Assignment - 1

Q1 :

Given a number x, determine whether the given number is Armstrong number or not. A positive integer of **n digits** is called an Armstrong number of **order n** (order is number of digits) if.

abcd... = pow(a,n) + pow(b,n) + pow(c,n) + pow(d,n) + ...

```
Input : 1253
Output : No
1253 is not a Armstrong Number
1*1*1*1 + 2*2*2*2 + 5*5*5*5 + 3*3*3*3 = 723
```

```
Input : 1634
Output : Yes
1*1*1*1 + 6*6*6*6 + 3*3*3*3 + 4*4*4*4 = 1634
```

```
#include <cmath>
#include <iostream>

using namespace std;

int main()
{
   int num, originalNum, remainder, n = 0, result = 0, power;
   cout << "Enter an integer: /n";
   cin >> num;

   originalNum = num;

 while (originalNum != 0) {
      originalNum /= 10;
      ++n;
   }
   originalNum = num;

   while (originalNum != 0) {
      remainder = originalNum % 10;
      power = round(pow(remainder, n));
      result += power;
      originalNum /= 10;
   }
```

```
  if (result == num)
    cout << num << " is an Armstrong number.";
  else
    cout << num << " is not an Armstrong number.";
  return 0;
}
```

Q2 :

Given a sorted array with possibly duplicate elements, the task is to find indexes of first and last occurrences of an element x in the given array.

Input : `arr[] = {1, 3, 5, 5, 5, 5, 7, 123, 125 }`

`x = 7`

Output : `First Occurrence = 6`

`Last Occurrence = 6`

```
#include <iostream>

using namespace std;

void findFirstAndLast(int arr[], int n, int x)
{
    int first = -1, last = -1;
    for (int i = 0; i < n; i++) {
        if (x != arr[i])
            continue;
        if (first == -1)
            first = i;
        last = i;
    }
    if (first != -1)
        cout << "First Occurrence = " << first
             << "\nLast Occurrence = " << last;
    else
```

```
        cout << "Not Found";
}

// Driver code
int main()
{
    int arr[] = { 1, 3, 5, 5, 5, 5, 7, 123, 125 };
    int n = sizeof(arr) / sizeof(int);
    int x = 7;
    findFirstAndLast(arr, n, x);
    return 0;
}
```

Q3 :

1. You are given a number n.

2. You've to create a pattern of * and separated by tab as shown in output format.

Input : 5

Output:

```
            *
          *  *
       *  *  *
    *  *  *  *
 *  *  *  *  *
```

```
#include <iostream>
using namespace std;

int main()
```

```
{
int i, j, k, n;
cout<<"enter a number to print pattern : "<<endl;
cin>>n;
for(i=n;i>=1;i--)
{
for(j=1;j<i;j++)
{
cout<<" ";
}
for(k=n;k>=i;k--)
{
cout<<"*";
}
cout<<"\n";
}
return 0;
}
```

Q4 :

1. You've to print all prime numbers between a range.
2. Take as input "low", the lower limit of range.
3. Take as input "high", the higher limit of range.
4. For the range print all the primes numbers between low and high (both included).

Input: low : 6
High: 24

Output: 7
11
13
17
19
23

```
#include <iostream>
using namespace std;

int main() {
    int low, high, i;
    bool isPrime = true;

    cout << "Enter low number : \n";
    cin >> low;
    cout << "Enter high number : \n";
    cin >> high;

    cout << "\nPrime numbers between " << low << " and " << high << " are: " << endl;

    while (low < high) {
        isPrime = true;
        if (low == 0 || low == 1) {
            isPrime = false;
        }
        else {
            for (i = 2; i <= low / 2; ++i) {
                if (low % i == 0) {
                    isPrime = false;
                    break;
                }
            }
```

```
        }

        if (isPrime)
            cout << low << " ";

        ++low;
    }

    return 0;
}
```

Q5 :

1. You are given a string that contains only lowercase and uppercase alphabets.
2. You have to toggle the case of every character of the given string.

Input : ProGraMMer

Output: pROgRAmmER

```
#include <iostream>
using namespace std;

int main()
{
        string str;
    cout<<"Enter the string : \n";
    getline(cin,str);
        for(int i=0;str[i]!='\0';i++)
        {
                if (str[i]>=65 && str[i]<=90 )
                        str[i] = str[i] + 32;
                else if (str[i]>=97 && str[i]<=122 )
                    str[i] = str[i] - 32;
        }

        cout<<"The converted string: \n"<< str;
```

```
return 0;
}
```