Q1

#include<iostream>

using namespace std;

class a

{

    protected:

    float base;

    public:

    void getbase()

    {

        cout<<"enter base \n";

        cin>>base;

    }

};

class b

{

     protected:

    float height;

    public:

    void getheight()

    {

        cout<<"enter height \n";

        cin>>height;

    }

};

class c:public a,public b

{

    float area;

    public:

    void area1(){

        area=height\*base\*0.5;

        cout<<"the area is : "<<area;

    }

};

int main()

{

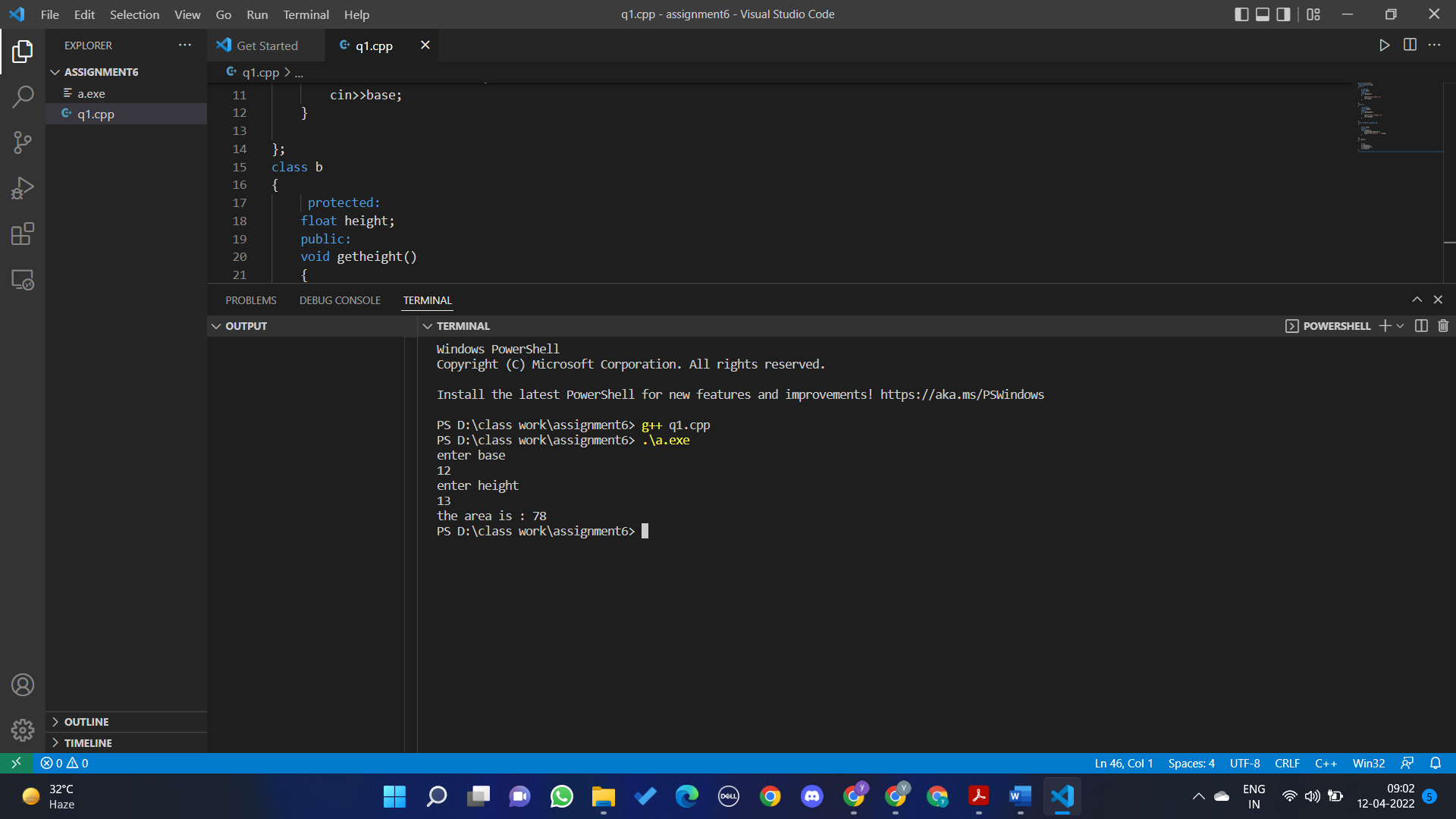
    c t1;

    t1.getbase();

    t1.getheight();

    t1.area1();

}



Q2

#include<iostream>

using namespace std;

class studentdetail{

    protected:

    string name;

    string enno;

    public:

    void detail(){

        cout<<"enter the student name : \n";

        cin>>name;

        cout<<"enter the enrollment no: \n";

        cin>>enno;

    }

};

class marks{

    protected:

    float marks[5];

    public:

    void getmarks()

    {

        for(int i=0;i<5;i++)

        {

            cout<<"enter the value of marks ["<<i<<"] ";

            cin>>marks[i];

        }

    }

};

class c:public studentdetail,public marks{

    float total;

    public:

    void sum()

    {

        float t=0;

        for(int i=0;i<5;i++)

        {

            t+=marks[i];

        }

        total=t;

        cout<<"TOTAL = "<<total;

    }

};

int main()

{

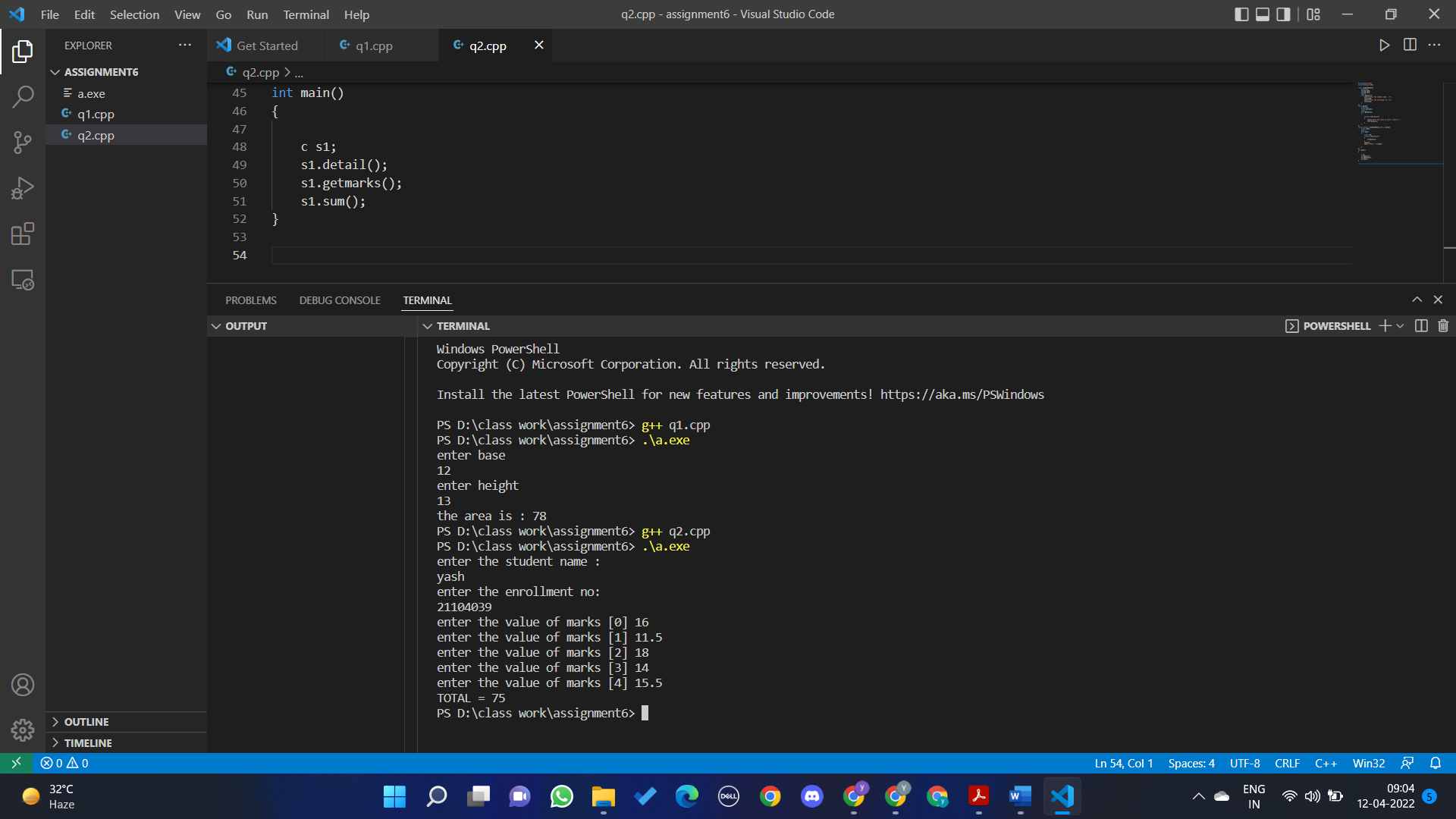
    c s1;

    s1.detail();

    s1.getmarks();

    s1.sum();

}



Q3

#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;

    }

    virtual 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()

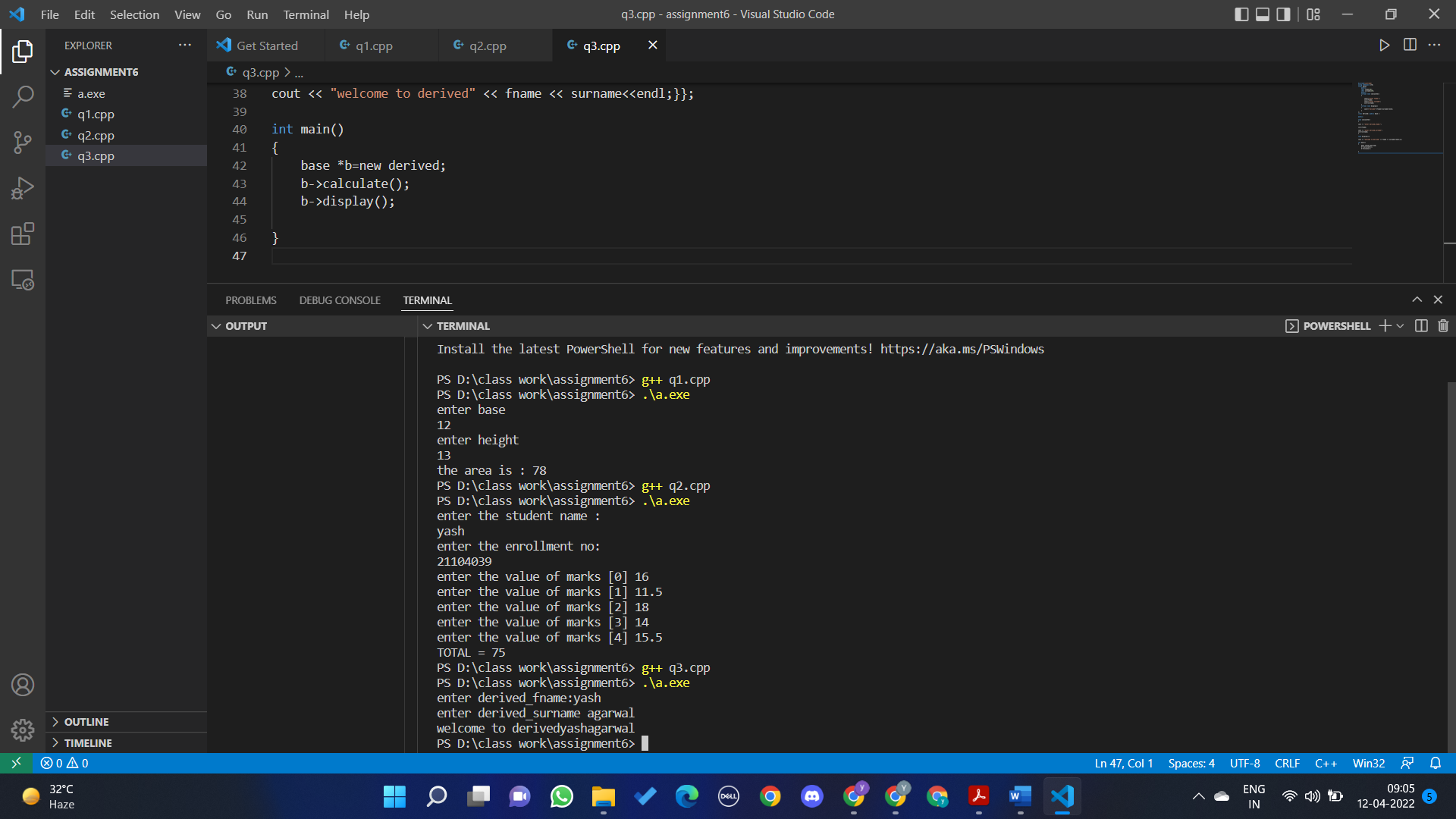
{

    base \*b=new derived;

    b->calculate();

    b->display();

}



Q4

#include <iostream>

using namespace std;

class base\_food\_items {

public:

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{};

int main()

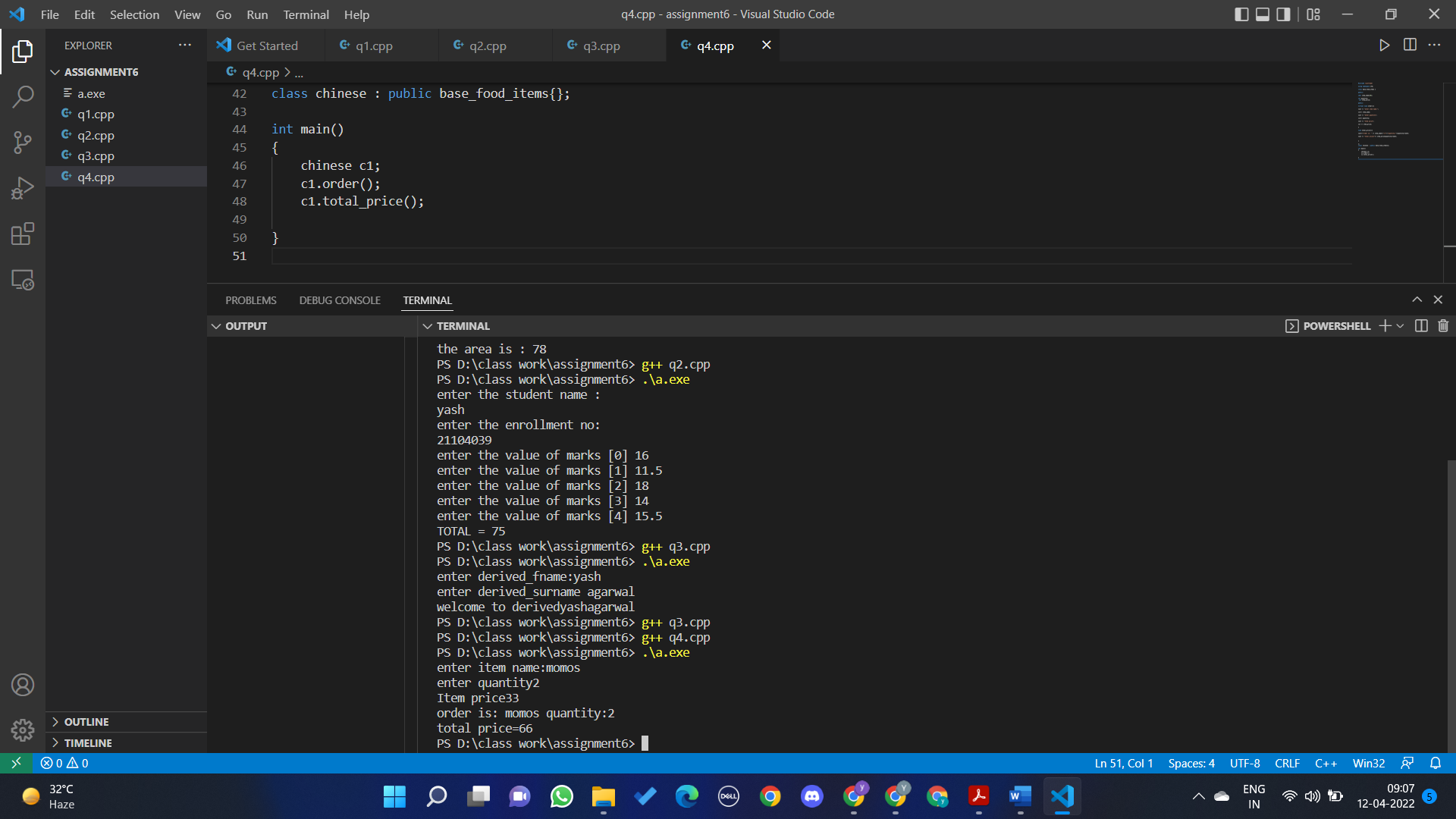
{

    chinese c1;

    c1.order();

    c1.total\_price();

}



Q5

#include <iostream>

using namespace std;

class a

{

    public:

    virtual void display1()=0;

    virtual void display2()=0;

};

class b:public a

{

    public:

    void display1(){

        cout<<"this is display 1 method of derived class\n";

    }

    void display2(){

        cout<<"this is display 2 method of derived class";

    }

};

int main()

{

    a \*a1 =new b;

    a1->display1();

     a1->display2();

}

Q6

#include <iostream>

using namespace std;

class a

{

    protected:

    int s1,s2,total;

    public:

    a()

    {

        cin>>s1>>s2;

        total=s1+s2;

    }

    virtual void display1()=0;

};

class b:public a

{

    public:

    b():a(){};

    void display1(){

        cout<<"the sum is "<<total;

    }

};

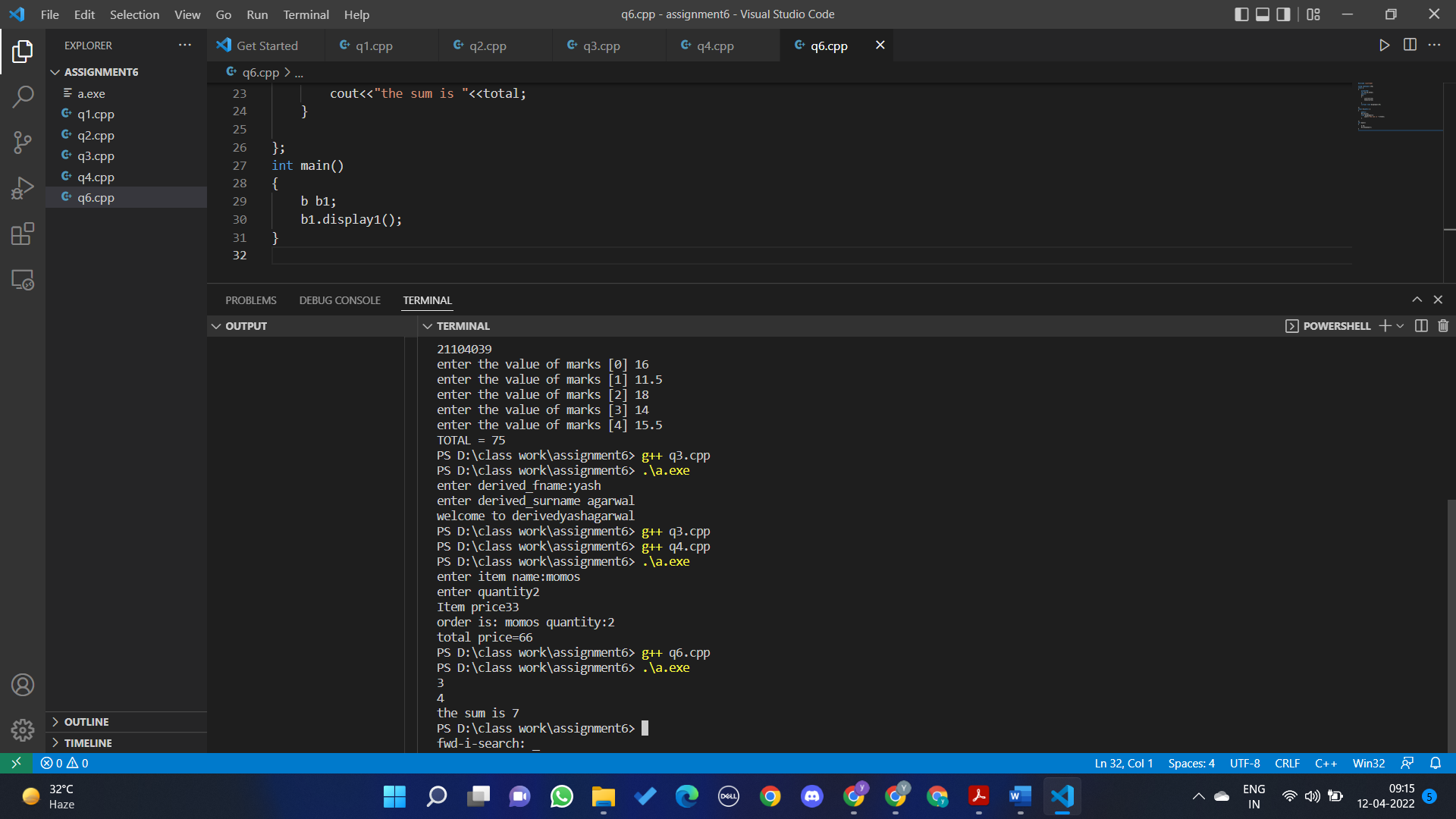
int main()

{

    b b1;

    b1.display1();

}



Q7

1. Error, because run time polymorphism does not occur due to absence of virtual variables,

#include<iostream>

using namespace std;

class Base {

public :

virtual ~Base(){

}

};

class Derived: public Base {

};

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;}