



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment – 1

- 1) **Title:** Introduction to Python Programming: Printing 'Hello, World!'
- 2) **Objective:** To introduce Python basics by writing and executing a program that prints "Hello, World!".
- 3) **Code and Output:**

+ Code + Text

```
 a="Hello World"  
print(a)  
 Hello World
```

- 4) **Viva Questions:**

- 1) What does this program do?
- 2) What is the purpose of the print() function?
- 3) Can we use single quotes instead of double quotes in the program?
- 4) Why do we use double quotes ("") or single quotes ('') around "Hello, World!"?

- 5) **MCQs:**

- 1) What is the correct syntax to print "Hello, World!" in Python?
  - a) echo "Hello, World!"
  - b) console.log("Hello, World!")
  - c) print("Hello, World!")
  - d) System.out.println("Hello, World!")

**Answer:** c) print("Hello, World!")

- 2) What is the type of the data "Hello, World!" in Python?
  - a) int
  - b) str
  - c) float
  - d) list

**Answer:** b) str



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

3) What does the print() function in Python do?

- a) Reads input from the user
- b) Displays output on the screen
- c) Terminates the program
- d) Saves data to a file

**Answer:** b) Displays output on the screen

4) Which version of Python treats print as a function?

- a) Python 2
- b) Python 3
- c) Both Python 2 and Python 3
- d) Neither Python 2 nor Python 3

**Answer:** b) Python 3

5) What will happen if you write print(Hello, World!) without quotes?

- a) It will print Hello, World!
- b) It will throw a SyntaxError
- c) It will throw a NameError
- d) It will execute without errors

**Answer:** c) It will throw a NameError



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-2

- 1) **Title:** Basic Arithmetic in Python: Adding Two Numbers.
- 2) **Objective:** To learn how to write a Python program that performs basic arithmetic operations by adding two numbers.
- 3) **Code and Output:**



```
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
sum = num1 + num2
print("The sum is:", sum)
```



```
Enter the first number: 2
Enter the second number: 4
The sum is: 6
```

---

### 4)Viva Questions:

- 1) What is the purpose of the input() function?
- 2) What will happen if the user enters a non-numeric value?
- 3) What is the output if the inputs are negative numbers?
- 4) Can this program handle very large numbers?

### 5)MCQs:

- 1) What is the default type of input taken by the input() function in Python?
  - a) Integer
  - b) String
  - c) Float
  - d) Boolean

**Answer:** b) String

- 2) What is the purpose of the + operator in this program?
  - a) To concatenate strings
  - b) To perform addition of numbers
  - c) To increment a number
  - d) None of the above

**Answer:** b) To perform addition of numbers



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

3) What is the purpose of the line **sum = num1 + num2** in the program?

- a) To declare the variables **num1** and **num2**
- b) To calculate the sum of **num1** and **num2**
- c) To print the sum of **num1** and **num2**
- d) To initialize the variable **sum** to zero

**Answer:** b) To calculate the sum of **num1** and **num2**

4) Which of the following is a better practice for naming the variable that stores the sum?

- a) sum
- b) total
- c) result
- d) both B and C

**Answer:** d) both b and c

5) If you want to print the sum with a formatted string, which of the following lines would you use instead of the original print statement?

- a) print("The sum is: {}".format(sum))
- b) print(f"The sum is: {sum}")
- c) print("The sum is: " + str(sum))
- d) All of the above

**Answer:** d) All of the above



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-3

- 1) **Title:** Swapping Two Variables in Python.
- 2) **Objective:** To understand the concept of variable manipulation by writing a Python program to swap the values of two variables.
- 3) **Code and Output:**



```
a = 5  
b = 10  
  
print("Before swapping:")  
print("a =", a)  
print("b =", b)  
  
a = a + b  
b = a - b  
a = a - b  
  
print("After swapping:")  
print("a =", a)  
print("b =", b)
```



Before swapping:

a = 5

b = 10

After swapping:

a = 10

b = 5

- 4) **Viva Questions:**

1) What are different methods to swap two variables in Python?

2) Can you explain how the arithmetic method of swapping works?

3) What are the advantages and disadvantages of using a temporary variable for swapping?

4) Why is tuple unpacking considered a Pythonic way to swap variables?

5) What are the potential issues with using arithmetic operations for swapping?



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## 5) MCQs:

1) What does the program do?

- a) Adds two numbers
- b) Multiplies two numbers
- c) Swaps the values of two variables
- d) Divides two numbers

**Answer:** c) Swaps the values of two variables

2) What is the formula used to update the value of b in the swapping logic?

- a)  $b = a + b$
- b)  $b = b - a$
- c)  $b = a - b$
- d)  $b = a + b - b$

**Answer:** c)  $b = a - b$

3) Why is no temporary variable used in this program?

- a) Temporary variables are unnecessary for swapping.
- b) Swapping can be achieved using arithmetic operations.
- c) It saves memory.
- d) All of the above

**Answer:** d) All of the above

4) Which line in the code swaps the value of a?

- a)  $a = a + b$
- b)  $b = a - b$
- c)  $a = a - b$
- d) `print("After swapping:")`

**Answer:** c)  $a = a - b$



## Experiment-4

- 1) **Title:** Generating Random Numbers in Python.
- 2) **Objective:** Write a Program to Generate a Random Number.
- 3) **Code and Output:**

```
▶ import random

# Generate and print a random number (convert float to integer)
random_number = int(random.uniform(1, 10))
print("The random number is:", random_number)
```

→ The random number is: 7

### 4) Viva Questions:

- 1) What does the random.uniform() function do?
- 2) Why do we use the int() function in this program?
- 3) What will be the range of random numbers generated by this program?
- 4) What is the purpose of the random module in Python?

### 5) MCQs:

- 1) Which of the following is NOT a function of the random module?

- a) randint()
- b) uniform()
- c) range()
- d) choice()

**Answer:** c) range()

- 2) What does the random.uniform(1, 11) function do?

- a) Generates a random integer between 1 and 11 (inclusive).
- b) Generates a random floating-point number between 1 and 11 (inclusive).
- c) Generates a random floating-point number between 1 (inclusive) and 11 (exclusive).
- d) Generates a random number between 1 and 10 (inclusive).

**Answer:** c) Generates a random floating-point number between 1 (inclusive) and 11 (exclusive).



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

**3) What is the purpose of the int() function in the program?**

- a) To round the number.
- b) To convert a floating-point number to an integer.
- c) To generate an integer directly.
- d) To print the random number.

**Answer:** b) To convert a floating-point number to an integer.

**4) What happens if the range is random.uniform(0, 0)?**

- a) The program will always output 0.0.
- b) The program will generate a random number between 0 and 1.
- c) The program will throw an error.
- d) The program will crash.

**Answer:** a) The program will always output 0.0.

**5) Which module needs to be imported to use random number functions?**

- a) math
- b) random
- c) numbers
- d) os

**Answer:** b) random



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-5

- 1) **Title:** Checking Odd or Even Numbers in Python.
- 2) **Objective:** To write a Python program that checks whether a given number is odd or even using conditional statements.
- 3) **Code and Output:**

```
▶ # Input a number from the user
    number = int(input("Enter a number: "))

    # Check if the number is even or odd
    if number % 2 == 0:
        print("The number is Even.")
    else:
        print("The number is Odd.)
```

→ Enter a number: 8  
The number is Even.

### 4)Viva Questions:

- 4) What is the purpose of the modulus operator (%) in this program?
- 5) What will happen if the user inputs a decimal number (e.g., 7.5)?
- 6) Why do we use int() to convert the input?

### 5)MCQs:

- 1) What is the purpose of the if-else statement in the program?
  - a) To check if the user entered a valid number
  - b) To compare two numbers.
  - c) To check if the number is divisible by 2 (even) or not (odd).
  - d) To display the number.

**Answer:** c) To check if the number is divisible by 2 (even) or not (odd).

- 2) Which Python function is used to get input from the user in this program?
  - a) print()
  - b) input()
  - c) str()
  - d) float()

**Answer:** b) input()



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-6

- 1) **Title:** Checking Leap Year in Python.
- 2) **Objective:** To write a Python program that determines whether a given year is a leap year or not based on specific conditions.
- 3) **Code and Output:**

```
▶ # Function to check if the year is a leap year
def is_leap_year(year):
    # If the year is divisible by 400, it's a leap year
    if (year % 400 == 0):
        return True
    # If the year is divisible by 100, it's not a leap year
    elif (year % 100 == 0):
        return False
    # If the year is divisible by 4, it's a leap year
    elif (year % 4 == 0):
        return True
    else:
        return False

# Input from the user
year = int(input("Enter a year: "))

# Check if the year is a leap year
if is_leap_year(year):
    print(f"{year} is a Leap Year.")
else:
    print(f"{year} is not a Leap Year.)
```

→ Enter a year: 2027  
2027 is not a Leap Year.

- 4) **Viva Questions:**

- 1) What are the rules for determining a leap year?
- 2) Can you explain the logic of your is\_leap\_year function?
- 3) Can you explain the importance of the elif statements in your function?



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## 5) MCQs:

- 1) What condition must a year meet to be classified as a leap year?
- a) It must be divisible by 2
  - b) It must be divisible by 3
  - c) It must be divisible by 4, but not by 100 unless also by 400
  - d) It must be a prime number

**Answer:** c) It must be divisible by 4, but not by 100 unless also by 400

- 2) What will happen if the user inputs a non-integer value?
- a) The program will run successfully.
  - b) The program will crash.
  - c) The program will return an error message.
  - d) The program will output "0".

**Answer:** b) The program will crash. (Without input validation)

- 3) What is the time complexity of the is\_leap\_year function?
- a) A) O(n)
  - b) O(log n)
  - c) O(1)
  - d) O( $n^2$ )

**Answer:** c) O(1)



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-7

- 1) **Title:** Finding the Largest Among Three Numbers in Python.
- 2) **Objective:** To write a Python program that compares three numbers and determines the largest among them using conditional statements.
- 3) **Code and Output:**

```
▶ a = int(input('enter first number'))  
b = int(input('enter second number'))  
c = int(input('enter third number'))  
if a>b and a>c:  
    print(f'the greatest number is {a}')  
elif b>a and b>c:  
    print(f'the greatest number is {b}')  
else:  
    print(f'the greatest number is {c}')
```

```
→ enter first number3  
enter second number7  
enter third number0  
the greatest number is 7
```

- 4) **Viva Questions:**

- 1) What are the limitations of this program?
- 2) What will happen if the user inputs a non-integer value?
- 3) What will the output be if all three numbers are equal?
- 4) What is the time complexity of this program?

### MCQs:

- 1) Which function is used to convert the user input into an integer?
  - a) str()
  - b) float()
  - c) int()
  - d) input()



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

2) What is the purpose of the given program?

- a) To find the smallest of three numbers
- b) To find the average of three numbers
- c) To find the greatest of three numbers
- d) To sort three numbers

**Answer:** c) To find the greatest of three numbers

3) Which statement correctly describes the conditional logic used in the program?

- a) It uses nested if statements.
- b) It uses only if statements.
- c) It uses if, elif, and else statements to determine the greatest number.
- d) It uses a switch-case statement.

**Answer:** c) It uses if, elif, and else statements to determine the greatest number.

4) Which of the following is a limitation of the program?

- a) It can only compare three numbers.
- b) It cannot handle negative numbers.
- c) It does not provide any output.
- d) It can only compare positive numbers.

**Answer:** a) It can only compare three numbers.



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-8

- 1) **Title:** Checking Prime Number in Python.
- 2) **Objective:** To write a Python program that checks whether a given number is prime by verifying if it has any divisors other than 1 and itself.
- 3) **Code and Output:**

```
▶ # Input from the user
    number = int(input("Enter a number: "))
    int: number
    8
# Check if the number
if number > 1:
    for i in range(2, number): # Check divisors from 2 to number-1
        if number % i == 0: # If divisible, it's not prime
            print(f"{number} is not a Prime Number.")
            break
    else:
        print(f"{number} is a Prime Number.")
else:
    print(f"{number} is not a Prime Number.")

→ Enter a number: 8
8 is not a Prime Number.
```

- 4) **Viva Questions:**

- 1) What is the time complexity of this program?
- 2) How could you modify the program to handle invalid input gracefully?
- 3) What does the break statement do in this program?
- 4) How would you test the program for edge cases?

- 5) **MCQs:**

1) What is the time complexity of the prime-checking algorithm in the program?

a) O(1)

b) O(log n)

c) O(n)

d) O( $n^2$ )

**Answer:** c) O(n)



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

2) What will the program output if the input is -5?

- a) -5 is a Prime Number.
- b) -5 is not a Prime Number.
- c) The program will crash.
- d) -5 is an even number

**Answer:** b) -5 is not a Prime Number.

3) Which of the following changes would improve the efficiency of the prime-checking algorithm?

- a) Check divisibility only up to the number itself.
- b) Check divisibility only up to the square root of the number.
- c) Check divisibility for all numbers from 1 to the number.
- d) Use a recursive function to check for primes.

**Answer:** b) Check divisibility only up to the square root of the number.

4) What will happen if the user inputs the number 0?

- a) The program will output "0 is a Prime Number."
- b) The program will output "0 is not a Prime Number."
- c) The program will return an error.
- d) The program will output "0 is an even number."

**Answer:** B) The program will output "0 is not a Prime Number."



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-9

- 1) **Title:** Printing All Prime Numbers in an Interval in Python.
- 2) **Objective:** To write a Python program that prints all prime numbers within a specified range by checking each number in the interval for primality.
- 3) **Code and Output:**

```
▶ def is_prime(number):  
    if number <= 1:  
        return False  
    for i in range(2, int(number**0.5) + 1):  
        if number % i == 0:  
            return False  
    return True  
  
def print_primes_in_interval(start, end):  
    print(f"Prime numbers between {start} and {end} are:")  
    for num in range(start, end + 1):  
        if is_prime(num):  
            print(num, end=" ")  
    print()  
  
# Example: Printing prime numbers in the interval 10 to 50  
start = int(input("Enter the start of the interval: "))  
end = int(input("Enter the end of the interval: "))  
  
print_primes_in_interval(start, end)
```

→ Enter the start of the interval: 4  
Enter the end of the interval: 9  
Prime numbers between 4 and 9 are:  
5 7

- 4) **Viva Questions:**

- 1) What are the possible optimizations for this program?
- 2) What is the purpose of the `is_prime` function?
- 3) Why do we use functions like `is_prime` instead of writing the logic in a single block?

## 5) MCQs:

- 1) Which data type is most suitable for storing the list of prime numbers?
  - a) Tuple
  - b) String
  - c) List
  - d) Dictionary

**Answer:** c) List



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

2) What is the time complexity of the `is_prime` function?

- a)  $O(n)$
- b)  $O(\log n)$
- c)  $O(\sqrt{n})$
- d)  $O(n \log n)$

**Answer:** c)  $O(\sqrt{n})$

3) What does the `is_prime` function return for the input 1?

- a) True
- b) False
- c) None
- d) An error

**Answer:** b) False

4) In the context of prime numbers, what does "composite number" mean?

- a) A number with only two divisors
- b) A number with more than two divisors
- c) A number that is always even
- d) A number that is divisible only by itself

**Answer:** b) A number with more than two divisors

5) What is the output format of the prime numbers printed by the program?

- a) Each prime number on a new line.
- b) All prime numbers in a single line, separated by commas.
- c) All prime numbers in a single line, separated by spaces.
- d) Prime numbers are printed in a list format.

**Answer:** c) All prime numbers in a single line, separated by spaces.



## Experiment-10

- 1) **Title:** Finding the Factorial of a Number in Python.
  - 2) **Objective:** To write a Python program that calculates the factorial of a given number using either iterative or recursive methods.
  - 3) **Code and Output:**

```
▶ number = int(input("Enter a number: "))
factorial = 1

if number < 0:
    print("Factorial is not defined for negative numbers.")
else:
    for i in range(1, number + 1):
        factorial *= i
    print(f"The factorial of {number} is {factorial}.")
```

→ Enter a number: 8  
The factorial of 8 is 40320.

#### **4) Viva Questions:**

- 4) What is the purpose of the program?
  - 5) What is the time complexity of the factorial calculation in this program?
  - 6) Can Python handle very large factorials? Why?

### **5) MCQs:**

- 1) What is the factorial of 0?

- a) 0
  - b) 1
  - c) Undefined
  - d) Infinity

**Answer:** b) 1

- 2) What is the time complexity of calculating the factorial using iteration?

- a) O(1)
  - b) O(log n)
  - c) O(n)
  - d) O( $n^2$ )

**Answer:** c) O(n)



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment-11

- 1) **Title:** Printing the Fibonacci Sequence in Python.
- 2) **Objective:** To write a Python program that generates and prints the Fibonacci sequence up to a specified number of terms.
- 3) **Code and Output:**

```
▶ n = int(input("Enter the number of terms: "))

a, b = 0, 1

if n <= 0:
    print("Please enter a positive integer.")
elif n == 1:
    print("Fibonacci sequence: 0")
else:
    print("Fibonacci sequence:", end=" ")
    print(a, b, end=" ")
    for _ in range(2, n):
        c = a + b
        print(c, end=" ")
        a, b = b, c
    print()
```

→ Enter the number of terms: 5  
Fibonacci sequence: 0 1 1 2 3

- 4) **Viva Questions:**

- 1) What is the Fibonacci sequence?
- 2) Can you generate the Fibonacci sequence using recursion?
- 3) What are some real-world applications of the Fibonacci sequence?
- 4) Why does the program check if  $n \leq 0$ ?

- 5) **MCQs:**

- 1) What is the first number in the Fibonacci sequence?
  - a) 0
  - b) 1
  - c) 2
  - d) Undefined



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

2) Which of the following is NOT true about the Fibonacci sequence?

- a) Each number is the sum of the two preceding numbers.
- b) The sequence starts with 0 and 1.
- c) Fibonacci numbers can be negative.
- d) Fibonacci numbers appear in natural patterns.

**Answer:** c) Fibonacci numbers can be negative.

3) What is the logic used to generate the Fibonacci sequence in this program?

- a) Add the two preceding numbers to get the next number.
- b) Multiply the two preceding numbers.
- c) Subtract the preceding number from the current number.
- d) Divide the current number by the preceding number.

**Answer:** a) Add the two preceding numbers to get the next number.

4) What is the first number in the Fibonacci sequence?

- a) 0
- b) 1
- c) 2
- d) Undefined

**Answer:** a) 0

5) What is the time complexity of the prime-checking algorithm in the program?

- a) O(1)
- b) O(log n)
- c) O(n)
- d) O( $n^2$ )

**Answer:** c) O(n)



## Experiment – 12

- 1) **Title:** Identifying Syntax and Semantic Errors in English Sentences.
- 2) **Objective:** Write an English sentence with understandable semantics but incorrect syntax.  
Write another English sentence which has correct syntax but has semantic errors.
- 3) **Code and Output:**

```
▶ # Incorrect syntax (grammatical mistake) but the meaning is clear
    sentence_with_syntax_error = "He go to the park every day."
    print("Sentence with syntax error:", sentence_with_syntax_error)
# Correct grammar but logically incorrect
    sentence_with_semantic_error = "The green cat drove a car to the moon."
    print("Sentence with semantic error:", sentence_with_semantic_error)
```

→ Sentence with syntax error: He go to the park every day.  
Sentence with semantic error: The green cat drove a car to the moon.

### Explanation:

- **Syntax Error Example:** "He go to the park every day."  
This sentence has a grammatical error (go instead of goes), but the meaning is still clear.
- **Semantic Error Example:** "The green cat drove a car to the moon."  
The grammar is correct, but the sentence doesn't make logical sense.



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment – 13

- 1) **Title:** Combining Two Favorite Foods in Python.
- 2) **Objective:** Write a program that allows a user to enter his or her two favorite foods. The program should then print out the name of a new food by joining the original food names together.
- 3) **Code and Output:**



```
# Prompt the user for their two favorite foods
food1 = input("Enter your first favorite food: ")
food2 = input("Enter your second favorite food: ")

# Combine the two food names
new_food = food1 + food2

# Print the name of the new food
print(f"Your new food is: {new_food}")
```



```
Enter your first favorite food: sandwich
Enter your second favorite food: pizza
Your new food is: sandwichpizza
```

- 4) **Viva Questions:**

1. What is the purpose of this program?
2. How does the program take user input and store it in variables?
3. Can you explain the process of concatenating strings in Python?
4. What is the significance of using the input() function in this program?

- 5) **MCQs:**

- 1. Which function is used to take user input in Python?**

- A) print()
- B) input()
- C) str()
- D) join()

**Answer:** B) input()



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## 2. What will be the output of the following code if the user enters "Pizza" and "Burger"?

```
food1 = input("Enter your favorite food: ")  
food2 = input("Enter another favorite food: ")  
  
print(food1 + food2)
```

- A) "Pizza Burger"
- B) "PizzaBurger"
- C) "Pizza + Burger"
- D) "None of the above"

**Answer:** B) "PizzaBurger"

## 3. How can you modify the program to add a space between the two food names when concatenated?

- A) food1 + " " + food2
- B) food1 + food2 + " "
- C) food1 + "food2"
- D) food1 + str(food2)

**Answer:** A) food1 + " " + food2

## 4. What is the main concept demonstrated in this program?

- A) String concatenation
- B) File handling
- C) Conditional statements
- D) Looping

**Answer:** A) String concatenation



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment – 14

- 1) **Title:** Creating a Custom Sum Function in Python.
- 2) **Objective:** Write your own sum function called mySum that takes a list as a parameter and returns the accumulated sum.
- 3) **Code and Output:**

```
▶ def mySum(numbers):  
    # Initialize a variable to store the sum  
    total = 0  
    # Loop through each number in the list and add it to the total  
    for num in numbers:  
        total += num  
    return total  
  
# Example usage  
numbers = [1, 2, 9, 4, 6]  
result = mySum(numbers)  
print(f"The sum of the list is: {result}")
```

→ The sum of the list is: 22

- 4) **Viva Questions:**

1. What is the purpose of the mySum function in this program?
2. How does the mySum function accumulate the sum of the list elements?
3. Can you explain how the for loop is used in the mySum function?
4. What type of data is passed as a parameter to the mySum function?

- 5) **MCQS:**

1. **What is the primary purpose of the mySum function?**

- A) To multiply all elements of the list
- B) To accumulate the sum of all elements in the list
- C) To sort the list in ascending order
- D) To find the maximum element in the list

**Answer:** B) To accumulate the sum of all elements in the list



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## 2. Which of the following is the correct way to implement the mySum function?

```
def mySum(lst):
```

```
    total = 0
```

```
    for num in lst:
```

```
        total += num
```

```
    return total
```

- A) The function uses a loop to add each element in the list to total
- B) The function multiplies each element in the list and returns the product
- C) The function subtracts each element from total
- D) The function divides total by each element of the list

**Answer:** A) The function uses a loop to add each element in the list to total

## 3. What will the following call to mySum([1, 2, 3, 4]) return?

- A) 24
- B) 10
- C) 12
- D) Error

**Answer:** B) 10

## 4. Which of the following is the parameter type for the mySum function?

- A) Integer
- B) String
- C) List
- D) Dictionary

**Answer:** C) List



## Experiment-15

**1. Title:** create a program in python using array.

**2. Objective:** wap idea is to store multiple items of the same type together.

**3. Code and output:**

```
▶ import array as arr  
  
# creating array of integers  
a = arr.array('i', [1, 2, 3])  
  
# accessing Araay  
print("First Element is", a[0])  
  
# Adding element to array  
a.append(5)  
print(a)
```

```
⇨ First Element is 1  
array('i', [1, 2, 3, 5])
```

**4. Viva Question:**

1. How many arrays are there in Python?
2. What is the default value of Array?
3. Can we declare array size as a negative number?

**5. MCQS:**

**1. What is an array in Python?**

- a) A collection of elements of different data types
- b) A collection of elements of the same data type
- c) A key-value pair collection
- d) A collection of functions

Answer: b) A collection of elements of the same data type

**2. How do you import the array module in Python?**

- a) import array
- b) import arrays
- c) from python import array
- d) include array

Answer: a) import array



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

**3. Which of the following is the correct way to create an array of integers in Python?**

- a) array.array('i', [1, 2, 3])
- b) array('i', [1, 2, 3])
- c) [1, 2, 3]
- d) (1, 2, 3)

Answer: a) array.array('i', [1, 2, 3])

**4. How do you add an element to the end of an array?**

- a) append(element)
- b) push(element)
- c) add(element)
- d) insert(element)

Answer: a) append(element)



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment:16

1. **Title:** list in python.
2. **Objective:** perform operations in a list.
3. **Code and Output:**

- CREATE A LIST:

```
❶ thislist = ["apple", "banana", "cherry"]
print(thislist)
```

```
['apple', 'banana', 'cherry']
```

- Allow Duplicates:

Since lists are indexed, lists can have items with the same value:

### ALLOW DUPLICATE

```
❶ thislist = ["apple", "banana", "cherry", "apple", "cherry"]
print(thislist)
```

```
['apple', 'banana', 'cherry', 'apple', 'cherry']
```

- List Length:

To determine how many items a list has, use the `len()` function

### list length

```
❶ thislist = ["apple", "banana", "cherry"]
print(len(thislist))
```

```
3
```



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

- Change List Items:

Change Item Value

To change the value of a specific item, refer to the index number:

```
✓ 08  ⏪ thislist = ["apple", "banana", "cherry"]
    thislist[1] = "blackcurrant"
    print(thislist)
```

- Change a Range of Item Values:

```
✓ 08  ⏪ thislist = ["apple", "banana", "cherry", "orange", "kiwi", "mango"]
    thislist[1:3] = ["blackcurrant", "watermelon"]
    print(thislist)
→   ['apple', 'blackcurrant', 'watermelon', 'orange', 'kiwi', 'mango']
```

- Insert Items:

To insert a new list item, without replacing any of the existing values, we can use the `insert()` method.

The `insert()` method inserts an item at the specified index:

```
✓ 08  ⏪ thislist = ["apple", "banana", "cherry"]
    thislist.insert(2, "watermelon")
    print(thislist)
>
→   ['apple', 'banana', 'watermelon', 'cherry']
```

- Sort the list alphabetically:

```
✓ 08  ⏪ thislist = ["orange", "mango", "kiwi", "pineapple", "banana"]
    thislist.sort()
    print(thislist)
→   ['banana', 'kiwi', 'mango', 'orange', 'pineapple']
```



## Join Two Lists:

```
Da  list1 = ["a", "b", "c"]
list2 = [1, 2, 3]

list3 = list1 + list2
print(list3)
```

```
['a', 'b', 'c', 1, 2, 3]
```

## 4. VA QUESTION:

1. What is a list in Python? How is it different from other sequences like strings or tuples?
2. How do you create a list in Python? Give an example.
3. What are the key characteristics of a list in Python?
4. Can a list contain elements of different data types? Give an example.

## 5. MCQ:

1. Which of the following is the correct way to create a list in Python?  
a) list = (1, 2, 3)  
b) list = [1, 2, 3]  
c) list = {1, 2, 3}  
d) list = "1, 2, 3"

Answer: b) list = [1, 2, 3]

2. What will be the output of the following code?

```
python
Copy code
my_list=[10, 20, 30, 40]
print(my_list[2])
a) 10
b) 20
c) 30
d) 40
```

Answer: c) 30

3. Which method is used to add an element to the end of a list?  
a) insert()  
b) append()  
c) extend()  
d) add()

Answer: b) append()



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment :17

1. **Title:** tuple in python.

2. **Objective:** perform operation in python using tuple.

3. **Code & Output:**

### Python - Loop Tuples

```
✓ 0s  thistuple = ("apple", "banana", "cherry")
    for x in thistuple:
        print(x)

→ apple
     banana
     cherry
```

### Loop Through the Index Numbers

You can also loop through the tuple items by referring to their index number.

Use the range() and len() functions to create a suitable iterable

```
✓ 0s  thistuple = ("apple", "banana", "cherry")
    for i in range(len(thistuple)):
        print(thistuple[i])

→ apple
     banana
     cherry
```

### Using a While Loop:

```
✓ 0s  thistuple = ("apple", "banana", "cherry")
    i = 0
    while i < len(thistuple):
        print(thistuple[i])
        i = i + 1

→ apple
     banana
     cherry
```

### Python - Join Tuples:

Join Two Tuples

To join two or more tuples you can use the + operator:



cherry

```
Os ➜ tuple1 = ("a", "b", "c")
tuple2 = (1, 2, 3)

tuple3 = tuple1 + tuple2
print(tuple3)
```

```
→ ('a', 'b', 'c', 1, 2, 3)
```

Multiply the fruits tuple by 2:

```
Os ➜ fruits = ("apple", "banana", "cherry")
mytuple = fruits * 2

print(mytuple)
→ ('apple', 'banana', 'cherry', 'apple', 'banana', 'cherry')
```

## Python - Tuple Methods:

Python has two built-in methods that you can use on tuples.

Method	Description
<a href="#">count()</a>	Returns the number of times a specified value occurs in a tuple
<a href="#">index()</a>	Searches the tuple for a specified value and returns the position of where it was found

## 4. VA QUESTION:

1. What is a tuple in Python? How is it different from a list?
2. How do you create a tuple in Python? Provide an example.
3. Can a tuple contain elements of different data types? Give an example.
4. What does it mean when we say tuples are immutable?

## 5. MCQ:

1. Which of the following is the correct way to create a tuple in Python?
  - a) tuple = [1, 2, 3]
  - b) tuple = (1, 2, 3)
  - c) tuple = {1, 2, 3}
  - d) tuple = 1, 2, 3

Answer: b) tuple = (1, 2, 3) and d) tuple = 1, 2, 3

2. What is the key difference between a tuple and a list?
  - a) Tuples are mutable, and lists are immutable.
  - b) Tuples are immutable, and lists are mutable.
  - c) Tuples can only hold integers.
  - d) Lists can contain duplicates, but tuples cannot.



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

Answer: b) Tuples are immutable, and lists are mutable.

3. How do you create a tuple with a single element?

- a) (1)
- b) (1,)
- c) [1]
- d) tuple(1)

Answer: b) (1,)



## Experiment:18

1. Title: set in python.

2. Objective: perform operation in set methods.

3. Code & Output:

### Access Items:

```
08  thisset = {"apple", "banana", "cherry"}  
  
    for x in thisset:  
        print(x)  
  
09      banana  
          apple  
            cherry
```

### Loop Items:

You can loop through the set items by using a for loop:

```
08  thisset = {"apple", "banana", "cherry"}  
09  for x in thisset:  
10      print(x)  
  
11      banana  
        apple  
          cherry
```

### Join Sets:

- There are several ways to join two or more sets in Python.
- The union() and update() methods joins all items from both sets.
- The intersection() method keeps ONLY the duplicates.
- The difference() method keeps the items from the first set that are not in the other set(s).
- The symmetric\_difference() method keeps all items EXCEPT the duplicates.

### Join Multiple Sets:

```
08  set1 = {"a", "b", "c"}  
09  set2 = {1, 2, 3}  
10  set3 = {"John", "Elena"}  
11  set4 = {"apple", "bananas", "cherry"}  
  
12  myset = set1.union(set2, set3, set4)  
13  print(myset)  
  
14  {1, 2, 3, 'apple', 'c', 'John', 'a', 'Elena', 'b', 'bananas', 'cherry'}
```

### Use | to join two sets:



```
✓ 0 ❶ set1 = {"a", "b", "c"}  
❷ set2 = {1, 2, 3}  
❸ set3 = {"John", "Elena"}  
❹ set4 = {"apple", "bananas", "cherry"}  
  
❺ myset = set1 | set2 | set3 | set4  
print(myset)  
  
→ {1, 2, 3, 'apple', 'c', 'John', 'a', 'Elena', 'b', 'bananas', 'cherry'}
```

## 4. VA QUESTION:

1. What is a set in Python? How is it different from a list or tuple?
2. How do you create a set in Python? Provide an example.
3. Can a set contain duplicate elements? Why or why not?
4. What types of data can be stored in a set?

## 5. MCQ:

1. Which of the following is the correct way to create a set in Python?
  - a) set1 = {1, 2, 3}
  - b) set1 = [1, 2, 3]
  - c) set1 = (1, 2, 3)
  - d) set1 = {1: "a", 2: "b"}

Answer: a) set1 = {1, 2, 3}

2. What is the key characteristic of a set in Python?
  - a) It is mutable and ordered.
  - b) It is immutable and unordered.
  - c) It is mutable and unordered.
  - d) It is immutable and ordered.

Answer: c) It is mutable and unordered.



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment:19

1. Title: dictionary in python.

2. Objective: perform operation in dictionary.

3. Code & Output:

- Create and print a dictionary:

```
✓ 1s 0s thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
print(thisdict)  
  
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

- Loop Through a Dictionary:

```
✓ 0s for x in thisdict.keys():  
    print(x)  
  
brand  
model  
year
```

- Copy a Dictionary:

```
✓ 0s thisdict = {  
    "brand": "Ford",  
    "model": "Mustang",  
    "year": 1964  
}  
mydict = thisdict.copy()  
print(mydict)  
  
{'brand': 'Ford', 'model': 'Mustang', 'year': 1964}
```

- Nested Dictionaries:



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

```
myfamily = {  
    "child1": {  
        "name": "Emil",  
        "year": 2004  
    },  
    "child2": {  
        "name": "Tobias",  
        "year": 2007  
    },  
    "child3": {  
        "name": "Linus",  
        "year": 2011  
    }  
}  
print(myfamily)  
  
{'child1': {'name': 'Emil', 'year': 2004}, 'child2': {'name': 'Tobias', 'year': 2007}, 'child3': {'name': 'Linus', 'year': 2011}}
```

## 4. VA QUESTION:

1. What is a dictionary in Python? How is it different from a list?
2. How do you create a dictionary in Python? Provide an example.
3. What are the key characteristics of a dictionary?
4. Can a dictionary have duplicate keys? Why or why not?
5. What types of data can be used as dictionary keys?

## 5. MCQ:

1. Which of the following is the correct way to create a dictionary in Python?

- a) dict1 = [1, 2, 3]
- b) dict1 = {1: "a", 2: "b", 3: "c"}
- c) dict1 = (1, 2, 3)
- d) dict1 = {1, 2, 3}

Answer: b) dict1 = {1: "a", 2: "b", 3: "c"}

2. What will the following code output?

python

Copy code

```
dict1 = {"a": 1, "b": 2, "c": 3}
```

```
print(dict1["b"])
```

- a) 1
- b) 2
- c) 3
- d) Error

Answer: b) 2



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment:20

1. **Title:** function in python.
2. **Objective:** operations on function .

### 3. Code & Output:

- Creating a Function:

```
def my_function():
    print("Hello from a function")
```

- Calling a Function:

```
✓ 0 def my_function():
    print("Hello from a function")

my_function()

→ Hello from a function
```

- Arguments:

1. Information can be passed into functions as arguments.
2. Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.
3. The following example has a function with one argument (fname). When the function is called, we pass along a first name, which is used inside the function to print the full name.

```
✓ 0 def my_function(fname):
    print(fname + " Refsnes")

my_function("Emil")
my_function("Tobias")
my_function("Linus")

→ Emil Refsnes
    Tobias Refsnes
    Linus Refsnes
```

- Arbitrary Arguments, \*args:

```
✓ 0 def my_function(*kids):
    print("The youngest child is " + kids[2])

my_function("Emil", "Tobias", "Linus")

→ The youngest child is Linus
```

- Default Parameter Value

```
✓ 0 def my_function(country = "Norway"):
    print("I am from " + country)

my_function("Sweden")
my_function("India")
my_function()
my_function("Brazil")

→ I am from Sweden
    I am from India
    I am from Norway
    I am from Brazil
```



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## 4. VA QUESTION:

1. What is a function in Python? Why do we use functions?
2. How do you define a function in Python? Provide an example.
3. What is the difference between a function and a method in Python?
4. What is the significance of the return statement in a function?
5. Can a function have multiple return statements? Explain.

## 5. MCQ:

1. How do you define a function in Python?

- a) function my\_function()
- b) def my\_function():
- c) func my\_function()
- d) define my\_function():

Answer: b) def my\_function()

2. Which of the following is true about functions in Python?

- a) Functions must always return a value.
- b) Functions can return multiple values.
- c) Functions cannot take arguments.
- d) Functions are defined using the function keyword.

Answer: b) Functions can return multiple values.



## Experiment:21

1. **Title:** recursion in python.
2. **Objective:** used recursion method.

### 3. Code & Output:

Recursion

1. Python also accepts function recursion, which means a defined function can call itself.
2. Recursion is a common mathematical and programming concept. It means that a function calls itself. This has the benefit of meaning that you can loop through data to reach a result.
3. The developer should be very careful with recursion as it can be quite easy to slip into writing a function which never terminates, or one that uses excess amounts of memory or processor power. However, when written correctly recursion can be a very efficient and mathematically-elegant approach to programming.
4. In this example, tri\_recursion() is a function that we have defined to call itself ("recurse"). We use the k variable as the data, which decrements (-1) every time we recurse. The recursion ends when the condition is not greater than 0 (i.e. when it is 0).
5. To a new developer it can take some time to work out how exactly this works, best way to find out is by testing and modifying.

```
def tri_recursion(k):
    if(k > 0):
        result = k + tri_recursion(k - 1)
        print(result)
    else:
        result = 0
    return result

print("Recursion Example Results:")
tri_recursion(6)
```

Recursion Example Results:  
1  
3  
6  
10  
15  
21  
28

### 4. VA QUESTION:

1. What is recursion in Python?
2. How is recursion different from iteration?
3. What is the base case in recursion? Why is it important?

### 5. MCQ:

1. What is recursion in Python?
  - a) A function calling another function
  - b) A function calling itself
  - c) A loop inside a function
  - d) A function without arguments



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

**Answer:** b) A function calling itself

2. Which of the following is essential for a recursive function?

- a) An infinite loop
- b) A base case
- c) A return statement
- d) Global variables

**Answer:** b) A base case



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

## Experiment:22

1. Title: class in python.

2. Objective: use single & multiple level inheritance.

### 3. Code & Output:

- Single level inheritance:

```
✓  # Base class (Parent class)
0s class Animal:
    def __init__(self, name):
        self.name = name

    def speak(self):
        return f"{self.name} makes a sound."

# Derived class (Child class)
class Dog(Animal):
    def speak(self):
        return f"{self.name} barks."

# Creating objects for both classes
animal = Animal("Generic Animal")
print(animal.speak()) # Output: Generic Animal makes a sound.

dog = Dog("Buddy")
print(dog.speak()) # Output: Buddy barks.
```

Generic Animal makes a sound.  
Buddy barks.

- Multiple inheritance:

```
✓  # Base class 1
0s class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def show_person_details(self):
        return f"Name: {self.name}, Age: {self.age}"

# Base class 2
class Employee:
    def __init__(self, employee_id, salary):
        self.employee_id = employee_id
        self.salary = salary

    def show_employee_details(self):
        return f"Employee ID: {self.employee_id}, Salary: {self.salary}"

# Derived class
class Manager(Person, Employee):
    def __init__(self, name, age, employee_id, salary, department):
        # Initialize attributes from both parent classes
        Person.__init__(self, name, age)
        Employee.__init__(self, employee_id, salary)
        self.department = department

    def show_manager_details(self):
        return f"{self.show_person_details()}, Employee ID: {self.employee_id}, Salary: {self.salary}, Department: {self.department}"

# Creating an object of the Manager class
manager = Manager("Alice", 35, "E123", 75000, "IT")
print(manager.show_manager_details())
```

Name: Alice, Age: 35, Employee ID: E123, Salary: 75000, Department: IT



# Shri Vaishnav Vidyapeeth Vishwavidyalaya

#### 4. VA QUESTION:

1. What is inheritance in Python? Why is it used?
2. What is single inheritance? Provide an example.
3. What is multiple inheritance? Provide an example.
4. What is the difference between single and multiple inheritance?
5. What are the benefits of using inheritance in Python?

#### 5. MCQ:

1. What is single inheritance in Python?
  - a) A class inheriting from multiple parent classes
  - b) A class inheriting from only one parent class
  - c) A class with no inheritance
  - d) Multiple classes inheriting from one parent class

Answer: b) A class inheriting from only one parent class

2. Which of the following is correct about single inheritance?
  - a) The child class cannot override methods of the parent class.
  - b) The child class has access to all public and protected members of the parent class.
  - c) Single inheritance is not supported in Python.
  - d) A child class can inherit from multiple classes in single inheritance.

Answer: b) The child class has access to all public and protected members of the parent class.