
Name: Hammad Rafique
Reg no: FA20-BSE-029
Subject: Artifical Intelligence

Activity 1:

Use loops to accept 5 values from user and store them in a list. Display all the values (objects) of the list.

In [18]:
```
mylist = []
for i in range(5):
    val = input("Enter your value:")
    mylist.append(val)
print("The given value in list is: ")
print(mylist)
```

```
Enter your value:1
Enter your value:2
Enter your value:3
Enter your value:4
Enter your value:5
The given value in list is:
['1', '2', '3', '4', '5']
```

Activity 2:

Repeat the above code by accepting 5 integer values from user. Store these values in a list and display the sum of given values.

In [6]:
```
mylist = []
sum = 0
for i in range(5):
    mylist.append(int(input("Enter your value: ")))
    sum = sum + mylist[i]
print(sum)
```

```
Enter your value: 2
Enter your value: 2
Enter your value: 2
Enter your value: 2
Enter your value: 2
10
```

Experiment:

Accept 5 int values from user. Store in list, get value to find duplicate elem and print index if found that element from list.

In [9]:
```
mylist = []
result = 0
for i in range(5):
    mylist.append(int(input("Enter your value: ")))

val = int(input("Enter you value to find: "))
for i in range(0, len(mylist)):
    if mylist[i] == val:
        print("\nValue is found at index: " + str(i))
        result = i
        break
    else:
        result = -1

if result == -1:
    print("Value not found")
```

```
Enter your value: 1
Enter your value: 2
Enter your value: 3
Enter your value: 4
Enter your value: 5
Enter you value to find: 5

Value is found at index: 4
```

Activity 3:

Accept 5 integer values from user. Store these values in a list and display the list in ascending order.

In [11]:
```
mylist = []
for i in range(5):
    mylist.append(int(input("Enter your value: ")))
mylist.sort()
print(mylist)
```

```
Enter your value: 1
Enter your value: 2
Enter your value: 3
Enter your value: 4
Enter your value: 5
[1, 2, 3, 4, 5]
```

Activity 4:

Accept two lists from user and display their join.

In [1]:
```
list1 = []
list2 = []
for i in range(3):
    list1.append(input("Enter value for 1 list: "))

for i in range(3):
    list2.append(input("Enter value for 2 list: "))
print("Combination of both list: ", list1+list2)
```

```
Enter value for 1 list: 1
Enter value for 1 list: 2
Enter value for 1 list: 3
Enter value for 2 list: 4
Enter value for 2 list: 5
Enter value for 2 list: 6
Combination of both list:  ['1', '2', '3', '4', '5', '6']
```

Activity 5:

Write a Python code to accept a list from user and find a required element in it.

In [12]:
```
mylist = []
for i in range(5):
    mylist.append(int(input("Enter your value: ")))

val = int(input("Enter you value to find: "))
if val in mylist:
    print("\nElement has been found!")
else:
    print("\nElement not found")
```

```
Enter your value: 1
Enter your value: 2
Enter your value: 3
Enter your value: 4
Enter your value: 5
Enter you value to find: 5

Element has been found!
```

Activity 6:

Write a function called say_hello that takes in a person's name as a parameter. The function should print a greeting message with the person's name. Then call the function three times with three different names.

In [17]:
```
def name_generator(name):
    print("Welcome! " + name + ", We're glad to see you\n")

for i in range(3):
    name_generator(input("Enter your name: "))
```

```
Enter your name: Hammad Rafique
Welcome! Hammad Rafique, We're glad to see you

Enter your name: M Nouman
Welcome! M Nouman, We're glad to see you

Enter your name: Umar Waseem
Welcome! Umar Waseem, We're glad to see you
```

Activity 7:

A palindrome is a string which is same read forward or backwards. For example: "dad" is the same in forward or reverse direction. Another example is "aibohphobia" which literally means, an irritable fear of palindromes.

Write a function in python that receives a string and returns True if that string is a palindrome and False otherwise. Remember that difference between upper and lower case characters are ignored during this determination.

In [12]:
```
def palindron_verify_method1(value):
    reverse = value[::-1]
    if reverse == value:
        print("Given string is palindron!")
    else:
        print("Given string is not palindron!")

value = input("Enter your string: ")
palindron_verify_method1(value.upper())
```

```
Enter your string: dad
Given string is palindron! DAD
```

Activity 8:

Imagine two matrices given in the form of 2D lists as under;

a = [[1, 0, 0], [0, 1, 0], [0, 0, 1]]

b = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]

Write a python code that finds another matrix/2D list that is a product of and b, i.e.,

In [1]:
```
a = [[1, 0, 0], [0, 1, 0], [0, 0, 1] ]
b = [[1, 2, 3], [4, 5, 6], [7, 8, 9] ]
c = []

for row in range(3):
    c.append([])
    for col in range(3):
        c[row].append(0)
        for ele in range(3):
            c[row][col] += a[row][ele] * b[col][ele]
print(c)
```

```
[[1, 4, 7], [2, 5, 8], [3, 6, 9]]
```

Activity 9:

A closed polygon with N sides can be represented as a list of tuples of N connected coordinates,

i.e., [(x1,y1), (x2,y2), (x3,y3), . . . , (xN,yN)]. A sample polygon with 6 sides (N=6) is shown below.

In [2]:
```
def perimeter(lst):
    size = len(lst)
    perimeter = 0
    for i in range(0, size-1):
        dist = (((lst[i][0]-lst[i+1][0])**2) + ((lst[i][1] - lst[i+1][1])**2)) ** 0.5
        perimeter = perimeter + dist
    perimeter = perimeter + (((lst[0][0] - lst[size-1][0])**2) + ((lst[0][1] - lst[size-1][1])**2))**0.5
    return perimeter
L = [(1,3), (2,7),(3,9),(-1,8)]
print(perimeter(L))
```

```
15.867444035869614
```

Alternative Method:

Performing all logic inside for loop. First assigning i to j var than assigning j=-1 after checking condition if j == size-1, because after we reached the end of list we start it from start which is 0 index.

In [29]:
```
def perimeter(lst):
    size = len(lst)
    per = 0
    for i in range(size):
        j = i
        if j == size-1: j = -1
        dst = (lst[i][0] - lst[j+1][0])**2 + (lst[i][1] - lst[j+1][1])**2
        dst = dst ** 0.5
        per = per + dst

    return per
L = [(1,3), (2,7),(3,9),(-1,8)]
print(perimeter(L))
```

```
15.867444035869614
```

Experiment:

In [10]:
```
mylist = []
for i in range(5):
    mylist.append(int(input("Enter your value: ")))
print("\nMax value iin list: " + str(max(mylist)))
```

```
Enter your value: 1
Enter your value: 2
Enter your value: 3
Enter your value: 4
Enter your value: 5

Max value iin list: 5
```