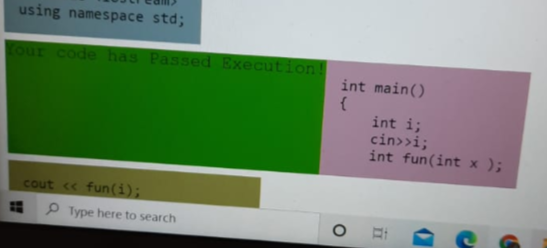
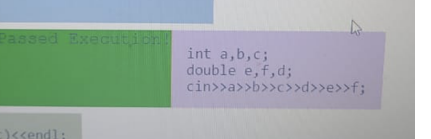
CAT-2 PDF

Debug

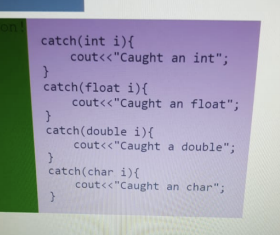
D1 (1)



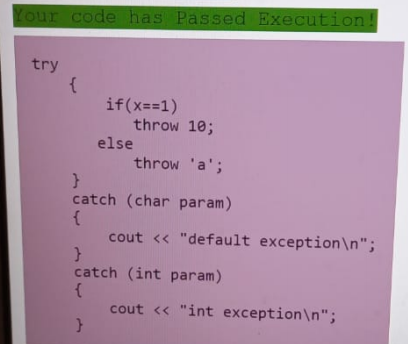
D1 (2)



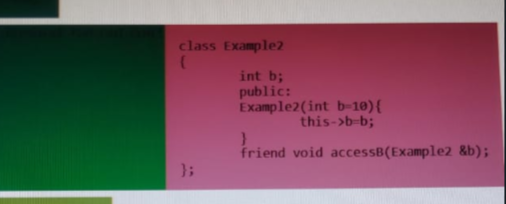
D1 (3)



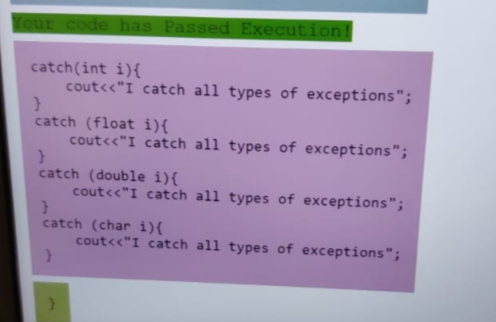
D2 (1)



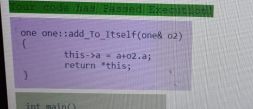
D2 (2)



D2 (3)



D3 (1)



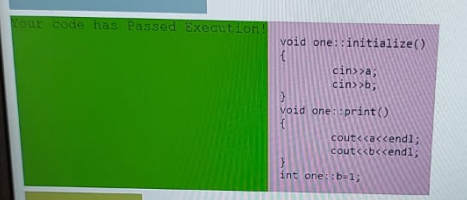
D3 (2)



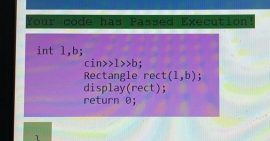
D3 (3)



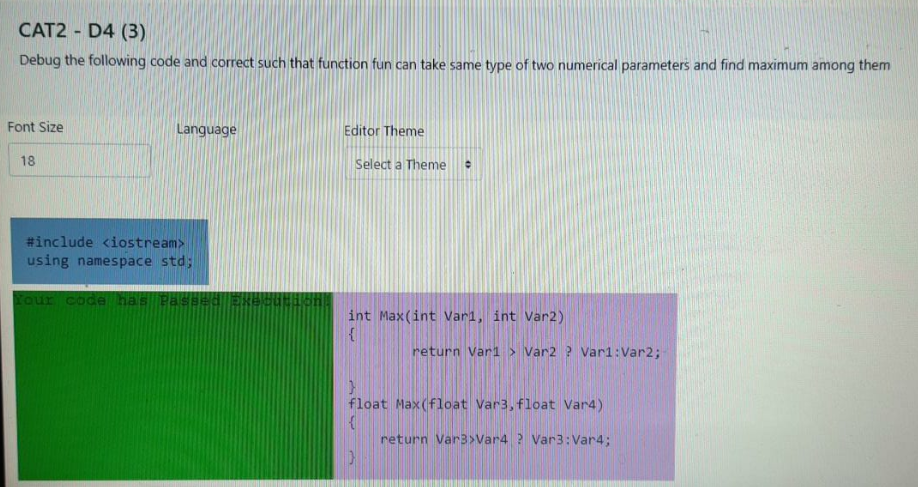
D4 (1)



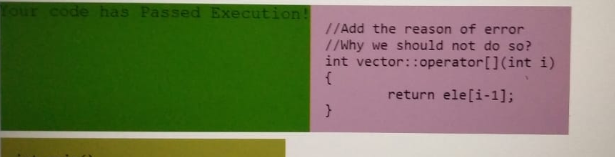
D4 (2)



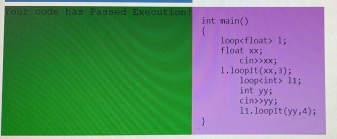
D4 (3)



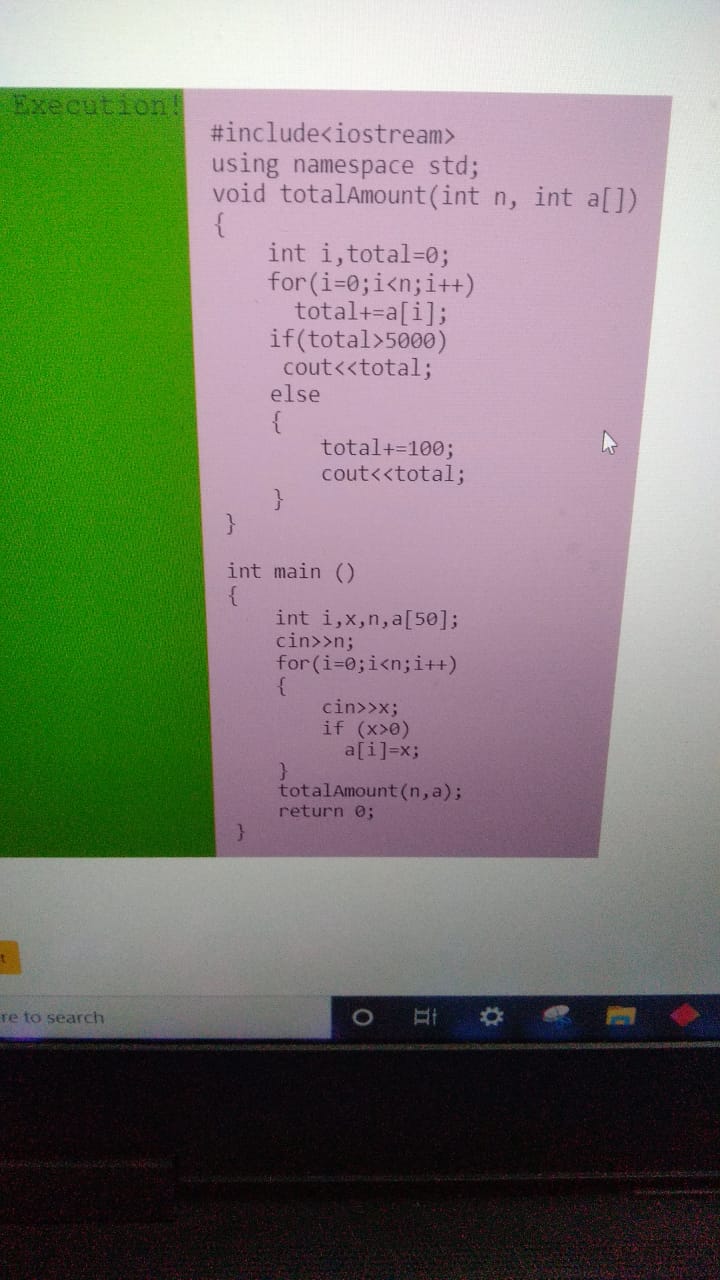
D5 (1)



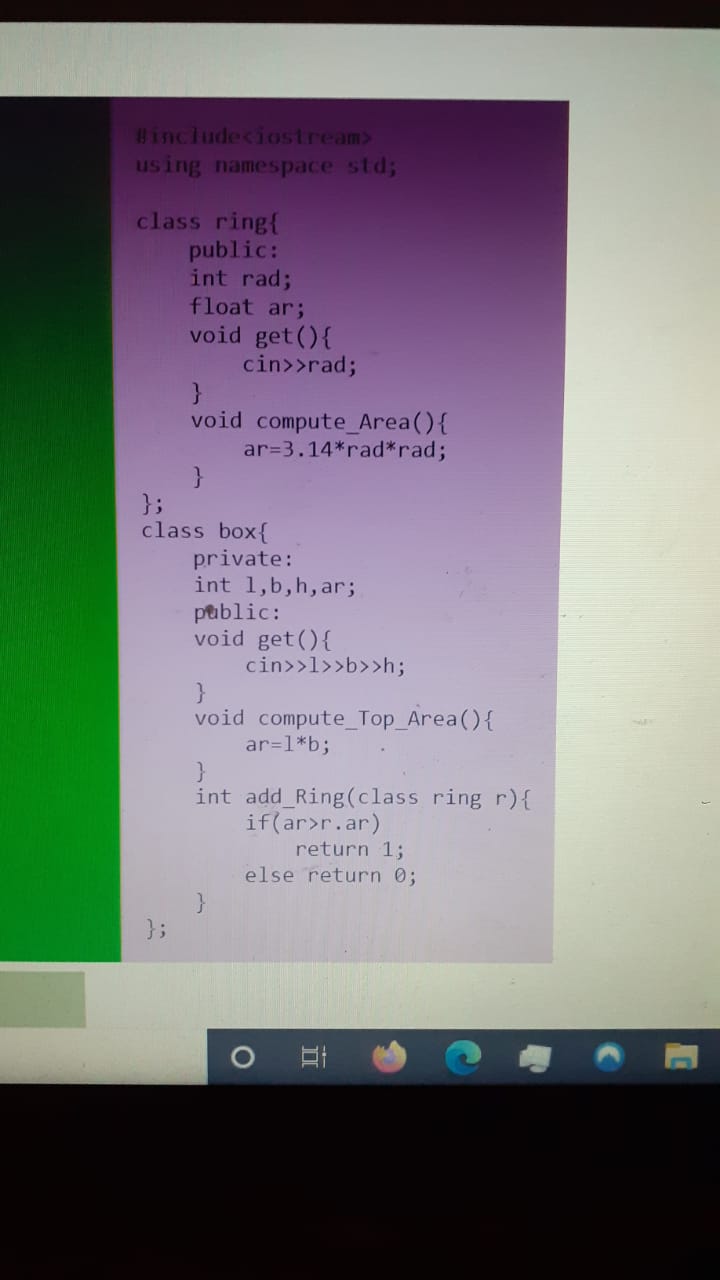
D5 (2)



Online Billing

****

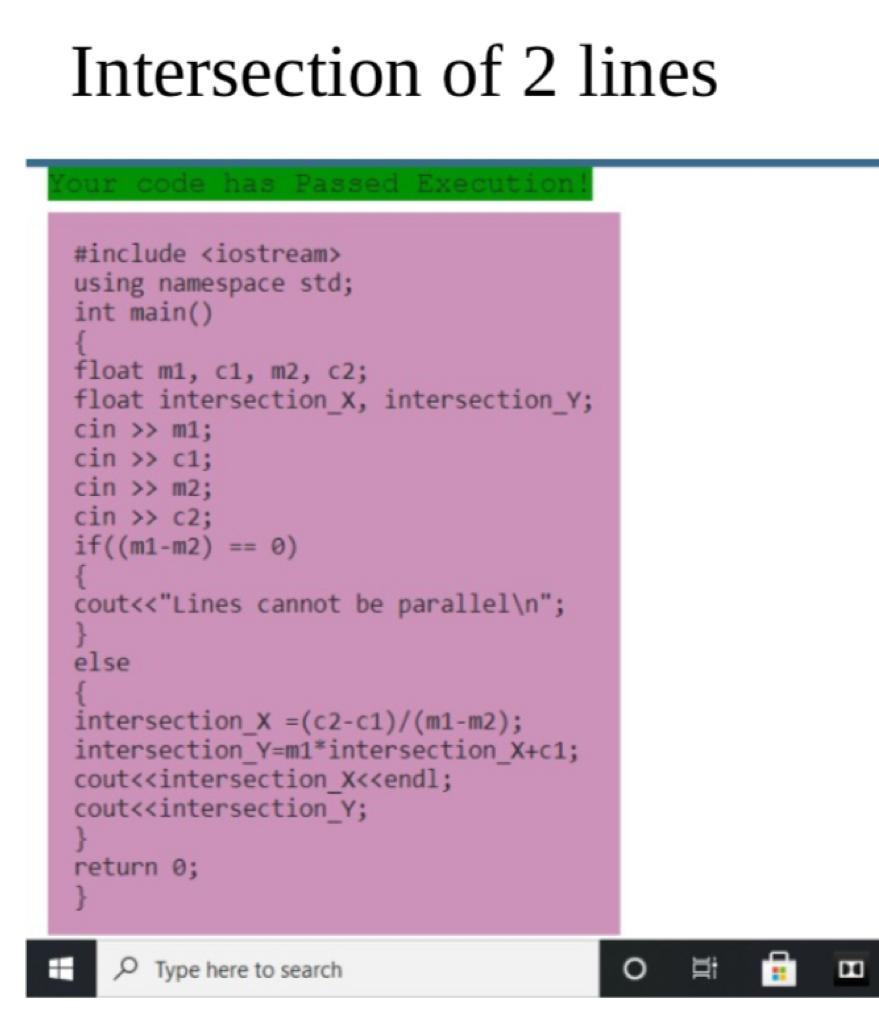
Put a Ring

****

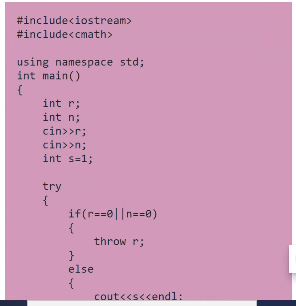
Descending Concatenation

****

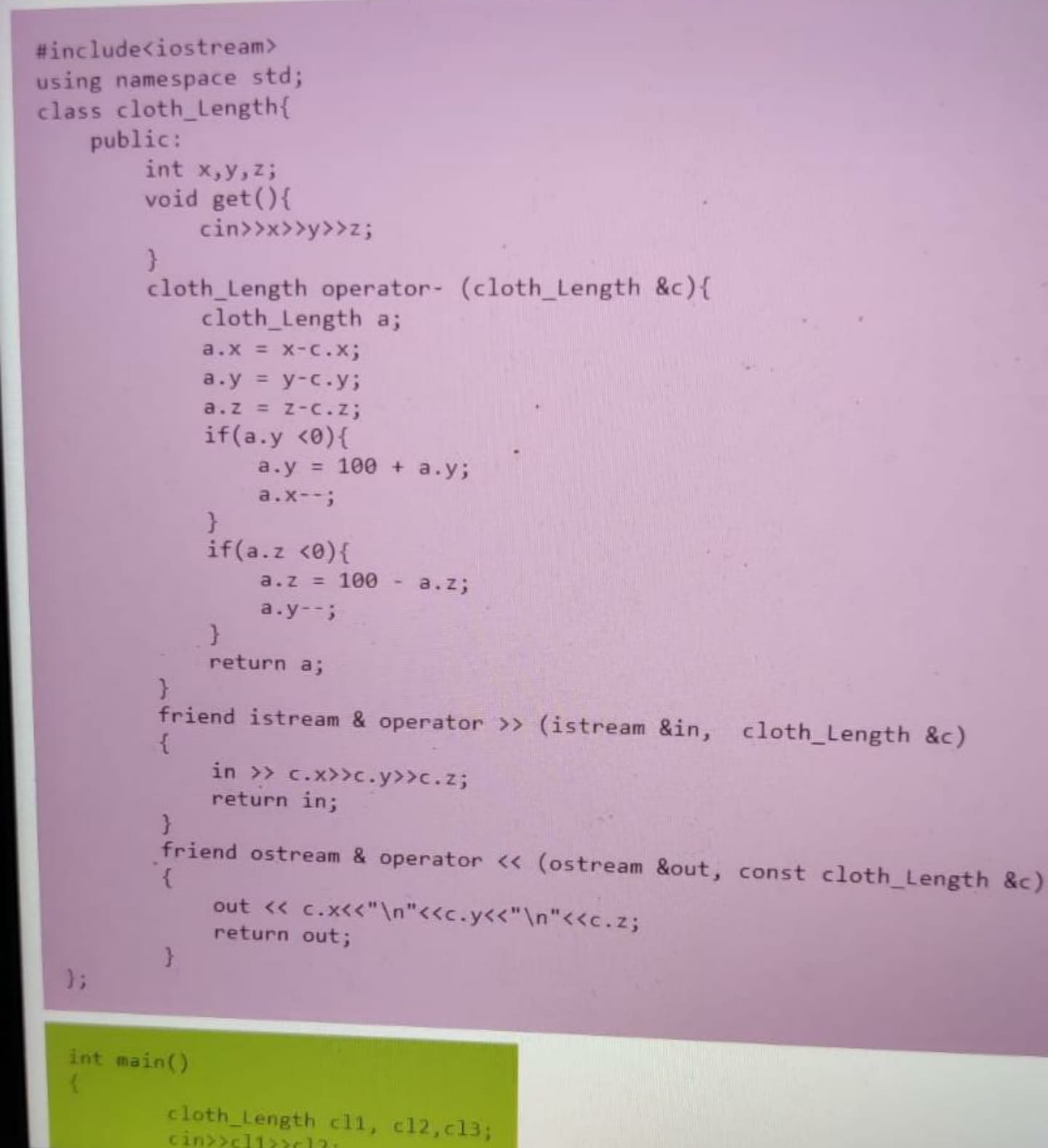
Intersection of 2 lines

****

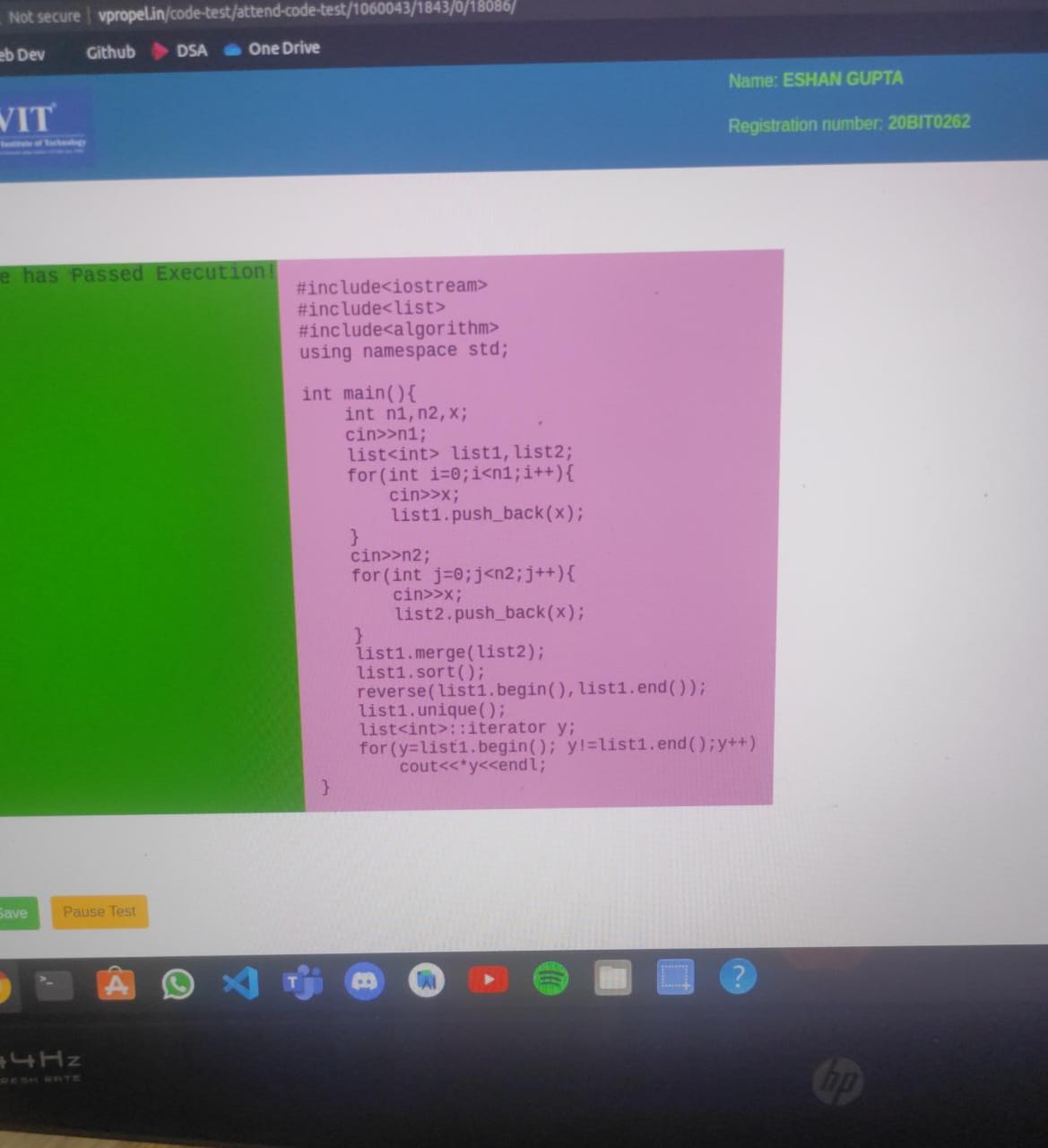
GP



Cloth

****

Sorted concentration

****

Search for high score

-----

#include <iostream>

#include <cmath>

#include <string.h>

using namespace std;

class student{

public:

string name;

string regno;

int marks;

void get(){

        cin>>name>>regno>>marks;

}

};

void search(student s[], int n, int smark){

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

     if(s[i].marks>=n){

            cout<<s[i].name<<endl;

         cout<<s[i].regno<<endl;

            cout<<s[i].marks<<endl;

     }

}

}

void search(student s[], int n, int hmark, int lmark){

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

     if(s[i].marks>lmark && s[i].marks<hmark){

            cout<<s[i].name<<endl;

            cout<<s[i].regno<<endl;

            cout<<s[i].marks<<endl;

     }

}

}

Round trip

---------

void date::get()

{

cin>>day>>month>>year;

}

void date::print()

{

cout<<day<<endl;

cout<<month<<endl;

cout<<year<<endl;

}

date date::ret\_Date(int d)

{

int months[]={31,28,31,30,31,30,31,31,30,31,30,31};

date r;

r.day = this->day;

r.month = this->month;

r.year = this->year;

if(d>30)

{

     cannot\_Add=true;

     return r;

}

else if(d<0)

{

     invalid\_Flag=true;

     return r;

}

else

     {

         r.day=r.day+d;

         if(r.day>months[r.month-1])

         {

             r.day=r.day-months[r.month-1];

             r.month=r.month+1;

         }

         if(r.month>12)

         {

             r.year=r.year+1;

             r.month=1;

         }

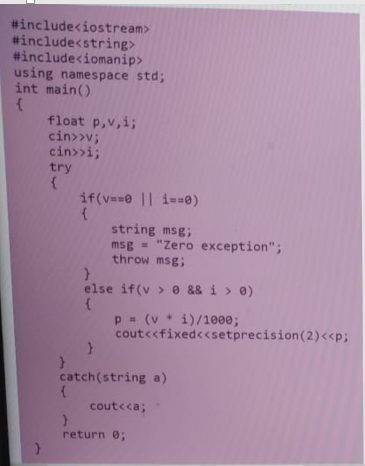
         return r;

     }

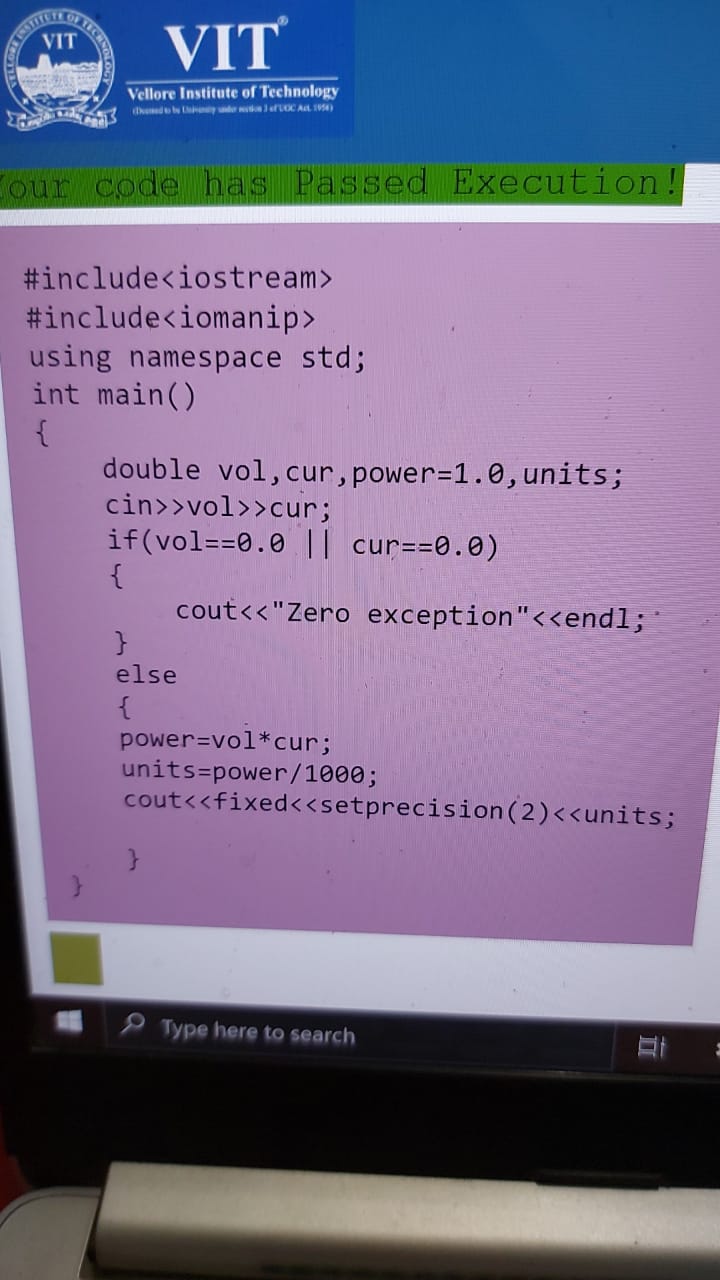
}

Power consumed

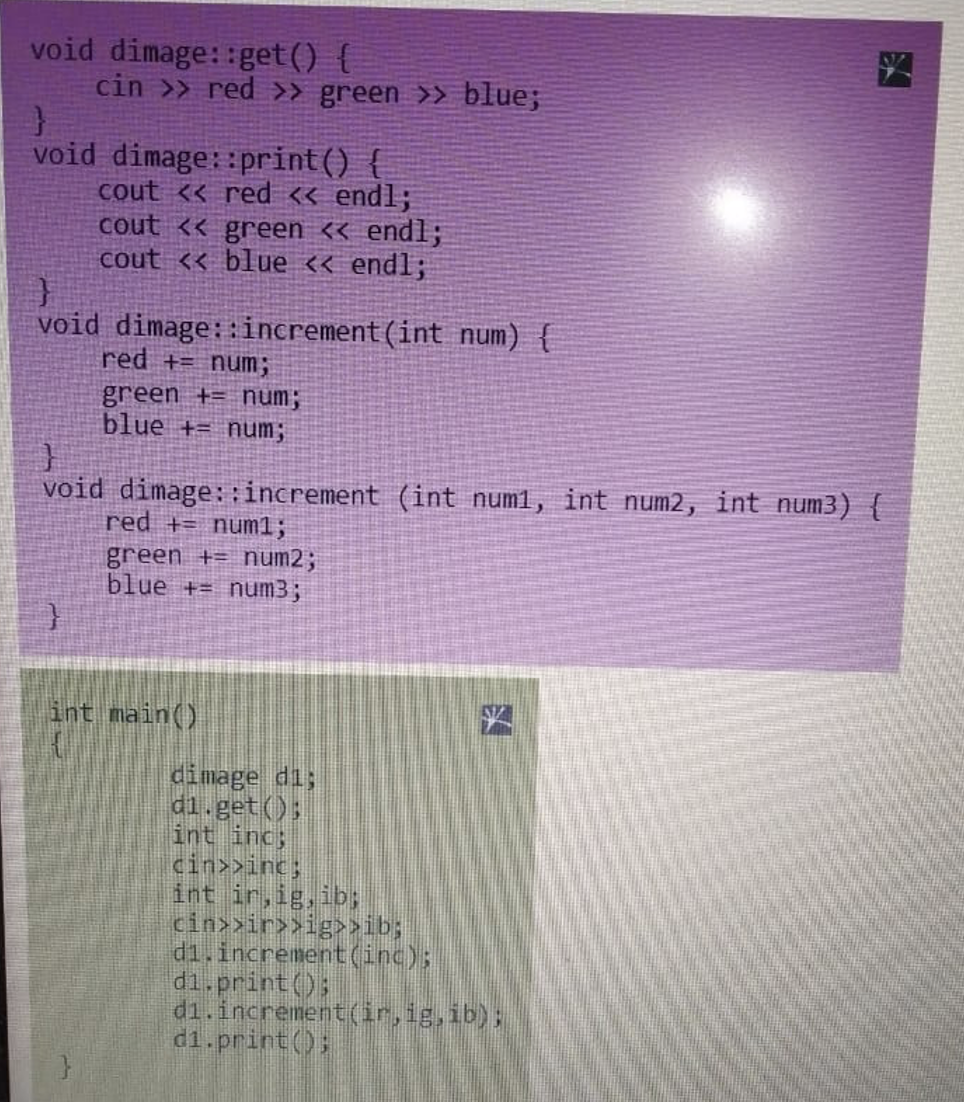
1st-



2nd-

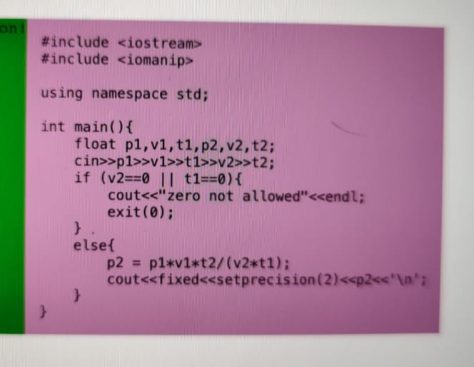


Digital Image

****

N2 Pressure

**1st Code**



#include<iostream>

#include <iomanip>

using namespace std;

int main()

{

float p1, v1, t1, p2, v2, t2;

cin >> v1 >> p1 >> t1 >> v2 >> t2;

if (v2==0 || t1==0)

{

     cout << "zero not allowed" << endl;

     exit(0);

}

else

{

     p2 = p1\*v1\*t2/(v2\*t1);

     cout << std::setprecision(2) << p2 << '\n';

}

}

**2nd Code**

#include<iostream>

#include<string>

#include<iomanip>

using namespace std;

int main()

{

float p,v,i;

cin>>v;

cin>>i;

try

{

     if(v==0 || i==0)

     {

         string msg;

         msg = "Zero Exception";

         throw msg;

     }

     else if(v > 0 && i > 0)

     {

         p = (v \* i)/1000;

            cout<<fixed<<setprecision(2)<<p;

     }

}

catch(string a)

{

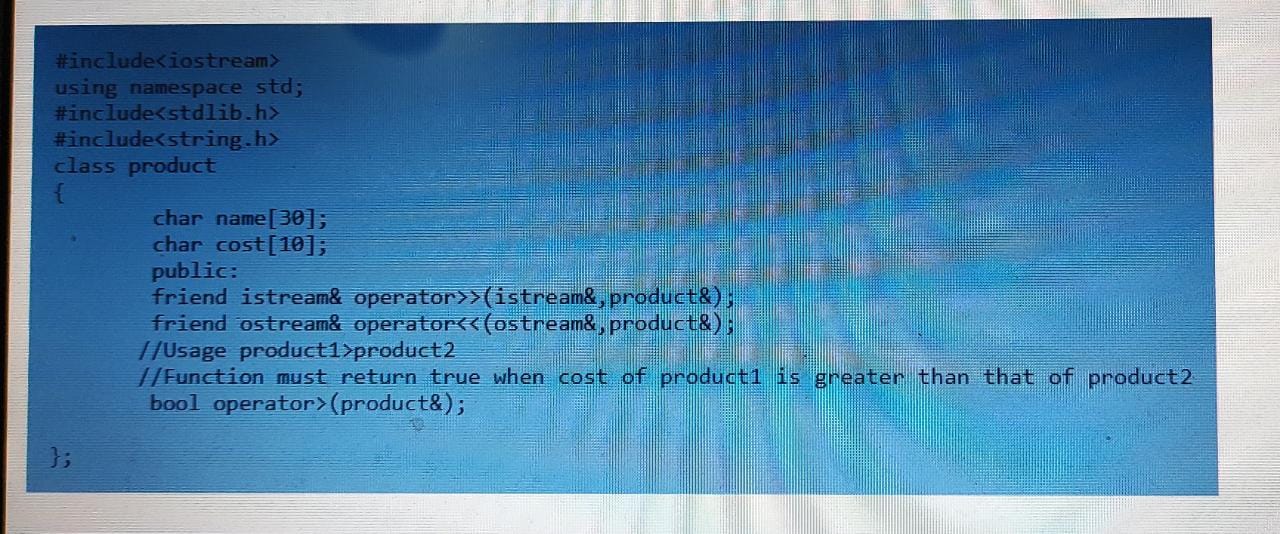
     cout<<a;

}

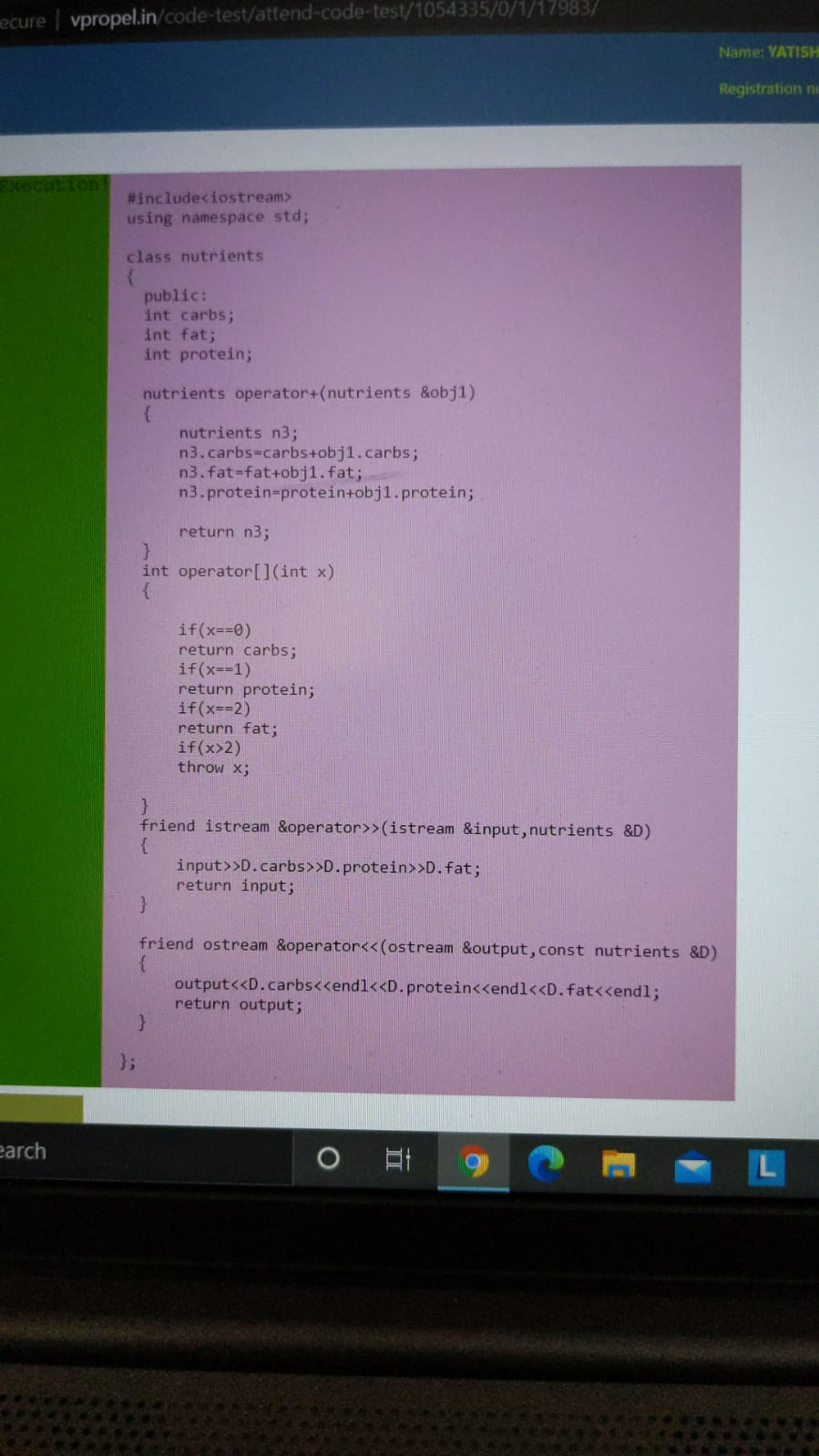
return 0;

}

Arrange Dolls

****

Dietician



Compute distance between two points

#include <iomanip>

#include <iostream>

#include <math.h>

using namespace std;

// function to print distance

void distance(int x1, int y1,

int z1, int x2,

int y2, int z2)

{

float d = sqrt(pow(x2 - x1, 2) +

pow(y2 - y1, 2) +

pow(z2 - z1, 2) \* 1.0);

float c = sqrt(pow(x2-x1,2)+pow(y2-y1,2)\*1.0);

std::cout << std::fixed;

std::cout << std::setprecision(2);

cout<<c<<endl<<d;

return;

}

// Driver Code

int main()

{

int x1,x2,y1,y2,z1,z2;

cin>>x1>>y1>>z1>>x2>>y2>>z2;

// function call for distance

distance(x1, y1, z1,

x2, y2, z2);

return 0;

}

Identify Late employees

#include <iostream>

using namespace std;

class UDtime{

int hours,min,sec;

public:

bool operator<(UDtime&);

friend istream& operator>>(istream&,UDtime&);

friend ostream& operator<<(ostream&,UDtime&);

};

istream& operator>>(istream& in,UDtime& t)

{

in>>t.hours>>t.min>>t.sec;

return in;

}

ostream& operator<<(ostream& out,UDtime& t)

{

out<<"(" <<t.hours << "," << t.min << "," << t.sec<<")";

return out;

}

bool UDtime::operator<(UDtime& t)

{

int t1 = hours\*3600+min\*60+sec;

int t2 = t.hours\*3600+t.min\*60+t.sec;

return t1<t2;

}

class emp{

int empid;

char name[20];

UDtime arr\_Time;

public:

friend istream& operator>>(istream&,emp&);

friend ostream& operator<<(ostream&,emp&);

UDtime get\_Time();

void operator[](int);

};

istream& operator>>(istream& in,emp& e)

{

in>>e.empid>>e.name>>e.arr\_Time;

return in;

}

ostream& operator<<(ostream& out,emp& e)

{

out<<"(" <<e.empid << "," << e.name <<")";

return out;

}

UDtime emp::get\_Time()

{

return arr\_Time;

}

int main() {

int num;

emp e[20];

UDtime exp\_Arr\_Time,a\_Time;

cin>>num;

for(int k=0;k<num;k++)

cin>>e[k];

cin>>exp\_Arr\_Time;

for(int j=0;j<num;j++)

{

a\_Time = e[j].get\_Time();

if(exp\_Arr\_Time<a\_Time)

cout<<e[j]<<endl;

}

return 0;

}

Total cost of transaction

#include <iostream>

#include <map>

#include <iomanip>

using namespace std;

int main()

{

map <string, pair<float, int> > items;

int n; cin >> n;

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

{

string s; cin >> s;

float cost;

int quan; cin >> cost >> quan;

items.insert(pair<string, pair<float, int> >(make\_pair(s, make\_pair(cost, quan))));

}

int m; cin >> m;

float finalPay = 0.00;

map <string, pair<float, int> > :: iterator itr;

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

{

string s; cin >> s;

int quan; cin >> quan;

itr = items.find(s);

if(itr != items.end())

{

pair<float, int> \*temp = &(itr -> second);

finalPay += quan \* (temp -> first);

}

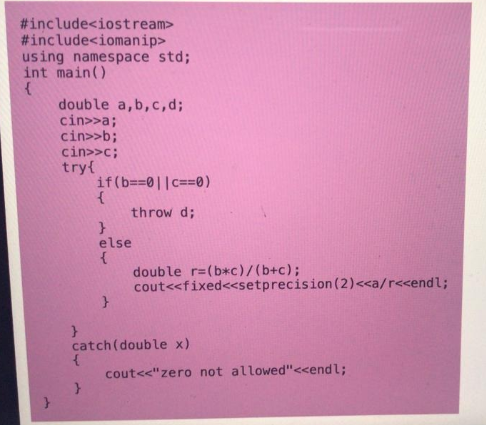
}

cout << fixed << setprecision(2) << (float)finalPay << endl;

return 0;

}

Current in parallel



Doll

void sortDolls(vector‹Doll\*›& list, int size)

{

if(isBigger<list.isBigger){

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

Doll \*innerDoll = new Doll(i);

list.push\_back(innerDoll);

}

}

int j=0,k=0;

bool swap = true;

string temp="";

while (swap)

{

for(int l=0;l<size-j;l++)

{

if(innerDoll[l]>temp)

{

temp=innerDoll[l];

k=l;

}

}

j++;

innerDoll[k]=innerDoll[size-j];

innerDoll[size-j]=temp;

k=0;

temp="";

for (int l = 0; l < size - j; l++)

{

if (innerDoll[l] > innerDoll[l + 1])

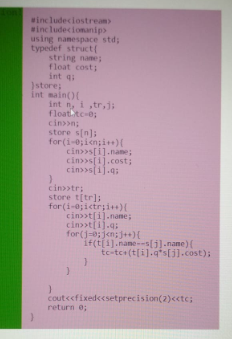
swap = true;

}

}

}

Total cost and library



#include <iostream>

#include <string>

using namespace std;

void search(string one, string two){

int counter = 0;

for(int i=0;i<one.size();i++){

if(one[i]==two[counter]){

counter++;

}

else if(counter>0){

break;

}

else{

;

}

}

if(counter==two.size()){

cout << 1;

}

else{

cout << -1 << endl;

}

}

void searchpos(string one, string two, int pos){

int counter = 0;

for(int i=pos-1;i<one.size();i++){

if(one[i]==two[counter]){

counter++;

}

else if(counter>0){

break;

}

else{

;

}

}

if(counter==two.size()){

cout << 1;

}

else{

cout << -1;

}

}

int main(){

string a;

string b;

cin>>a>>b;

int pos = 0;

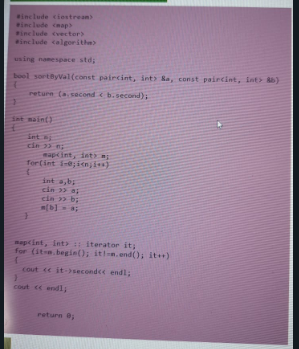
cin>>pos;

search(a,b);

searchpos(a,b,pos);

}

Plants using maps





Minimum marks

tem

#include <iostream>

using namespace std;

class student

{

public:

int rno;

string name;

int marks;

};

int Min(int arr[],int n)

{

int i;

int min;

min=arr[0];

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

{

if(min>arr[i])

{

min=arr[i];

}

}

return(min);

}

int main()

{

int n,i,a;

cin>>n;

student s[n];

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

{

cin>>s[i].rno;

cin>>s[i].name;

cin>>s[i].marks;

}

int b[n];

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

{

b[i]=s[i].marks;

}

a=Min(b,n);

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

{

if(s[i].marks==a)

{

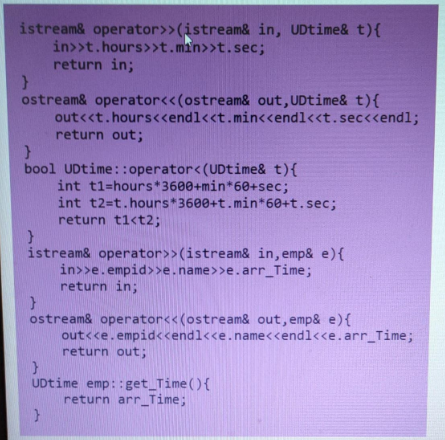
cout<<s[i].rno;

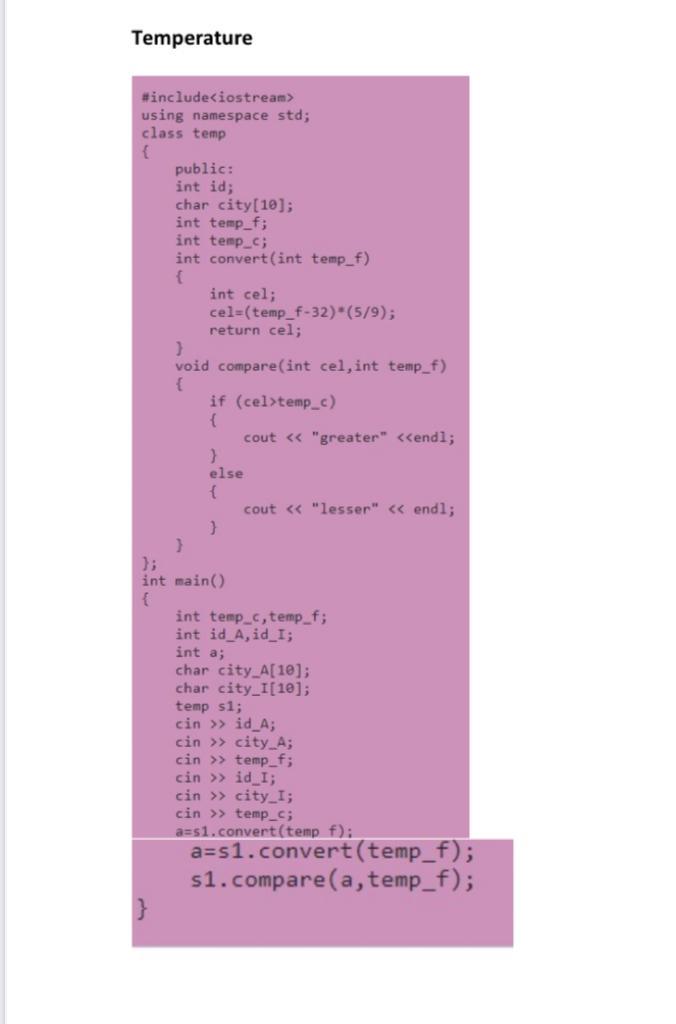
}

}

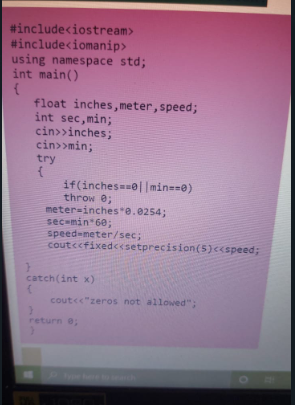
}

Late employee



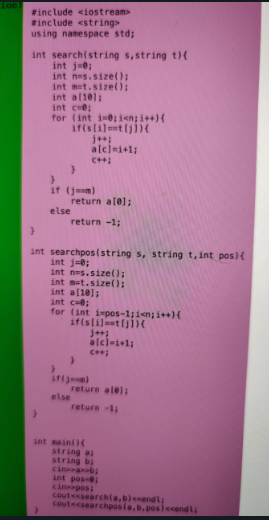
Temperature  


Speed of tortoise



Search a sub sequence

1st code-



**2nd code-**

**#include <iostream>**

**#include <string>**

**using namespace std;**

**int check(string a,string b)**

**{**

**int s1,s2;**

**s1 = a.size();**

**s2 = b.size();**

**int x[10];**

**int y = 0;**

**int j = 0;**

**for(int i=0;i<s1;i++)**

**{**

**if(a[i]==b[j])**

**{**

**j++;**

**x[y] = i+1;**

**y++;**

**}**

**}**

**if(j==s2)**

**return x[0];**

**else**

**return -1;**

**}**

**int check\_position(string a,string b,int position)**

**{**

**int s1,s2;**

**s1 = a.size();**

**s2 = b.size();**

**int x[10];**

**int y = 0;**

**int j = 0;**

**for(int i=(position-1);i<s1;i++)**

**{**

**if(a[i]==b[j])**

**{**

**j++;**

**x[y] = i+1;**

**y++;**

**}**

**}**

**if(j==s2)**

**return x[0];**

**else**

**return -1;**

**}**

**int main()**

**{**

**string a,b;**

**cin>>a>>b;**

**int position=0;**

**cin>>position;**

**cout<<check(a,b)<<endl;**

**cout<<check\_position(a,b,position)<<endl;**

**}**

Books in library

#include<iostream>

using namespace std;

#include<string.h>

class library;

class book

{

char author\_name[20];

char title[20];

char publisher[20];

int stock;

public:

void get();

void print();

friend class library;

};

void book::get()

{

cin >> author\_name >> title >> publisher >>stock;

}

void book::print()

{

cout << author\_name << endl << title << endl << publisher << endl << stock << endl;

}

class library

{

int num\_Of\_Books;

book b[20];

public:

void get();

//title or author

int search(char\*)

//title and author

int search(char\*,char\*);

void print(int);

};

void library::print(int i)

{

b[i].print();

}

void library::get()

{

cin >> num\_Of\_Books;

for(int i=0;i<num\_Of\_Books;i++)

{

b[i].get();

}

}

void library::search(char arr[20])

{

for(int i=0;i<num\_Of\_Books;i++)

{

if(b[i].author\_name==arr)

{

print(i);

}

}

}

void library::search(char arr1[20],char arr2[20])

{

for(int i=0;i<num\_Of\_Books;i++)

{

if(b[i].author\_name==arr1 && b[i].title==arr2)

{

print(i);

}

}

}

int main()

{

library books;

books.get();

int option;

cin >> option;

if(option==1)

{

char auth[20];

cin >> auth;

books.search(auth);

}

else if(option==2)

{

char auth[20],title[20];

cin >> auth >> title ;

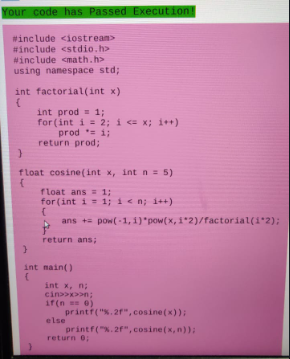
books.search(auth,title);

}

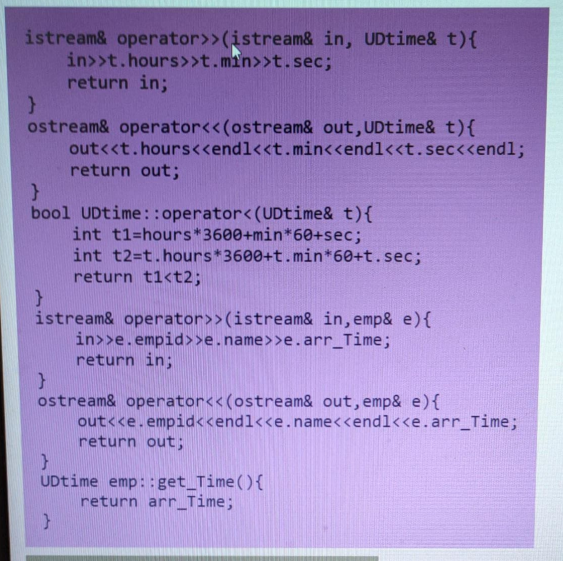
return 0;

}

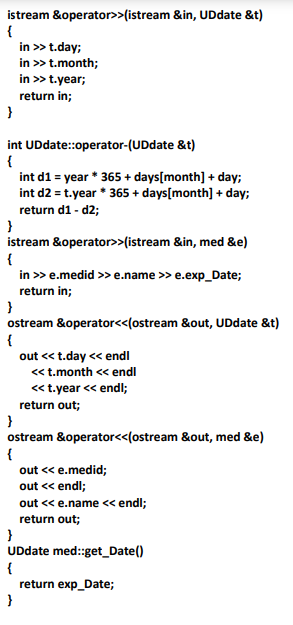
Cosine

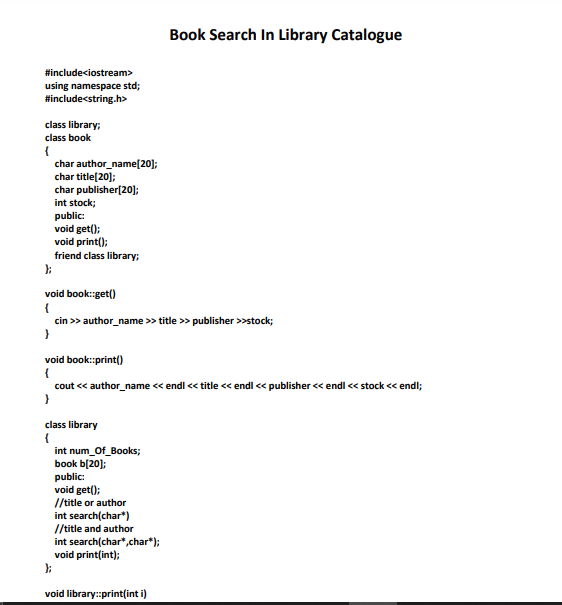


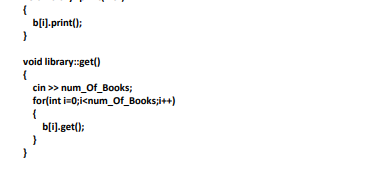
Late fine

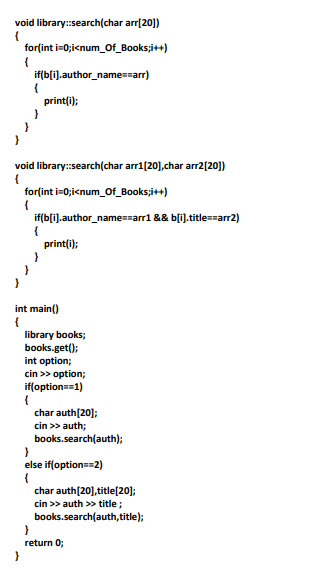


Expired medicine

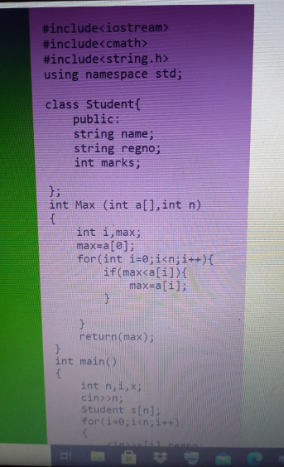


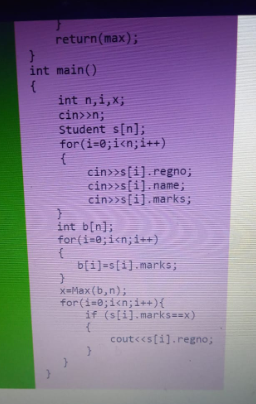




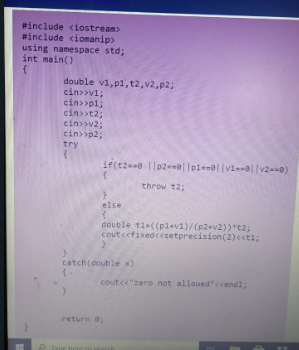


Maximum Marks





Gas Temperature



Campus Interview

#include<iostream>

using namespace std;

class comp1

{

private:

float ap,tr1,tr2,hr;

float res1;

public:

void get()

{

cin>>ap>>tr1>>tr2>>hr;

}

void set()

{

res1=1.25\*ap+2.5\*tr1+2.5\*tr2+3.75\*hr;

cout<<res1<<endl;

}

};

class comp2

{

private:

float ap2,tr12,hr2;

float res2;

public:

void get1()

{

cin>>ap2>>tr12>>hr2;

}

void set1()

{

res2=3\*ap2+5\*tr12+2\*hr2;

cout<<res2<<endl;

}

};

int main()

{

comp1 c1;

comp2 c2;

c1.get();

c2.get1();

c1.set();

c2.set1();

return 0;

}

Common Elements in Ascending Order

#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

int main()

{

int m, n;

cin >> m;

vector<int> v1(m);

for (int i = 0; i < m; i++)

{

int temp1;

cin >> temp1;

v1.push\_back(temp1);

}

cin >> n;

vector<int> v2(n);

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

{

int temp2;

cin >> temp2;

v2.push\_back(temp2);

}

sort(v1.begin(), v1.end());

sort(v2.begin(), v2.end());

vector<int> v3(v1.size() + v2.size());

vector<int>::iterator it, end;

end = set\_intersection(v1.begin(), v1.end(), v2.begin(), v2.end(), v3.begin());

for (it = v3.begin(); it != end; it++)

if(\*it!=0)

{

cout << \*it <<endl;

}

cout << endl;

return 0;

}

Performance Index

`CSS Your code has Passed Execution!

istream& operator >>(istream& in,company& c)

{

in>>c.num;

for(int i=0;i<c.num;i++)

{

in>>c.s[i];

}

return in;

}

ostream& operator <<(ostream& out,company& c)

{

for(int i=0;i<c.num;i++)

{

out<<c.s[i]<<endl;

}

return out;

}

istream& operator >>(istream& in, employee& e)

{

in>>e.empid>>e.name>>e.project>>e.team>>e.bug>>e.onsite;

++e;

return in;

}

ostream& operator <<(ostream& out, employee& e)

{

out<<e.empid<<endl<<e.name<<endl<<e.pi;

return out;

}

void employee::operator ++()

{

pi = project+team+bug+onsite;

}

employee company::operator [](int i)

{

return s[i-1];

}

bool employee::operator<(employee& e)

{

if(pi>e.pi)

{

return 0;

}

else

{

return 1;

}

}

void company:: sort()

{

for(int i=0;i<num;i++)

{

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

{

if(s[i]<s[j])

{

employee temp;

temp = s[i];

s[i] = s[j];

s[j] = temp;

}

}

}

}