is not defined for negative numbers."

elif number == 0:
    return 1

else:
    for i in range(1, number + 1):
        factorial *= i
        return factorial

num1 = int(input("Enter the number: "))

factorial1 = calculate_factorial(num1)

print(f"The factorial of {num1} is: {factorial1}")

Output:

Enter the number: 24
The factorial of 24 is: 620448401733239439360000</pre>
```

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

```
def is_prime(number):
    if number < 2:
        return False
    for i in range(2, int(number**0.5) + 1):
        if number % i == 0:
        return False
    return True</pre>
```



### Vidyavardhini's College of Engineering & Technology

### Department of Computer Engineering

```
num = int(input("Enter a number to check if it is prime: "))
if is prime(num):
  print(f"{num} is a prime number.")
else:
  print(f"{num} is not a prime number.")
Output:
Enter a number to check if it is prime: 23
23 is a prime number.
Enter a number to check if it is prime: 24
24 is not a prime number.
Code: Write a python program for student details using class
class Student:
  def init (self, name, roll number, grade):
    self.name = name
    self.roll number = roll number
    self.grade = grade
  def display details(self):
    print(f"Name: {self.name}")
```

print(f"Roll Number: {self.roll number}")

print(f"Grade: {self.grade}")

student1.display details()

student1 = Student("Siddhi Wade", "73", "c")

student2 = Student("Jidnyasa Naik", "79", "A")





### Department of Computer Engineering

student2.display\_details()

#### **Output:**

Name: Siddhi Wade Roll Number: 73

Grade: c

Name: Jidnyasa Naik Roll Number: 79

Grade: A

#### **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.