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)
```

```python
# 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)
```

```python
# 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]))
```

```python
# 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"))
```

```python
# 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.
```

```python
# 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

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL

Frequency in [101-125]: 32
Frequency in [126-150]: 25
Frequency in [151-175]: 24
Frequency in [176-200]: 19
PS C:\Users\hello\Documents\SEM3\PYTHON\Lab6> python -u "c:\Users\hello\Documents\SEM3\PYTHON\Lab6\A2.py"
26
PS C:\Users\hello\Documents\SEM3\PYTHON\Lab6> python -u "c:\Users\hello\Documents\SEM3\PYTHON\Lab6\A3.py"
[5, 4, 3, 2, 1]
PS C:\Users\hello\Documents\SEM3\PYTHON\Lab6> python -u "c:\Users\hello\Documents\SEM3\PYTHON\Lab6\A4.py"
AAACGHIKLLMR
PS C:\Users\hello\Documents\SEM3\PYTHON\Lab6> python -u "c:\Users\hello\Documents\SEM3\PYTHON\Lab6\A5_A6.py"
Enter the number of elements: 5
Enter number: 1
Enter number: 3
Enter number: 5
Enter number: 2
Enter number: 6
Sorted in Descending Order: [6, 5, 3, 2, 1]
Enter number: 6
Sorted in Descending Order: [6, 5, 3, 2, 1]
Enter number: 6
Sorted in Descending Order: [6, 5, 3, 2, 1]
Enter number: 6
Sorted in Descending Order: [6, 5, 3, 2, 1]
Enter number: 6
Sorted in Descending Order: [6, 5, 3, 2, 1]
Enter number: 6
Sorted in Descending Order: [6, 5, 3, 2, 1]
Enter the element to search (using linear search): 2
Search Successful
Enter the element to search (using binary search): 2
Search Successful

⊗ 0 ⚠ 0                                   Ln 3, Col 125   Spaces: 4   UTF-8   CRL
```