1.

(a)

#include<bits/stdc++.h>

using namespace std;

int main(){

    vector<int> v = {1,1,2,4,1};

    cout << count(v.begin(),v.end(),v[0]);

}

(b)

#include<bits/stdc++.h>

using namespace std;

#define r 3

#define c 2

void transpose(int a[][c] ){

    int b[c][r];

    for(int i=0;i<r;i++){

        for(int j=0;j<c;j++){

            b[j][i] = a[i][j];

        }

    }

    for(int i=0;i<c;i++){

        for(int j=0;j<r;j++){

            cout << b[i][j] << " ";

        }

        cout << endl;

    }

}

int main(){

    int a[r][c];

    for(int i=0;i<r;i++){

        for(int j=0;j<c;j++){

            cin >> a[i][j];

        }

    }

    transpose(a);

}

2.

**a)**

**f(n) = Θ(g(n)) means there are positive constants c1, c2, and k, such that 0 ≤ c1g(n) ≤ f(n) ≤ c2g(n) for all n ≥ n0. The values of c1, c2, and n0 must be fixed for the function f and must not depend on n.**

**b)**

**f(n) = O(g(n)) means there are positive constants c and n0, such that 0 ≤ f(n) ≤ cg(n) for all n ≥ n0**

**Let's assume c = 1**

**it's true for all n >= 1**

3.

**a)**

**f(n) = O(g(n)) means there are positive constants c and n0, such that 0 ≤ f(n) ≤ cg(n) for all n ≥ n0**

**(dependent on n)**

**b)**

**for f(n) = Θ(g(n)), then f(n) = O(g(n)) and f(n) = Ω(g(n)) must both be true**

**so, let's first check whether f(n) = Ω(g(n)) is true or not**

**all n ≥ n0**

**for f(n) = Ω(g(n)) then c must be a positive constant. here c is found to be non-negative**

4.

#include<bits/stdc++.h>

using namespace std;

class Complex{

    float a,b;

    public:

        Complex(){

            a = 0,b = 0;

            cout << "Default Constructor: "<<a<<"+"<<b<<"i"<<endl;

        }

};

int main(){

    Complex ans;

}

5.

#include<bits/stdc++.h>

using namespace std;

class Quadratic{

   public:

        int a,b,c;

        Quadratic operator+(Quadratic &d){

            Quadratic func;

            func.a=a+d.a;

            func.b=b+d.b;

            func.c=c+d.c;

            return func;

        }

        void output(){

            cout<<a<<"x^2"<<"+"<<b<<"x"<<"+"<<c<<"="<<"0"<<endl;

        }

};

int main(){

    Quadratic Q1,Q2,Q3;

    cout<<"Enter first polynomial: "<<endl;

    cin>>Q1.a>>Q1.b>>Q1.c;

    cout <<endl<<"Enter second polynomial: "<<endl;

    cin>>Q2.a>>Q2.b>>Q2.c;

    Q3=Q1+Q2;

    cout <<endl<<"First polynomial is: "<<endl;

    Q1.output();

    cout <<endl<<"Second polynomial is: "<<endl;

    Q2.output();

    cout <<endl<<"Addition of two polynomial: "<<endl;

    Q3.output();

}

6.

#include<bits/stdc++.h>

template <typename T> class bag

{

    protected:

        T \*arr;

        int capacity;

        int top;

    public:

        bag(int bagCapacity = 10){

            capacity = 10;

            arr = new T[10];

            top=0;

        };

        virtual int Size() const{

            return top;

        }

        virtual bool IsEmpty() const{

            return top==0;

        }

        virtual T Element() const{

            return arr[top];

        }

        virtual void Push(const T &x){

            arr[top++] = x;

        }

        virtual void Pop(){

            --top;

        }

};

template <typename TT> class Queue : public bag<TT>{

    private:

        int l , r;

    public:

        Queue(int n = 10){

            l = 0 , r = -1;

        }

        virtual void Pop(){

            ++l;

        }

        virtual void Push(const TT &x){

            this->arr[++r] = x;

        }

        virtual TT Element(){

            return this->arr[l];

        }

        virtual int Size(){

            return r-l+1;

        }

        virtual int IsEmpty(){

            return r-l+1==0;

        }

};

int main(){

    Queue<int> q;

    q.Push(3);

    q.Push(17);

    q.Push(2);

    std::cout << q.Element() << '\n';

    q.Pop();

    std::cout << q.Element() << '\n';

}

7.

A \* B \* C = \*C\*AB

8.

#include<bits/stdc++.h>

using namespace std;

class sparseMatrix{

    public:

        sparseMatrix(int \_r,int \_c){

            r = \_r; c = \_c;

            m = new int\*[r];

            for(int i=0;i<r;i++){

                m[i] = new int[c];

            }

            for(int i=0;i<r;i++){

                for(int j=0;j<c;j++){

                    m[i][j] = 0;

                }

            }

            for(int i=0;i<r;i++){

                for(int j=0;j<c;j++){

                    if(i==j){

                        m[i][j] = 2;

                    }else if(i==0 && j==(c-1)){

                        m[i][j] = 7;

                    }else{

                        m[i][j] = 0;

                    }

                }

            }

        }

        sparseMatrix(const sparseMatrix &M){

            r = M.r;

            c = M.c;

            m = new int\* [r];

            for(int i=0;i<r;i++){

                m[i] = new int[c];

            }

            for(int i=0;i<r;i++){

                for(int j=0;j<c;j++){

                    m[i][j] = M.m[i][j];

                }

            }

        }

        void print(){

            for(int i=0;i<r;i++){

                for(int j=0;j<c;j++){

                    cout << m[i][j] << " ";

                }

                cout << endl;

            }

        }

    private:

        int\*\* m;

        int r,c;

};

int main(){

    sparseMatrix sparsematrix\_1(12,12);

    sparseMatrix sparsematrix\_2(sparsematrix\_1);

    cout << "First sparsematrix" << endl;

    sparsematrix\_1.print();

    cout << endl;

    cout << "Copy sparsematrix" << endl;

    sparsematrix\_2.print();

}

9.

#include<bits/stdc++.h>

using namespace std;

void printArray(int a[],int len){

    for(int i=0;i<len;i++){

        cout<<a[i]<<" ";

    }

    cout<<endl;

}

void Sort(int a[],int len){

    for(int i=1;i<len;i++){

        int temp = a[i];

        int j = i-1;

        while(temp < a[j] && j>=0){

            a[j+1] = a[j];

            j--;

        }

        a[j+1] = temp;

    }

}

int main(){

    int a[5] = {3,2,4,1,5};

    cout<<"original:\n";

    printArray(a,5);

    Sort(a,5);

    cout<<"sorted:\n";

    printArray(a,5);

}

10.

#include<bits/stdc++.h>

using namespace std;

int main(){

    string s;

    cin >> s;

    int len = s.size();

    bool isP = 1;

    for(int l=0,r=s.size()-1;l<=r;l++,r--){

        if(s[l]!=s[r]){

            isP = 0;

        }

    }

    if(isP==1){

        cout<<"Yes";

    }else cout<<"No";

}