**ASSIGNMENT 6 – Lab6**

**ARGHA MALLICK – 11500122014**

# A1.Generate 100 random numbers between 101 and 200. Count the frequency of numbers in the different ranges

# [101-125,  126-150, 151-175, 176-200]

import random

range\_101\_125 = 0

range\_126\_150 = 0

range\_151\_175 = 0

range\_176\_200 = 0

for \_ in range(100):

    random\_number = random.randint(101, 200)

    if 101 <= random\_number <= 125:

        range\_101\_125 += 1

    elif 126 <= random\_number <= 150:

        range\_126\_150 += 1

    elif 151 <= random\_number <= 175:

        range\_151\_175 += 1

    elif 176 <= random\_number <= 200:

        range\_176\_200 += 1

print("Frequency in [101-125]:", range\_101\_125)

print("Frequency in [126-150]:", range\_126\_150)

print("Frequency in [151-175]:", range\_151\_175)

print("Frequency in [176-200]:", range\_176\_200)

# A2. Write a function to generate a random alphanumeric string with 6 characters. There must be one uppercase, one lower case, one digit in the string and all string should start with an uppercase letter.

# Call the function 100 times and check how many time a digit is available at second position.

import random

import string

def random\_str():

    upper = random.choice(string.ascii\_uppercase)

    lower = random.choice(string.ascii\_lowercase)

    digit = random.choice(string.digits)

    chars = string.ascii\_letters + string.digits

    remaining = ''.join(random.choice(chars) for \_ in range(3))

    str = upper + lower + digit + remaining

    str\_list = list(str)

    random.shuffle(str\_list)

    return ''.join(str\_list)

count\_digits = 0

for \_ in range(100):

    random\_string = random\_str()

    if random\_string[1].isdigit():

        count\_digits = count\_digits+1

print(count\_digits)

# 3. Write a program to reverse a List(without using reverse function)

def reverse(lst):

    rev\_lst = []

    for n in lst:

        rev\_lst.insert(0, n)

    return rev\_lst

print(reverse([1,2,3,4,5]))

# 4. Write a program to sort the alphabets in a string(without using sort function)

def sort(str):

    chars = list(str)

    n = len(chars)

    for i in range(n-1):

        for j in range(n-1-i):

            if chars[j] > chars[j+1]:

                # swap

                chars[j], chars[j+1] = chars[j+1], chars[j]

    sorted\_str = ''.join(chars)

    return sorted\_str

print(sort("ARGHAMALLICK"))

# 5.A) Using function, write a program to input n numbers into a list and arrange the numbers in descending order using Bubble sort technique.

# 5.B) Accept another number(P) to search the sorted list using linear search algorithm. If the search element is present in the list then print “Search successful else print “Search unsuccessful”.

# 6. Using question No. 5(A): Accept another number(P) to search the sorted list using binary search algorithm. If the search element is present in the list then print “Search successful” else print “Search unsuccessful”.

def bubble\_sort\_descending(lst):

    n = len(lst)

    for i in range(n-1):

        for j in range(n-1-i):

            if lst[j] < lst[j+1]:

                lst[j], lst[j+1] = lst[j+1], lst[j]

    return lst

def linear\_search(lst, p):

    for n in lst:

        if n == p:

            print("Search Successful")

            return

    print("Search Unsuccessful")

def binary\_search(sorted\_lst, p):

    first = 0

    last = len(sorted\_lst)-1

    while first <= last:

        mid = (first+last) // 2

        if sorted\_lst[mid] == p:

            print("Search Successful")

            return

        elif sorted\_lst[mid] < p:

            last = mid-1

        else:

            first = mid+1

    print("Search Unsuccesful")

def main():

    lst = []

    n = int(input("Enter the number of elements: "))

    for \_ in range(n):

        lst.append(int(input("Enter number: ")))

    sorted\_lst = bubble\_sort\_descending(lst)

    print("Sorted in Descending Order:", sorted\_lst)

    p = int(input("Enter the element to search (using linear search): "))

    linear\_search(lst, p)

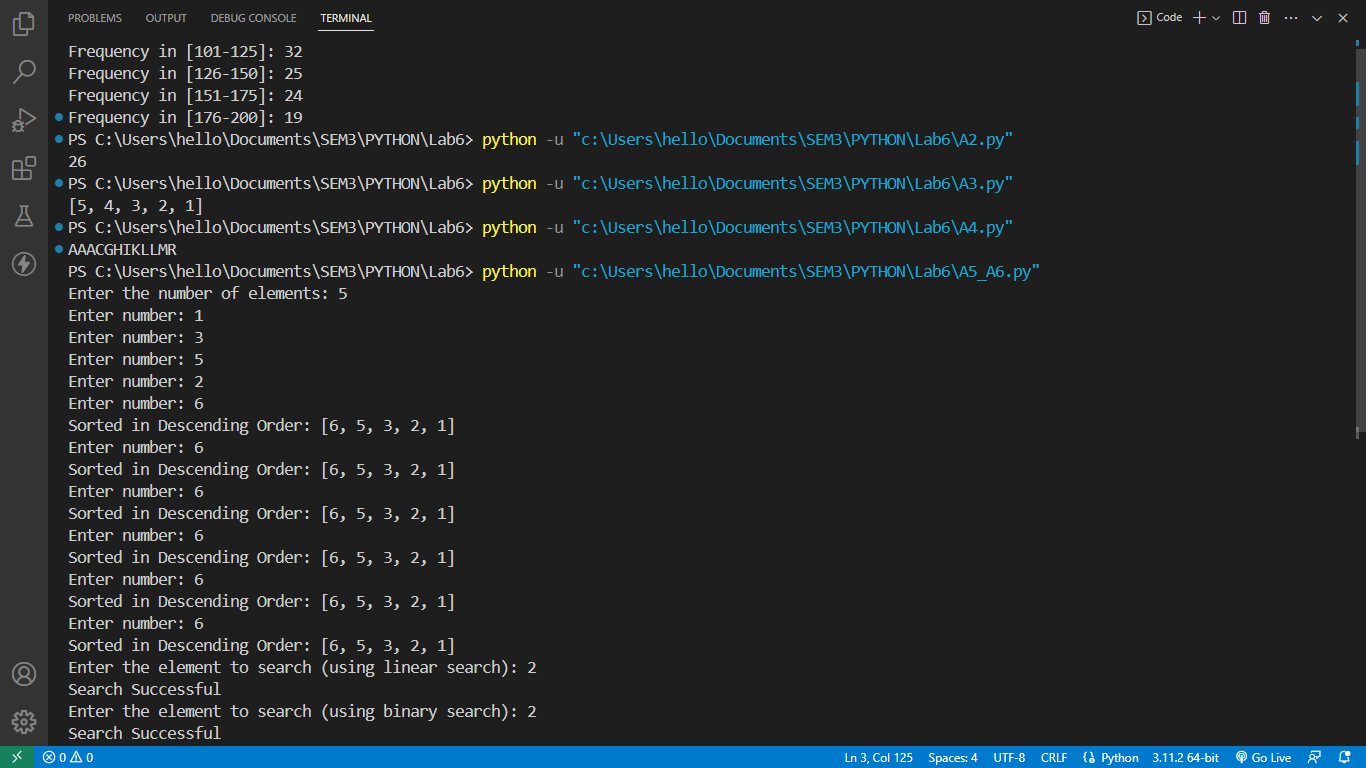
    p = int(input("Enter the element to search (using binary search): "))

    binary\_search(sorted\_lst, p)

if \_\_name\_\_ == "\_\_main\_\_":

    main()

**OUTPUT**

****