Q.1

#include<iostream>

using namespace std;

Void swap(int &x,int &y){

Int temp=x;

x=y

y=temp;

}

Int main(){

int a=10;

Int b=8;

swap(a,b);

cout<<a<<" "<<endl;

}

Q.2

#include <iostream>

using namespace std;

int main() {

float n1, n2, n3;

cout << "Enter three numbers: ";

cin >> n1 >> n2 >> n3;

if(n1 >= n2 && n1 >= n3)

cout << "Largest number: " << n1;

if(n2 >= n1 && n2 >= n3)

cout << "Largest number: " << n2;

if(n3 >= n1 && n3 >= n2)

cout << "Largest number: " << n3;

return 0;

}

Q.3

#include <iostream>

using namespace std;

int main() {

int year;

cout << "Enter a year: ";

cin >> year;

if (year % 4 == 0) {

if (year % 100 == 0) {

if (year % 400 == 0)

cout << year << " is a leap year.";

else

cout << year << " is not a leap year.";

}

else

cout << year << " is a leap year.";

}

else

cout << year << " is not a leap year.";

return 0;

}

Q.4#include <iostream>

using namespace std;

int main() {

int n, t1 = 0, t2 = 1, nextTerm = 0;

cout << "Enter the number of terms: ";

cin >> n;

cout << "Fibonacci Series: ";

for (int i = 1; i <= n; ++i) {

// Prints the first two terms.

if(i == 1) {

cout << t1 << ", ";

continue;

}

if(i == 2) {

cout << t2 << ", ";

continue;

}

nextTerm = t1 + t2;

t1 = t2;

t2 = nextTerm;

cout << nextTerm << ", ";

}

return 0;

}

5.Q

#include <iostream>

using namespace std;

int main() {

int i, n;

bool isPrime = true;

cout << "Enter a positive integer: ";

cin >> n;

// 0 and 1 are not prime numbers

if (n == 0 || n == 1) {

isPrime = false;

}

else {

for (i = 2; i <= n / 2; ++i) {

if (n % i == 0) {

isPrime = false;

break;

}

}

}

if (isPrime)

cout << n << " is a prime number";

else

cout << n << " is not a prime number";

return 0;

}

Q.6

#include <iostream>

using namespace std;

int main()

{

int rows;

cout << "Enter number of rows: ";

cin >> rows;

for(int i = 1; i <= rows; ++i)

{

for(int j = 1; j <= i; ++j)

{

cout << "\* ";

}

cout << "\n";

}

return 0;

}

Q.7#include <iostream>

using namespace std;

int main(){

int n, num[50], largest, second;

cout<<"Enter number of elements: ";

cin>>n;

for(int i=0; i<n; i++){

cout<<"Enter Array Element"<<(i+1)<<": ";

cin>>num[i];

}

if(num[0]<num[1]){

largest = num[1];

second = num[0];

}

else{

largest = num[0];

second = num[1];

}

for (int i = 2; i< n ; i ++) {

if (num[i] > largest) {

second = largest;

largest = num[i];

}

else if (num[i] > second && num[i] != largest) {

second = num[i];

}

}

cout<<"Second Largest Element in array is: "<<second;

return 0;

Q.8

#include <bits/stdc++.h>

using namespace std;

string ltrim(const string &);

string rtrim(const string &);

vector<string> split(const string &);

vector<int> rotateLeft(int d, vector<int> arr) {

}

int main()

{

    ofstream fout(getenv("OUTPUT\_PATH"));

    string first\_multiple\_input\_temp;

    getline(cin, first\_multiple\_input\_temp);

    vector<string> first\_multiple\_input = split(rtrim(first\_multiple\_input\_temp));

    int n = stoi(first\_multiple\_input[0]);

    int d = stoi(first\_multiple\_input[1]);

    string arr\_temp\_temp;

    getline(cin, arr\_temp\_temp);

    vector<string> arr\_temp = split(rtrim(arr\_temp\_temp));

    vector<int> arr(n);

    for (int i = 0; i < n; i++) {

        int arr\_item = stoi(arr\_temp[i]);

        arr[i] = arr\_item;

    }

    vector<int> result = rotateLeft(d, arr);

    for (size\_t i = 0; i < result.size(); i++) {

        fout << result[i];

        if (i != result.size() - 1) {

            fout << " ";

        }

    }

    fout << "\n";

    fout.close();

    return 0;

}

string ltrim(const string &str) {

    string s(str);

    s.erase(

        s.begin(),

        find\_if(s.begin(), s.end(), not1(ptr\_fun<int, int>(isspace)))

    );

    return s;

}

string rtrim(const string &str) {

    string s(str);

    s.erase(

        find\_if(s.rbegin(), s.rend(), not1(ptr\_fun<int, int>(isspace))).base(),

        s.end()

    );

    return s;

}

vector<string> split(const string &str) {

    vector<string> tokens;

    string::size\_type start = 0;

    string::size\_type end = 0;

    while ((end = str.find(" ", start)) != string::npos) {

        tokens.push\_back(str.substr(start, end - start));

        start = end + 1;

    }

    tokens.push\_back(str.substr(start));

    return tokens;

}