#SMART INTERS-APSCHE

#AI/ML Training

#Assingment-1

```python
#1.Write a Python program to calculate the area of a rectangle given its
length and width.
def calculate_rectangle_area(length, width):
    return length * width

def main():
    length = float(input("Enter the length of the rectangle: "))
    width = float(input("Enter the width of the rectangle: "))

    area = calculate_rectangle_area(length, width)
    print("The area of the rectangle is:", area)

if_name_== "_main__":
    main()
```

output

```
Enter the length of the rectangle: 12
Enter the width of the rectangle: 34
The area of the rectangle is: 408.0
```

```python
#2.Write a program to convert miles to kilometers
def miles_to_kilometers(miles):
    kilometers = miles * 1.60934
    return kilometers

def main():
    miles = float(input("Enter the distance in miles: "))
    kilometers = miles_to_kilometers(miles)
    print(f"{miles} miles is equal to {kilometers} kilometers.")

if_name_== "_main__":
    main()
    for num in numbers:
        if num > largest:
            second_largest = largest
            largest = num
        elif num > second_large
```

OUTPUT

```
Enter the distance in miles: 6
6.0 miles is equal to 9.65604 kilometre’s.
```

```python
#3. Write a function to check if a given string is a palindrome
def is_palindrome(s):
    # Convert the string to lowercase and remove non-alphanumeric characters
    s = ''.join(char.lower() for char in s if char.isalnum())

    # Check if the string is equal to its reverse
    return s == s[::-1]

# Test the function
def main():
    test_string = input("Enter a string: ")
    if is_palindrome(test_string):
        print("The string is a palindrome.")
    else:
        print("The string is not a palindrome.")

if__name__== "__main__":
```

OUTPUT

```
Enter a string: palindrome
The string is not a palindrome.
```

```python
#4. Write a Python program to find the second largest element in a list
def second_largest(numbers):
    if len(numbers) < 2:
        return "List must have at least two elements"

    largest = float('-inf') # Initialize largest to negative infinity
    second_largest = float('-inf')  # Initialize second largest to negative
infinity
    st and num != largest:
            second_largest = num

    if second_largest == float('-inf'):
        return "There is no second largest element"
    else:
        return second_largest

# Test the function
def main():
    nums = [int(x) for x in input("Enter elements of the list separated by
space: ").split()]
    result = second_largest(nums)
    print("The second largest element in the list is:", result)

if_name_== "_main__":
    main()
```

OUTPUT

```
Enter elements of the list separated by space: 2 4 5
The second largest element in the list is: 4
```

```python
#5. Explain what indentation means in Python
x=10
if x > 5:
    print("x is greater than 5")
    print("This line is also part of the if block")

print("This line is not indented, so it's not part of the if block")
```

OUTPUT

```
x is greater than 5
This line is also part of the if block
This line is not indented, so it's not part of the if block
```

```python
#6. Write a program to perform set difference operation
def set_difference_using_operator(set1, set2):
    return set1 - set2

def set_difference_using_method(set1, set2):
    return set1.difference(set2)

# Test the functions
def main():
    set1 = {1, 2, 3, 4, 5}
    set2 = {3, 4, 5, 6, 7}

    difference_operator = set_difference_using_operator(set1, set2)
    difference_method = set_difference_using_method(set1, set2)

    print("Set difference using operator:", difference_operator)
    print("Set difference using method:", difference_method)

if_name_== "_main__":
    main()
```

OUTPUT

```
Set difference using operator: {1, 2}
Set difference using method: {1, 2}
```

```python
#7. Write a Python program to print numbers from 1 to 10 using a while loop
def print_numbers():
    num = 1
    while num <= 10:
        print(num)
        num += 1

# Test the function
def main():
    print("Numbers from 1 to 10:")
    print_numbers()

if_name_== "_main__":
    main()
```

OUTPUT

```
Numbers from 1 to 10:
1
2
3
4
5
6
7
8
9
10
```

```python
#8.Write a program to calculate the factorial of a number using a while loop
def factorial(n):
    if n < 0:
        return "Factorial is not defined for negative numbers"
    elif n == 0:
        return 1
    else:
        result = 1
        while n > 0:
            result *= n
            n -= 1
        return result

# Test the function
def main():
    num = int(input("Enter a number to calculate its factorial: "))
    print("Factorial of", num, "is", factorial(num))

if_name_== "_main__":
    main()
```

OUTPUT

```
Enter a number to calculate its factorial: 10
Factorial of 10 is 3628800
```

```python
#9. Write a Python program to check if a number is positive, negative, or zero using if-elif-else
def check_number(num):
    if num > 0:
        print("The number is positive.")
    elif num < 0:
        print("The number is negative.")
    else:
        print("The number is zero.")

# Test the function
def main():
    num = float(input("Enter a number: "))
    check_number(num)

if_name_== "_main__":
    main()
```

OUTPUT

```
Enter a number: 12
The number is positive.
```

```
#10. Write a program to determine the largest among three numbers using
conditional
def find_largest(num1, num2, num3):
    if num1 >= num2 and num1 >= num3:
        return num1
    elif num2 >= num1 and num2 >= num3:
        return num2
    else:
        return num3

# Test the function
def main():
    num1 = float(input("Enter the first number: "))
    num2 = float(input("Enter the second number: "))
    num3 = float(input("Enter the third number: "))

    largest = find_largest(num1, num2, num3)
    print("The largest number among", num1, ",", num2, ", and", num3, "is",
largest)

if_name_== "_main__":
    main()
```

OUTPUT

```
Enter the first number: 12
Enter the second number: 23
Enter the third number: 45
The largest number among 12.0 , 23.0 , and 45.0 is 45.0
```

```python
#11. Write a Python program to create a numpy array filled with ones of given shape
import numpy as np

def create_ones_array(shape):
    return np.ones(shape)

# Test the function
def main():
    shape = tuple(map(int, input("Enter the shape of the array (separated by spaces): ").split()))
    ones_array = create_ones_array(shape)
    print("Array of ones with shape", shape, ":\n", ones_array)

if_name_== "_main__":
    main()
```

```python
#12. Write a program to create a 2D numpy array initialized with random
integers
import numpy as np

def create_random_array(rows, cols):
    return np.random.randint(0, 100, size=(rows, cols))

# Test the function
def main():
    rows = int(input("Enter the number of rows: "))
    cols = int(input("Enter the number of columns: "))
    random_array = create_random_array(rows, cols)
    print("Random array:\n", random_array)

if_name_== "_main__":
    main()
```

13.Write a Python program to generate an array of evenly spaced numbers over a specified
#range using linspace

```
import numpy as np

def generate_evenly_spaced(start, stop, num):
    return np.linspace(start, stop, num)

# Test the function
def main():
    start = float(input("Enter the start value: "))
    stop = float(input("Enter the stop value: "))
    num = int(input("Enter the number of elements: "))
    evenly_spaced_array = generate_evenly_spaced(start, stop, num)
    print("Array of evenly spaced numbers:\n", evenly_spaced_array)

if_name_== "_main__":
    main()
```