

## ASSIGNMENT 1

1. Write a Python program to calculate the area of a rectangle given its length and width.

### PROGRAM

```
def calculate_rectangle_area(length, width):  
    return length * width  
  
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)
```

### Output

```
Enter the length of the rectangle: 5  
Enter the width of the rectangle: 4  
The area of the rectangle is: 20.0  
> |
```

2. Write a program to convert miles to kilometers

### PROGRAM

```
def miles_to_kilometers(miles):  
    return miles * 1.60934  
  
miles = float(input("Enter the distance in miles: "))  
kilometers = miles_to_kilometers(miles)  
print(f'{miles} miles is equal to {kilometers:.2f} kilometers.")
```

### Output

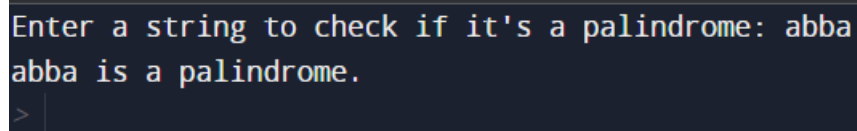
```
Enter the distance in miles: 500  
500.0 miles is equal to 804.67 kilometers.  
> |
```

3. Write a function to check if a given string is a palindrome.

**PROGRAM**

```
def is_palindrome(s):  
    s = s.replace(" ", "").lower()  
    return s == s[::-1]  
  
input_string = input("Enter a string to check if it's a palindrome: ")  
  
if is_palindrome(input_string):  
    print(f'{input_string} is a palindrome.')  
else:  
    print(f'{input_string} is not a palindrome.')
```

**Output**



```
Enter a string to check if it's a palindrome: abba  
abba is a palindrome.  
> |
```

4. Write a Python program to find the second largest element in a list.

**PROGRAM**

```
def find_second_largest(nums):  
    if len(nums) < 2:  
        return "List should have at least two elements."  
  
    largest = second_largest = float('-inf')  
  
    for num in nums:  
        if num > largest:  
            second_largest = largest  
            largest = num  
        elif num > second_largest and num != largest:  
            second_largest = num
```

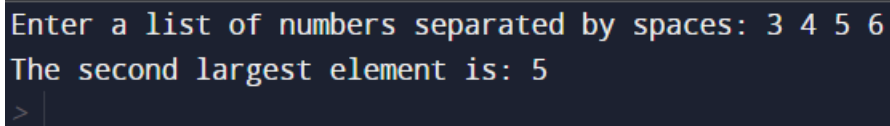
```

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

input_list = [int(x) for x in input("Enter a list of numbers separated by spaces: ").split()]
result = find_second_largest(input_list)
print("The second largest element is:", result)

```

### Output



```

Enter a list of numbers separated by spaces: 3 4 5 6
The second largest element is: 5
>

```

### 5. Explain what indentation means in Python.

In Python, indentation is a fundamental aspect of the syntax and serves to define code blocks. It indicates the scope of statements within loops, conditionals, functions, and classes. Proper indentation enhances code readability, and the consistent use of spaces or tabs (but not both) is crucial for Python to interpret the structure of the code correctly. Incorrect indentation may result in syntax errors or unintended program behavior.

### 6. Write a program to perform set difference operation.

#### PROGRAM

```

def set_difference(set1, set2):
    return set1 - set2

set_a = {1, 2, 3, 4, 5}
set_b = {3, 4, 5, 6, 7}
result = set_difference(set_a, set_b)
print("Set A:", set_a)
print("Set B:", set_b)
print("Set Difference (A - B):", result)

```

### Output

```
Set A: {1, 2, 3, 4, 5}
Set B: {3, 4, 5, 6, 7}
Set Difference (A - B): {1, 2}
>
```

7. Write a Python program to print numbers from 1 to 10 using a while loop.

#### PROGRAM

```
number = 1
while number <= 10:
    print(number)
    number += 1
```

#### Output

```
1
2
3
4
5
6
7
8
9
10
>
```

8. Write a program to calculate the factorial of a number using a while loop.

#### PROGRAM

```
def calculate_factorial(n):
    factorial = 1
    while n > 1:
        factorial *= n
        n -= 1
    return factorial

number = int(input("Enter a number to calculate its factorial: "))
result = calculate_factorial(number)
```

```
print(f"The factorial of {number} is: {result}")
```

### Output

```
Enter a number to calculate its factorial: 25
The factorial of 25 is: 15511210043330985984000000
> |
```

9. Write a Python program to check if a number is positive, negative, or zero using if-elif-else statements.

### PROGRAM

```
number = float(input("Enter a number: "))
if number > 0:
    print("The number is positive.")
elif number < 0:
    print("The number is negative.")
else:
    print("The number is zero.")
```

### Output

```
Enter a number: 5
The number is positive.
> |
```

10. Write a program to determine the largest among three numbers using conditional statements.

### PROGRAM

```
num1 = float(input("Enter the first number: "))
num2 = float(input("Enter the second number: "))
num3 = float(input("Enter the third number: "))
if num1 >= num2 and num1 >= num3:
    largest = num1
elif num2 >= num1 and num2 >= num3:
```

```

    largest = num2
else:
    largest = num3
print(f"The largest number among {num1}, {num2}, and {num3} is: {largest}")

```

### Output

```

Enter the first number: 34
Enter the second number: 56
Enter the third number: 23
The largest number among 34.0, 56.0, and 23.0 is: 56.0
>

```

11. Write a Python program to create a numpy array filled with ones of given shape.

### PROGRAM

```

import numpy as np

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

rows = int(input("Enter the number of rows: "))
columns = int(input("Enter the number of columns: "))

array_shape = (rows, columns)
result_array = create_ones_array(array_shape)
print(f"NumPy array of shape {array_shape} filled with ones:\n{result_array}")

```

### Output

```

Enter the number of rows: 4
Enter the number of columns: 5
NumPy array of shape (4, 5) filled with ones:
[[1.  1.  1.  1.  1.]
 [1.  1.  1.  1.  1.]
 [1.  1.  1.  1.  1.]
 [1.  1.  1.  1.  1.]]
>

```

12. Write a program to create a 2D numpy array initialized with random integers.

## PROGRAM

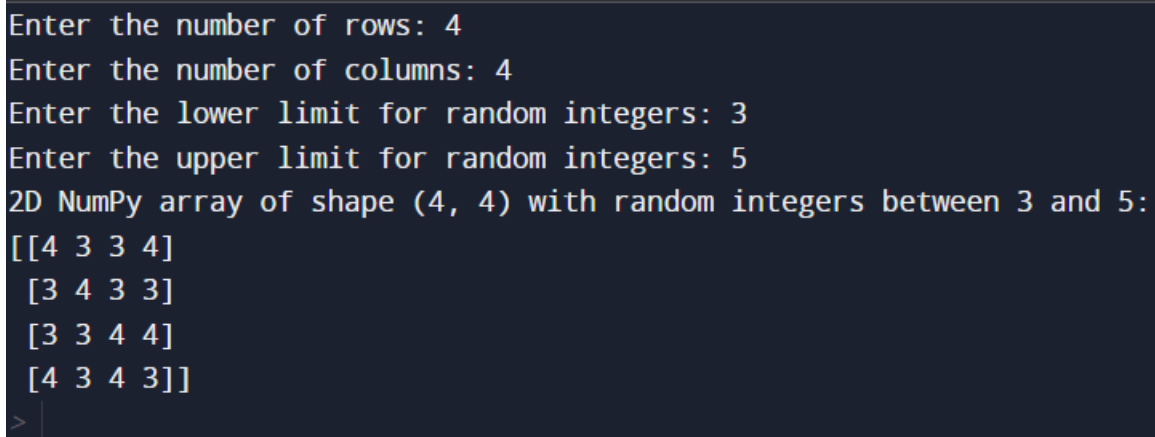
```
import numpy as np

def create_random_integers_array(rows, columns, low, high):
    random_array = np.random.randint(low, high, size=(rows, columns))
    return random_array

rows = int(input("Enter the number of rows: "))
columns = int(input("Enter the number of columns: "))
low_limit = int(input("Enter the lower limit for random integers: "))
high_limit = int(input("Enter the upper limit for random integers: "))
result_array = create_random_integers_array(rows, columns, low_limit,
high_limit)

print(f"2D NumPy array of shape ({rows}, {columns}) with random integers
between {low_limit} and {high_limit}:\n{result_array}")
```

## Output



```
Enter the number of rows: 4
Enter the number of columns: 4
Enter the lower limit for random integers: 3
Enter the upper limit for random integers: 5
2D NumPy array of shape (4, 4) with random integers between 3 and 5:
[[4 3 3 4]
 [3 4 3 3]
 [3 3 4 4]
 [4 3 4 3]]
> |
```

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

## PROGRAM

```
import numpy as np

def generate_linspace_array(start, end, num_points):
    linspace_array = np.linspace(start, end, num_points)
    return linspace_array
```

```

start_value = float(input("Enter the start value of the range: "))
end_value = float(input("Enter the end value of the range: "))
num_points = int(input("Enter the number of evenly spaced points: "))
result_array = generate_linspace_array(start_value, end_value, num_points)
print(f"NumPy array of {num_points} evenly spaced numbers between
{start_value} and {end_value}:\n{result_array}")

```

14. Write a program to generate an array of 10 equally spaced values between 1 and 100 using linspace.

#### PROGRAM

```

import numpy as np
result_array = np.linspace(1, 100, 10)
print("NumPy array of 10 equally spaced values between 1 and 100:")
print(result_array)

```

#### Output

```

NumPy array of 10 equally spaced values between 1 and 100:
[  1.  12.  23.  34.  45.  56.  67.  78.  89. 100.]
>

```

15. Write a Python program to create an array containing even numbers from 2 to 20 using arange.

#### PROGRAM

```

import numpy as np
result_array = np.arange(1, 10.5, 0.5)
print("NumPy array containing numbers from 1 to 10 with a step size of 0.5:")
print(result_array)

```

#### Output

```

NumPy array containing numbers from 1 to 10 with a step size of 0.5:
[ 1.  1.5  2.  2.5  3.  3.5  4.  4.5  5.  5.5  6.  6.5  7.  7.5
 8.  8.5  9.  9.5 10.]
>

```



16. Write a program to create an array containing numbers from 1 to 10 with a step size of 0.5 using arange

**PROGRAM**

```
import numpy as np
result_array = np.arange(1, 10.5, 0.5)
print("NumPy array containing numbers from 1 to 10 with a step size of 0.5:")
print(result_array)
```

**Output**

```
NumPy array containing numbers from 1 to 10 with a step size of 0.5:
[ 1.  1.5  2.  2.5  3.  3.5  4.  4.5  5.  5.5  6.  6.5  7.  7.5
 8.  8.5  9.  9.5 10.]
> |
```