**WEEK-7**

**NAME-Sanchit Jain**

**Batch – B-7**

**Enroll No. – 21103192**

#include<iostream>

using namespace std;

class A{

    protected:

    float height;

    public:

    void getheight(){

        cout<<"Enter height :";

        cin>>height;

    }

};

class B{

    protected:

    float base;

    public:

    void getbase(){

        cout<<"Enter base :";

        cin>>base;

    }

};

class C : public A,public B{

    protected:

    float ar;

    public:

    void area(){

        getbase();

        getheight();

        ar =(base\*height)/2;

        cout<<"Area is : "<<ar;

    }

};

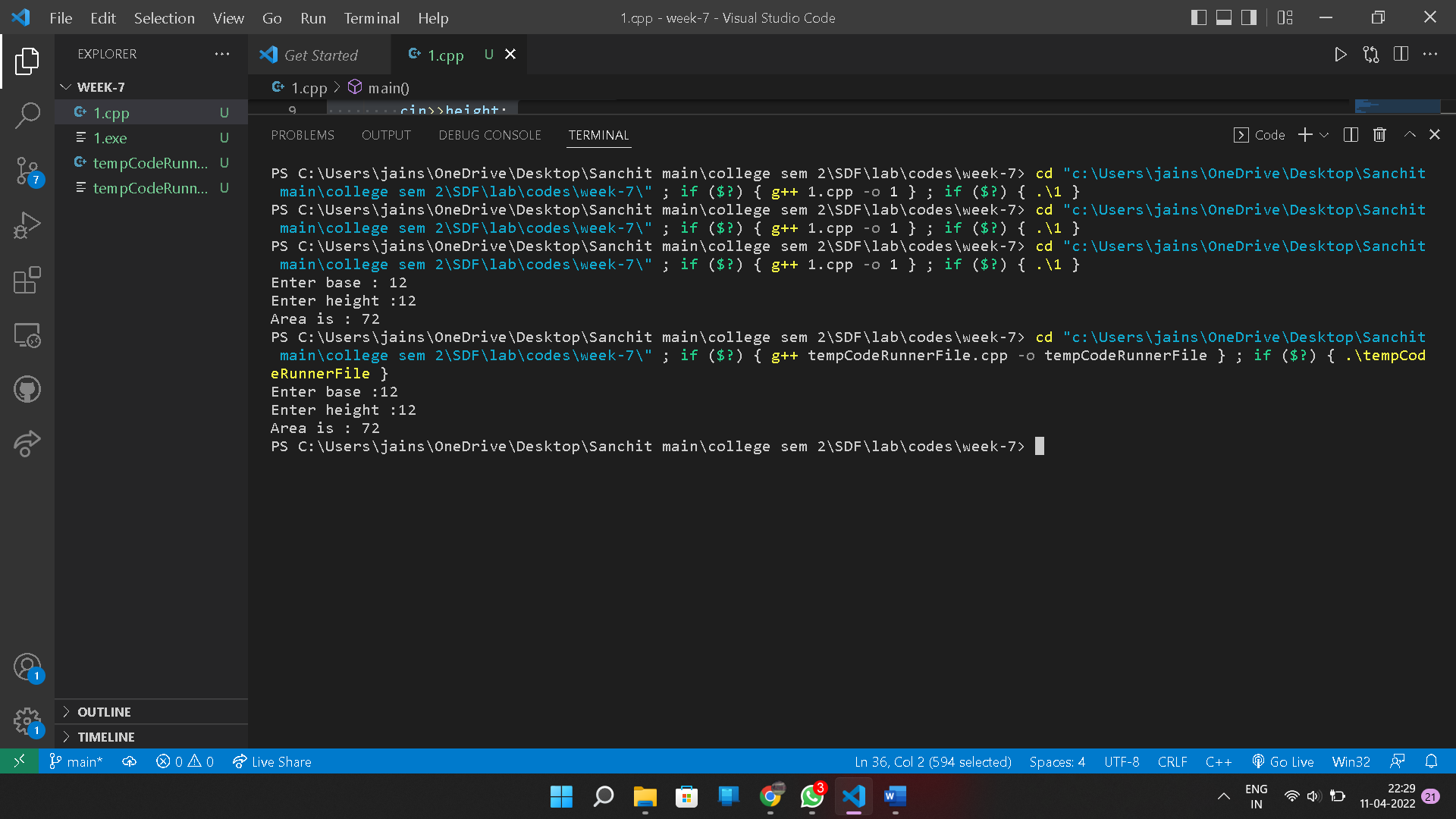
int main(){

    C obj;

    obj.area();

    return 0;

}



**2.** #include<iostream>

using namespace std;

class A{

    protected:

    string name;

    int enroll;

    public:

    void getdetails(){

        cout<<"Enter Name and Enroll :";

        cin>>name>>enroll;

        cout<<endl;

    }

};

class B{

    protected:

    int marks[5];

    public:

    void getmarks(){

        cout<<"Enter marks of 5 subjects :"<<endl;

        for(int i=0;i<5;i++){

            cout<<"Enter marks for "<<i<< "Subject :";

            cin>>marks[i];

        }

    }

};

class C : public A,public B{

    int sum;

    public:

    void all(){

        getdetails();

        getmarks();

        sum=0;

        for(int i=0;i<5;i++){

            sum +=marks[i];

        }

        cout<<"the sum is :"<<sum<<endl;

        cout<<"Your name is :"<<name<<" "<<"and enroll is "<<enroll;

    }

};

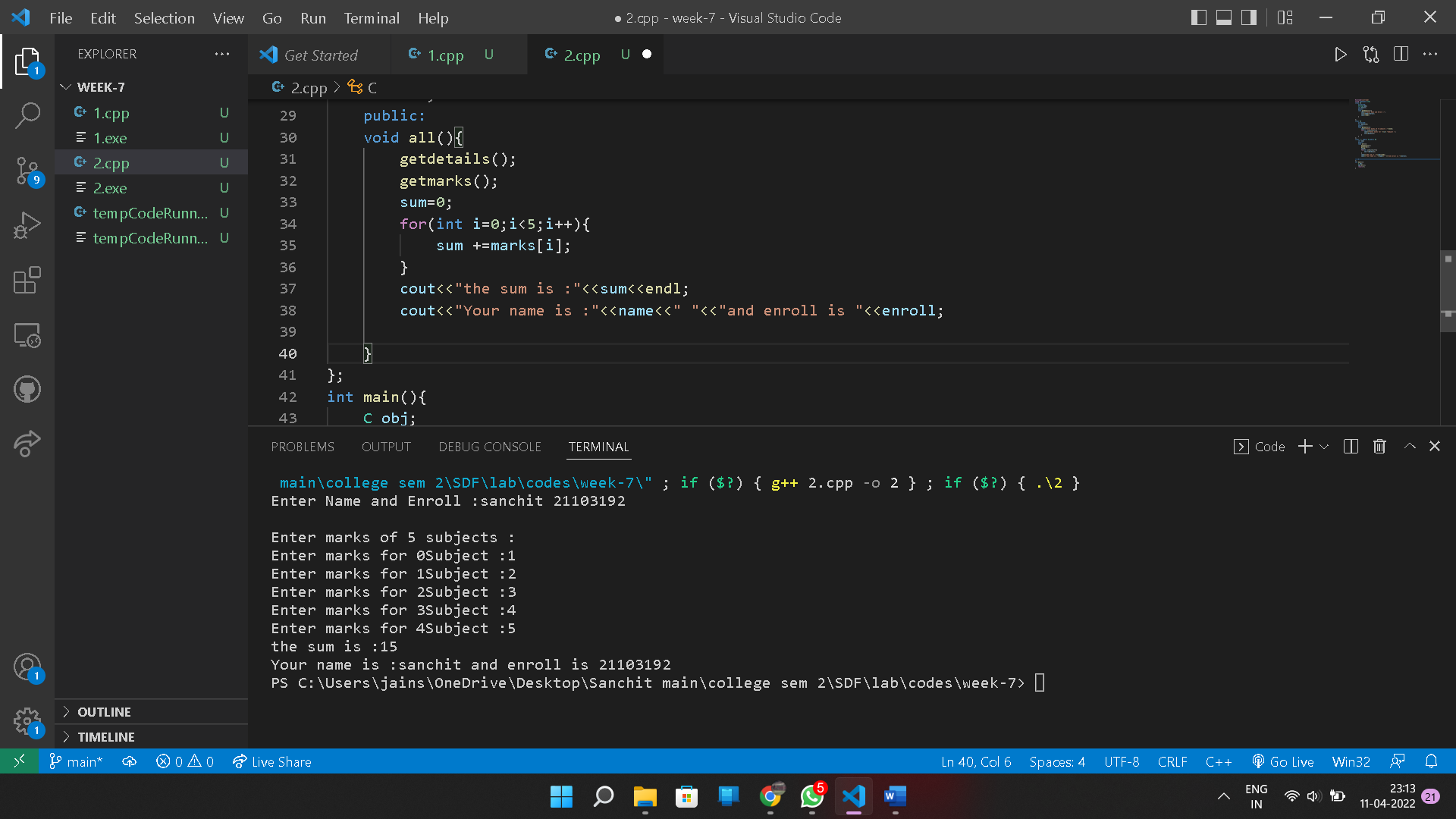
int main(){

    C obj;

    obj.all();

    return 0;

}



**3.** #include <iostream>

using namespace std;

class base

{

public:

    char fname[20];

    char surname[20];

public:

    virtual void calculate()

    {

        cout << "enter fname:";

        cin >> fname;

        cout << "enter surname";

        cin >> surname;

    }

    void display()

    {

        cout << "welcome" << fname << surname << endl;

    }

};

class derived : public base

{

public:

    void calculate()

    {

        cout << "enter derived\_fname:";

        cin >> fname;

        cout << "enter derived\_ surname";

        cin >> surname;

    }

    void display()

    {

        cout << "welcome to derived" << fname << surname << endl;

    }

};

int main()

{

    base \*b;

    derived d;

    b=&d;

    b->calculate();

    b->display();

    return 0;

}

**4.** #include <iostream>

using namespace std;

class base\_food\_items {

protected:

    char item\_name[20];

    int quantity;

    int item\_price;

public:

    virtual void order()

    {

    cout<< "enter item name:";

    cin>> item\_name;

    cout<< "enter quantity";

    cin>> quantity;

    cout<< "Item price";

    cin>> item\_price;

    }

    void total\_price()

    {

        cout<<"order is: " << item\_name<<"\t"<<"quantity:"<<quantity<<endl; cout << "total price=" << item\_price\*quantity<<endl;

    }

};

class Chinese: public base\_food\_items

{

public:

    void order()

    {

        cout<<"Welcome to Chinese Section !"<<endl;

        cout<< "enter item name:";

        gets(item\_name);

        cout<< "enter quantity";

        cin>> quantity;

        cout<< "Item price";

        cin>> item\_price;

    }

    void total\_price()

    {

        cout<<"order is: " << item\_name<<"\t"<<"quantity:"<<quantity<<endl; cout << "total price=" << item\_price\*quantity<<endl;

    }

};

int main()

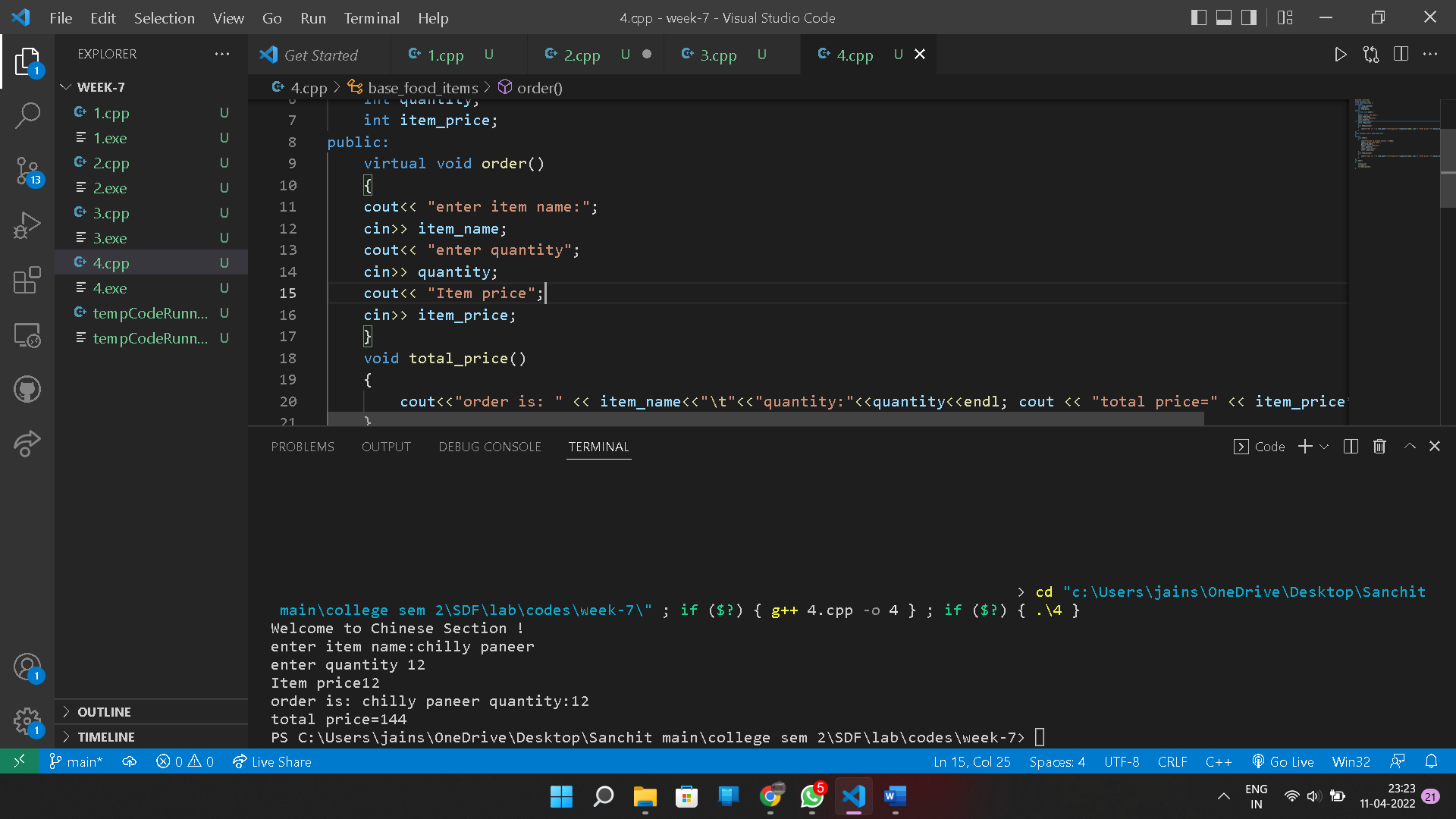
{

    Chinese c1;

    c1.order();

    c1.total\_price();

}



**5.** #include<iostream>

using namespace std;

class base{

    public:

    virtual void display1()=0;

    virtual void display2()=0;

};

class derived:public base{

    public:

    void display1(){

        cout<<"This is Display1() method of Derived Class"<<endl;

    }

    void display2(){

        cout<<"This is Display2() method of Derived Class"<<endl;

    }

};

int main(){

    base \*b;

    derived d;

    b=&d;

    b->display1();

    b->display2();

    return 0;

}

**6.** #include<iostream>

using namespace std;

class A{

    protected:

    int x;

    public:

    virtual void sum()=0;

    A(int i){

        x=i;

    }

};

class B : public A{

     protected:

     int y;

     public:

     B(int a,int b):A(b){

         y=a;

     }

     void sum(){

         cout<<"The sum is "<<x+y;

     }

};

int main(){

    int a,b;

    cout<<"Enter two numbers :"<<endl;

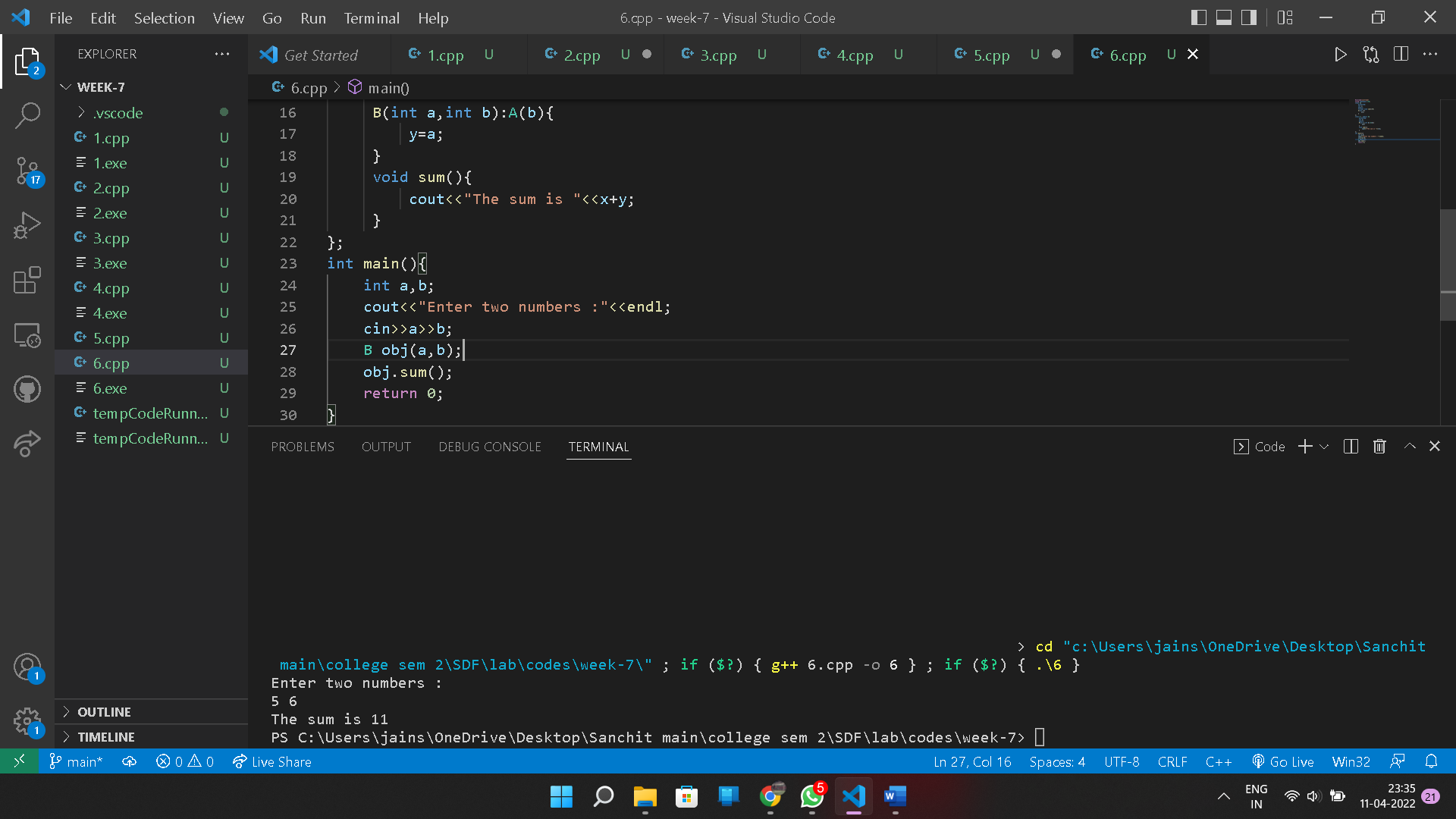
    cin>>a>>b;

    B obj(a,b);

    obj.sum();

    return 0;

}



**7. error: cannot dynamic\_cast ‘base\_ptr’ (of type ‘class Base\*’) to type ‘class Derived\*’ (source type is not polymorphic)**

#include<iostream>

using namespace std;

class Base

{

    public:

    virtual void print()

    {

        cout<<"In Base";

    }

};

class Derived: public Base

{

    public:

    void print()

    {

        cout<<"In Derived";

    }

};

int main()

{

Base \*base\_ptr = new Derived;

Derived \*derived\_ptr;

derived\_ptr = dynamic\_cast<Derived\*>(base\_ptr);

if(derived\_ptr != NULL)

{

    cout<<"It is working";

}

else

{

    cout<<"cannot cast Base\* to Derived\*";

}

return 0;

}