**Assignment - 1**

**SMART INTERNZ - APSCHE**

**AI / ML Training**:

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

**A)** # Write a Program to find the Area of a Rectangle

w = float(input('Please Enter the Width of a Rectangle: '))

h = float(input('Please Enter the Height of a Rectangle: '))

# calculate the area

Area = w \* h

# calculate the Perimeter

Perimeter = 2 \* (w + h)

print("\n Area of a Rectangle is: %.2f" %Area)

Output:

Please Enter the Width of a Rectangle: 10

Please Enter the Height of a Rectangle: 5

Area of a Rectangle is: 50.00

**2.Write a program to convert miles to kilometers?**

**A)**program to convert miles to kilometers.

miles = float(input("Please enter miles:"))

kilometers = miles \* 1.6

print(kilometers, " Kilometers")

Output:

Please enter miles: 22

35.2 Kilometers”

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

**A)**Palindrome:

A palindrome is a sequence of characters that, when reversed, would result in the exact same sequence of characters.

Methods to Check if a String is a Palindrome

Method 1 – Using the reverse and compare methods

#Define a function

def isPalindrome(string):

    if (string == string[::-1]) :

        return "The string is a palindrome."

    else:

        return "The string is not a palindrome."

#Enter input string

string = input ("Enter string: ")

print(isPalindrome(string))

Output:

Enter the string: radar

The string is Palindrome.

Method 2 – Using for loop

#Enter input string

string = input("Enter string : ")

#Declare an empty string variable

revstr = ""

#Iterate string with for loop

for i in string:

    revstr = i + revstr

print("Reversed string : ", revstr)

if(string == revstr):

   print("The string is a palindrome.")

else:

   print("The string is not a palindrome.")

Output:

Enter string: madam

Reversed string: madam

The string is Palindrome.

Method 3 – Using a while loop

#Define a function

def isPalindrome(string):

    string = string.lower().replace(' ', '')

    first, last = 0, len(string) - 1

    while(first < last):

        if(string[first] == string[last]):

            first += 1

            last -= 1

        else:

            return "The string is not a palindrome."

    return "The string is a palindrome."

#Enter input string

str1 = input("Enter string : ")

print(isPalindrome(str1))

#Returns true

Output:

Enter The String:level

The string is palindrome.

Method 4 – Using the reverse and join method

#Define a function

def isPalindrome(string):

    revstr=''.join(reversed(string))

    if string==revstr:

         return "The string is a palindrome."

    return "The string is not a palindrome."

#Enter input string

string = input ("Enter string: ")

print(isPalindrome(string))

Output:

Enter The String: tenet

The String is Palindrome.

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

**A)**Python program to find second largest number in a list.

There are various methods let we know the some of the methods :

Method 1: Sorting is an easier but less optimal method. Given below is an O(n) algorithm to do the same.

# Python program to find second largest

# number in a list

# list of numbers - length of

# list should be at least 2

list1 = [10, 20, 4, 45, 99]

mx = max(list1[0], list1[1])

secondmax = min(list1[0], list1[1])

n = len(list1)

for i in range(2,n):

if list1[i] > mx:

secondmax = mx

mx = list1[i]

elif list1[i] > secondmax and \

mx != list1[i]:

secondmax = list1[i]

elif mx == secondmax and \

secondmax != list1[i]:

secondmax = list1[i]

print("Second highest number is : ",\

str(secondmax)

Output:

Second highest number is : 45

Method 2: Sort the list in ascending order and print the second last element in the list.

# Python program to find largest number

# in a list

# List of numbers

list1 = [10, 20, 20, 4, 45, 45, 45, 99, 99]

# Removing duplicates from the list

list2 = list(set(list1))

# Sorting the list

list2.sort()

# Printing the second last element

print("Second largest element is:", list2[-2])

Output:

Second largest element is: 45

Time Complexity: O(nlogn)

Auxiliary Space: O(1)

Method 3: By removing the max element from the list

# Python program to find second largest number

# in a list

# List of numbers

list1 = [10, 20, 4, 45, 99]

# new\_list is a set of list1

new\_list = set(list1)

# Removing the largest element from temp list

new\_list.remove(max(new\_list))

# Elements in original list are not changed

# print(list1)

print(max(new\_list))

Output:

45.

Time Complexity: O(n)

Auxiliary Space: O(1)

Method 4: Find the max list element on inputs provided by the user

# Python program to find second largest

# number in a list

# creating list of integer type

list1 = [10, 20, 4, 45, 99]

'''

# sort the list

list1.sort()

# print second maximum element

print("Second largest element is:", list1[-2])

'''

# print second maximum element using sorted() method

print("Second largest element is:", sorted(list1)[-2])

Outputs:

Second largest element is: 45

Time Complexity: O(NlogN) where N is the number of elements in the Array.

Auxiliary Space: O(1)

Method 5: Traverse once to find the largest and then once again to find the second largest.

def findLargest(arr):

secondLargest = 0

largest = min(arr)

for i in range(len(arr)):

if arr[i] > largest:

secondLargest = largest

largest = arr[i]

else:

secondLargest = max(secondLargest, arr[i])

# Returning second largest element

return secondLargest

# Calling above method over this array set

print(findLargest([10, 20, 4, 45, 99]))

Output:

45

Time Complexity: O(N)

Auxiliary Space: O(1)

**5.Explain what indentation means in Python?**

**A)**

Python Indentations:

Indentation refers to the spaces at the beginning of a code line.

Where in other programming languages the indentation in code is for readability only, the indentation in Python is very important.

Python uses indentation to indicate a block of code.

if 5 > 2:

print("Five is greater than two!")

Output:

Five is greater than two!

Python will give you an error if you skip the indentation:

Example

Syntax Error:

if 5 > 2:

print("Five is greater than two!")

Output:

File "demo\_indentation\_test.py", line 2

print("Five is greater than two!")

^

IndentationError: expected an indented block

You have to use the same number of spaces in the same block of code, otherwise Python will give you an error:

Example

Syntax Error:

if 5 > 2:

print("Five is greater than two!")

print("Five is greater than two!")

Output:

File "demo\_indentation2\_error.py", line 3

print("Five is greater than two!")

^

IndentationError: unexpected IndentationError

**6.Write a program to perform set difference operation?**

**A)**

Python offers a datatype called set whose elements must be unique. It can be used to perform different set operations like union, intersection, difference and symmetric difference.

# Program to perform different set operations

# define three sets

E = {0, 2, 4, 6, 8};

N = {1, 2, 3, 4, 5};

# set union

print("Union of E and N is",E | N)

# set intersection

print("Intersection of E and N is",E & N)

# set difference

print("Difference of E and N is",E - N)

# set symmetric difference

print("Symmetric difference of E and N is",E ^ N)

Output:

Union of E and N is {0, 1, 2, 3, 4, 5, 6, 8}

Intersection of E and N is {2, 4}

Difference of E and N is {8, 0, 6}

Symmetric difference of E and N is {0, 1, 3, 5, 6, 8}

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

**A)**Python program to print numbers from 1 to 10 using while loop:

# Python program to print numbers from 1 to 10

print('Numbers from 1 to 10:')

number = 1

while True:

if number > 10:

break

print(number)

number += 1

Output:

Numbers from 1 to 10:

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?**

**A)**C Program to Find Factorial of a Number using While Loop:

#include <stdio.h>

int main()

{

int n,i,f;

f=i=1;

printf("Enter a Number to Find Factorial: ");

scanf("%d",&n);

while(i<=n)

{

f\*=i;

i++;

}

printf("The Factorial of %d is : %d",n,f);

return 0;

}

Output:

Enter a Number to Find Factorial: 5

The Factorial of 5 is : 120

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

**statements.**

**A)** Python program to check if a number is positive, negative, or zero using if-elif-else

statements.

# Prompt the user to input a number, and convert the input to a floating-point number.

num = float(input("Input a number: "))

# Check if the number is greater than zero.

if num > 0:

# If true, print that it is a positive number.

print("It is a positive number")

# Check if the number is equal to zero.

elif num == 0:

# If true, print that it is zero.

print("It is zero")

else:

# If the above conditions are not met, print that it is a negative number.

print("It is a negative number")

Output:

Input a number: 150

It is positive number

# Prompt the user to input a number and convert the input to a floating-point number.

n = float(input('Input a number: '))

# Use a conditional expression (ternary operator) to determine if the number is positive, zero, or negative.

# The conditional expression checks if the number is greater than 0, equal to 0, or less than 0.

# Depending on the condition met, it prints a corresponding message.

print('Number is Positive.' if n > 0 else 'It is Zero!' if n == 0 else 'Number is Negative.')

Output:

Input a number: 0

It is Zero!

# Prompt the user to input a number and convert the input to a floating-point number.

n = float(input("Input a number: "))

# Check if the input number is greater than or equal to 0.

if n >= 0:

# If the number is zero, print that it is zero.

if n == 0:

print("It is Zero!")

# If the number is greater than zero, print that it is a positive number.

else:

print("Number is Positive number.")

# If the number is less than zero, print that it is a negative number.

else:

print("Number is Negative number.")

Output:

Input a number: -150

Number is Negative number.

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

**statements.**

**A)**program to determine the largest among three numbers using conditional

statements.

#include<stdio.h>

int main(){

int a,b,c;

int big;

printf("Enter any there numbers: ");

scanf("%d%d%d",&a,&b,&c);

if(a>b && a>c)

big = a;

else if(b>c)

big = b;

else

big = c;

printf("Largest number is: %d",big);

return 0;

}

output:

Enter any three numbers: 13

25

6

Largest number is: 25

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

**A)**import numpy as np

import numpy.ma as ma

# To return a new array of given shape and type, filled with ones, use the numpy.ones() method in Python Numpy

# The 1st parameter sets the shape of the new array

arr = np.ones((4))

# Displaying our array

print("Array...",arr)

# Get the datatype

print("Array datatype...",arr.dtype)

# Get the dimensions of the Array

print("Array Dimensions...",arr.ndim)

# Get the shape of the Array

print("Our Array Shape...",arr.shape)

# Get the number of elements of the Array

print("Elements in the Array...",arr.size)

Output:

Array...

[1. 1. 1. 1.]

Array datatype...

float64

Array Dimensions...

1

Our Array Shape...

(4,)

Elements in the Array...

4

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

**A)**program to create a 2D numpy array initialized with random integers

import numpy as np

Data = np.array([[1, 2, 3, 4], [5, 6, 7, 8]])

print("The array is:\n", Data)

print(type(Data))

print(np.ndim(Data))

print(Data.size)

print(Data.shape)

Output:

The array is:

[[1 2 3 4]

[5 6 7 8]]

<class 'numpy.ndarray'>

2

8

(2, 4)

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

**range using linspace.?**

**A)**Python program to generate an array of evenly spaced numbers over a specified

range using linspace.

import numpy as np

linear\_space = np.linspace(0, 1, num=10)

print(linear\_space)

Output:

[0. 0.11111111 0.22222222 0.33333333 0.44444444 0.55555556

0.66666667 0.77777778 0.88888889 1. ]

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

**linspace.?**

**A)**program to generate an array of 10 equally spaced values between 1 and 100 using

linspace

import numpy as np

# Generate an array of 10 equally spaced values between 1 and 100

result\_array = np.linspace(1, 100, 10)

# Print the generated array

print("Generated Array:", result\_array)

Output:

Generated Array:

[ 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?**

**A)**Python program to create an array containing even numbers from 2 to 20 using

arange.

import numpy as np

# Create an array containing even numbers from 2 to 20

even\_array = np.arange(2, 21, 2)

# Output the array

print(even\_array)

Output:

[ 2 4 6 8 10 12 14 16 18 20]

This program uses np.arange(start, stop, step) to generate an array starting from 2, ending at 20 (exclusive), with a step size of 2, which results in an array of even numbers.

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

**using arange?**

**A)**program to create an array containing numbers from 1 to 10 with a step size of 0.5

using arange.

import numpy as np

# Create an array containing numbers from 1 to 10 with a step size of 0.5

array\_with\_step = np.arange(1, 10.5, 0.5)

# Output the array

print(array\_with\_step)

Output:

[ 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. ]

This program uses np.arange(start, stop, step) to generate an array starting from 1, ending at 10.5 (exclusive), with a step size of 0.5.