PNS SCHOOL OF ENGINEERING AND TECHNOLOGY

NISHAMANI VIHAR, MARSHAGHAI, KENDRAPARA

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

LAB MANUAL

Semester: 5th Semester

Subject: Python Programming Lab (Pr3)

Prepared By:

Mr. BISWARANJAN SWAIN

HOD DEPT. OF CSE

List of Practical:

No.1

a) **Create a string containing at least five words and store it in a variable.**
b) **Print out the string**

Sol*: my_string = "The quick brown fox jumps"*
print(my_string)

c) Convert the string to a list of words using the string split method.

```
my_string = "The quick brown fox jumps"
# Convert the string to a list of words using split()
word_list = my_string.split()
# Print the resulting list
print(word_list)
```

Output:

['The', 'quick', 'brown', 'fox', 'jumps']

Explanation:

- The split() method, by default, splits the string at whitespace (spaces, tabs, or newlines) into a list of substrings.
- Each word in the string "The quick brown fox jumps" becomes an element in the list word_list.
- The print statement displays the resulting list.

c) Sort the list into reverse alphabetical order using some of the list methods
d) Print out the sorted, reversed list of words.

```
# Create a string with five words
my_string = "The quick brown fox jumps"

# Convert the string to a list of words using split()
word_list = my_string.split()

# Sort the list in reverse alphabetical order
word_list.sort(reverse=True)

# Print the sorted list
print(word_list)
```

Output:

['The', 'quick', 'jumps', 'fox', 'brown']

Explanation:

- The split() method converts the string "The quick brown fox jumps" into the list ['The', 'quick', 'brown', 'fox', 'jumps'].
- The sort() method is a built-in list method that sorts the list in place. By default, it sorts in ascending alphabetical order for strings.
- Setting the parameter reverse=True in sort(reverse=True) reverses the order, resulting in descending alphabetical order (Z to A).
- The sorted list is printed, showing the words in reverse alphabetical order.

No. 2 Write a program that determines whether the number is prime.

Sol:

```
# Program to check if a number is prime
def is_prime(number):
    # Check if number is less than or equal to 1
    if number <= 1:
        return False

    # Check for divisibility from 2 to square root of number
    for i in range(2, int(number ** 0.5) + 1):
        if number % i == 0:
            return False

    return True

# Input from user
try:
    num = int(input("Enter a number to check if it is prime: "))

    # Call the function and display result
    if is_prime(num):
        print(f"{num} is a prime number.")
    else:
        print(f"{num} is not a prime number.")
except ValueError:
    print("Please enter a valid integer.")
```

No.3 Find all numbers which are multiple of 17, but not the multiple of 5, between 2000 and 2500?

Sol:

```
# Program to find numbers between 2000 and 2500 that are multiples of 17 but not 5
def find_numbers():
    # List to store the numbers
    result = []

    # Check each number in the range 2000 to 2500
    for num in range(2000, 2501):
        # Check if number is a multiple of 17 and not a multiple of 5
        if num % 17 == 0 and num % 5 != 0:
            result.append(num)

    return result

# Call the function and print results
numbers = find_numbers()

# Display the results
print("Numbers between 2000 and 2500 that are multiples of 17 but not 5:")
print(numbers)
print(f"Total count: {len(numbers)}")
```

Output

Numbers between 2000 and 2500 that are multiples of 17 but not 5:

[2006, 2023, 2040, 2057, 2074, 2091, 2108, 2125, 2142, 2159, 2176, 2193, 2210, 2227, 2244, 2261, 2278, 2295, 2312, 2329, 2346, 2363, 2380, 2397, 2414, 2431, 2448, 2465, 2482, 2499]

Total count: 30

Explanation

- Function find_numbers():
  - Creates an empty list result to store numbers that meet the criteria.
  - Iterates through the range from 2000 to 2500 (inclusive, so range(2000, 2501)).
  - Checks if a number is a multiple of 17 (num % 17 == 0) and not a multiple of 5 (num % 5 != 0).
  - If both conditions are met, the number is added to the result list.
  - Returns the list of valid numbers.

No. 4 Swap two integer numbers using a temporary variable. Repeat the exercise using the code format: a, b = b, a. Verify your results in both the cases.

```
# Swap using a temporary variable
a = int(input("Enter first number (a): "))
b = int(input("Enter second number (b): "))

print("\nBefore swapping:")
print("a =", a, "b =", b)

# Swapping
temp = a
a = b
b = temp

print("\nAfter swapping using temp variable:")
print("a =", a, "b =", b)
```

Output:

a = 10
b = 5
Before swapping:
a = 10 b = 5

After swapping:
a = 5 b = 10

No. 5 Find the largest of n numbers, using a user defined function largest().

Sol:

```
def largest(numbers):
    """
    Function to find the largest number in a list
    """
    max_num = numbers[0]  # Assume first number is the largest
    for num in numbers[1:]:
        if num > max_num:
            max_num = num
    return max_num

# Main program
n = int(input("Enter how many numbers: "))
nums = []

print("Enter", n, "numbers:")
for i in range(n):
    number = float(input(f"Number {i+1}: "))
    nums.append(number)

# Call the function
max_value = largest(nums)

# Display result
print("\nThe largest number is:", max_value)
```

Sample Output:

```
Enter how many numbers: 5
Enter 5 numbers:
Number 1: 21
Number 2: 17
Number 3: 88
Number 4: 45
Number 5: 9

The largest number is: 88.0
```

No. 6 Write a function my Reverse() which receives a string as an input and returns the reverse of the string.

```
def myReverse(s):
    """
    This function takes a string s and returns the reversed string.
    """
    return s[::-1]  # String slicing to reverse

# Main program to test the function
text = input("Enter a string: ")
reversed_text = myReverse(text)

print("Reversed string:", reversed_text)
```

Output:

Enter a string: Hello World
Reversed string: dlroW olleH

Explanation:

- *s[::-1] is a Python slicing technique that returns the string in reverse order.*
- *You can also implement it manually using a loop if slicing is not allowed (for educational purpose):*

Alternative: (without slicing):

```
def myReverse(s):
    reversed_str = ''
    for char in s:
        reversed_str = char + reversed_str  # Add each character to the front
    return reversed_str
```

No. 7 Check if a given string is palindrome or not.

Sol:

```
def is_palindrome(s):
    """
    This function checks if a string is a palindrome.
    Returns True if palindrome, else False.
    """
    s = s.lower().replace(" ", "")  # Optional: ignore case and spaces
    return s == s[::-1]  # Compare string with its reverse

# Main program
text = input("Enter a string: ")

if is_palindrome(text):
    print("The string is a palindrome.")
else:
    print("The string is NOT a palindrome.")
```

Output:

Enter a string: Madam
The string is a palindrome.

Enter a string: Hello
The string is NOT a palindrome.

Explanation:

- s[::-1] reverses the string.
- s.lower() makes the comparison case-insensitive.
- replace(" ", "") ignores spaces (optional, can be removed).

No. 8 WAP to convert Celsius to Fahrenheit

```
def celsius_to_fahrenheit(celsius):
    fahrenheit = (celsius * 9/5) + 32
    return fahrenheit

# Main program
c = float(input("Enter temperature in Celsius: "))
f = celsius_to_fahrenheit(c)

print(f"{c}°C = {f:.2f}°F")
```

Output:

```
Enter temperature in Celsius: 25
25.0°C = 77.00°F
```

No.9 Find the ASCII value of charades

Sol:

```
# Input a character
char = input("Enter a character: ")

# Check if exactly one character is entered
if len(char) == 1:
    ascii_value = ord(char)  # Get ASCII using ord()
    print(f"The ASCII value of '{char}' is {ascii_value}")
else:
    print("Please enter a single character only.")
```

Output:

```
Enter a character: A
The ASCII value of 'A' is 65

Enter a character: $
The ASCII value of '$' is 36
```

Explanation:

- ord(character) → Returns the ASCII (or Unicode) value of the character.
- chr(number) → Converts ASCII number back to character (opposite of ord()).

No. 10 WAP for simple calculator
Sol:

```
def add(a, b):
    return a + b

def subtract(a, b):
    return a - b

def multiply(a, b):
    return a * b

def divide(a, b):
    if b != 0:
        return a / b
    else:
        return "Error: Division by zero"

# Main program
print("Simple Calculator")
print("Select operation:")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")

choice = input("Enter choice (1/2/3/4): ")

num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))

if choice == '1':
    print("Result:", add(num1, num2))
elif choice == '2':
    print("Result:", subtract(num1, num2))
elif choice == '3':
    print("Result:", multiply(num1, num2))
elif choice == '4':
    print("Result:", divide(num1, num2))
else:
    print("Invalid input")
```

Output:

Simple Calculator
Select operation:
1. Add
2. Subtract
3. Multiply
4. Divide
Enter choice (1/2/3/4): 3
Enter first number: 5
Enter second number: 4
Result: 20.0

Explanation:

- Uses **functions** to organize operations.
- Handles **division by zero** error safely.
- Uses if-elif to process user choice.