

1)positive negative and zero:

#include<iostream>

using namespace std;

int main()

{

int n;

cout<<"enter number";

cin>>n;

if(n>0)

{

cout<<"positive number:"<<n;

}

else

if(n<0)

{

cout<<"negative number:"<<n;

}

else

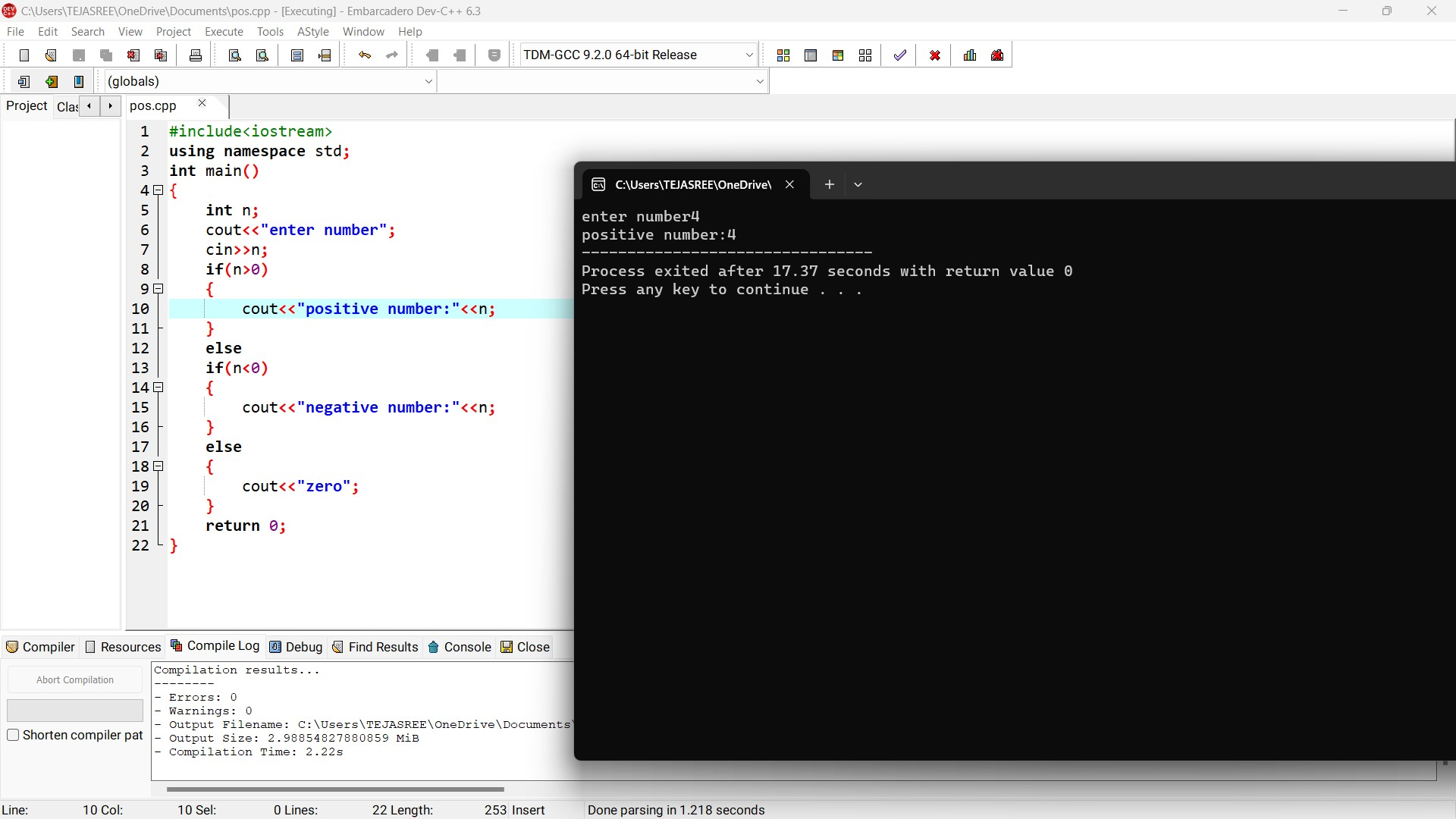
{

cout<<"zero";

}

return 0;

}



2)greatest common divisor:

#include <iostream>

using namespace std;

int main() {

int n1,n2;

cout << "Enter two numbers: ";

cin>>n1>>n2;

while(n1 != n2) {

if(n1>n2)

n1=n1-n2;

else

n2=n2-n1;

}

cout<<"gcd"<<n1;

return 0;

}



3)pattern:

#include<iostream>

using namespace std;

int main()

{

int n,i,j;

cout<<"enter number:";

cin>>n;

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

{

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

{

cout<<"\*";

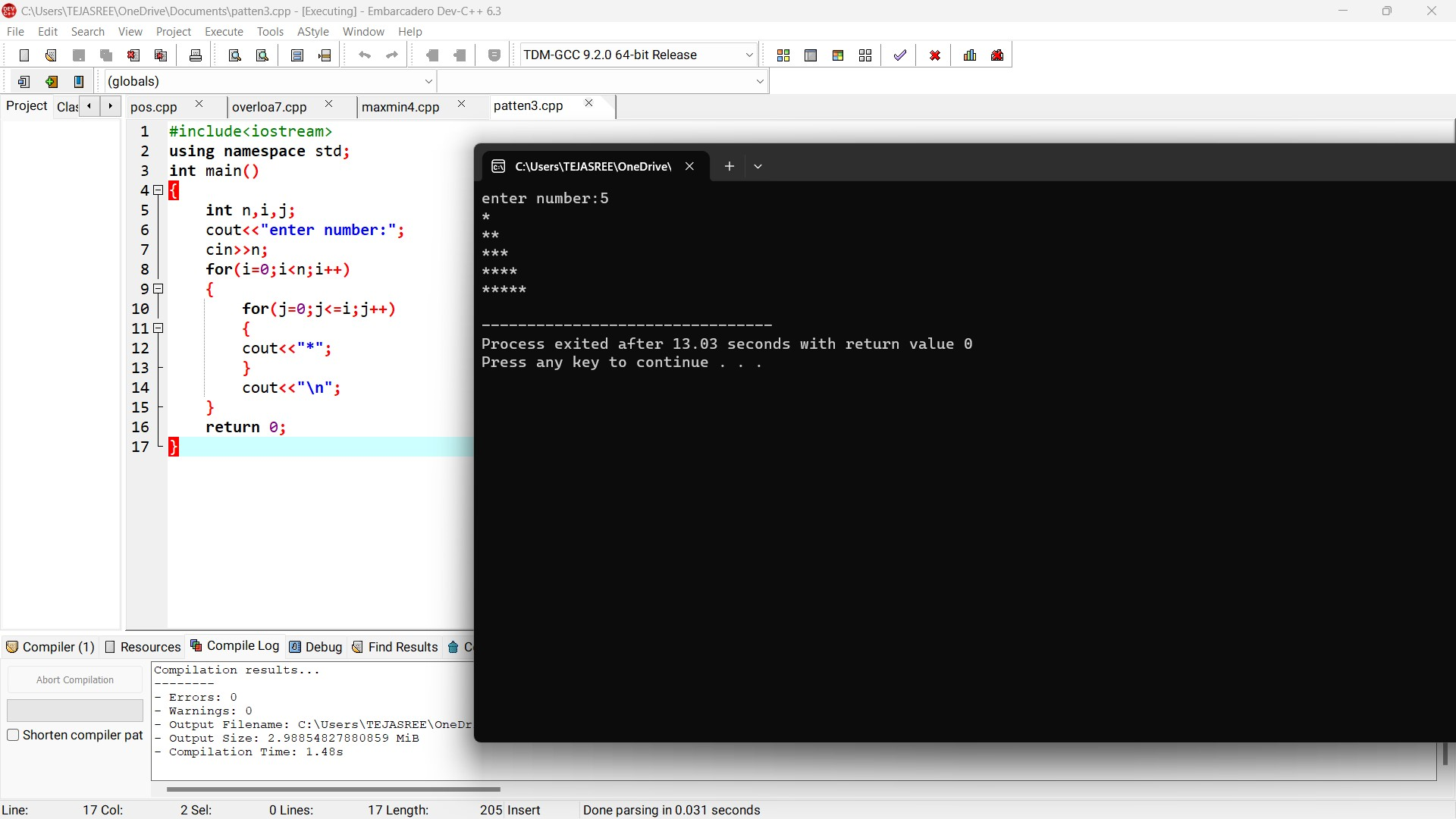
}

cout<<"\n";

}

return 0;

}



4)maximum and minimum number:

#include<iostream>

using namespace std;

int main()

{

int arr[100];

int i, max, min, size;

cout<<"Enter size of the array: ";

cin>>size;

cout<<"Enter elements in the array: ";

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

{

cin>>arr[i];

}

max = arr[0];

min = arr[0];

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

{

if(arr[i] > max)

{

max = arr[i];

}

if(arr[i] < min)

{

min = arr[i];

}

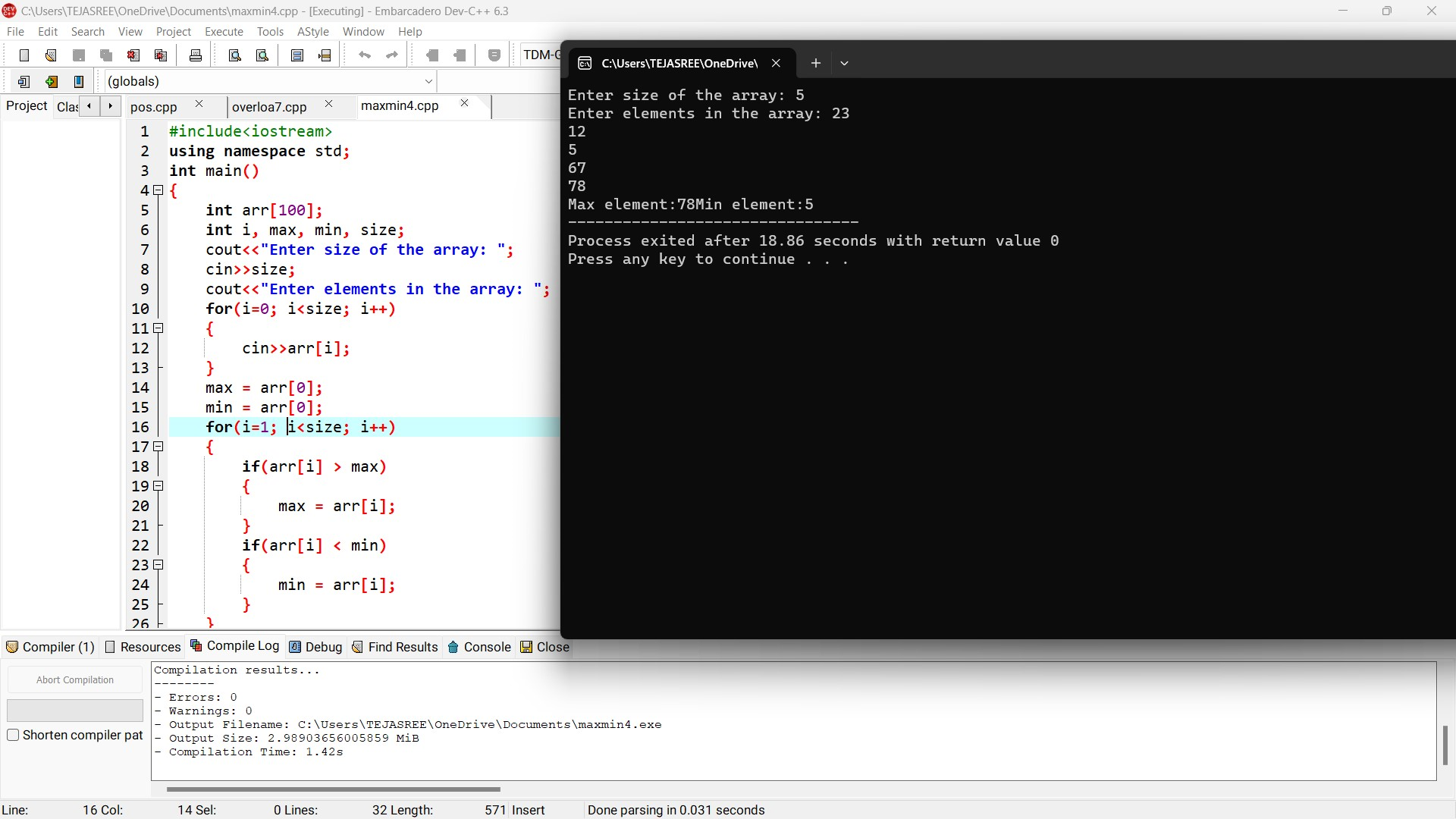
}

cout<<"Max element:"<<max;

cout<<"Min element:"<<min;

return 0;

}



5)vehicle class:

#include<iostream>

using namespace std;

class vehicle

{

public:

int types;

void set()

{

cout<<"enetr types of vehicles:";

cin>>types;

}

virtual void drive()

{

cout<<types<<endl;

}

};

class car:public vehicle

{

public:

int drivers;

string name;

void set1()

{

cout<<"enter no.of drivers:";

cin>>drivers;

cout<<"enetr name of driver:";

cin>>name;

}

void drive(){

cout<<drivers<<endl;

cout<<name<<endl;

}

};

class truck:public vehicle

{

public:

int cost;

string truckname;

void set2()

{

cout<<"enter cost:";

cin>>cost;

cout<<"enetr name of truck:";

cin>>truckname;

}

void drive(){

cout<<cost<<endl;

cout<<truckname<<endl;

}

};

int main()

{

vehicle v;

v.set();

v.drive();

car c;

c.set1();

c.drive();

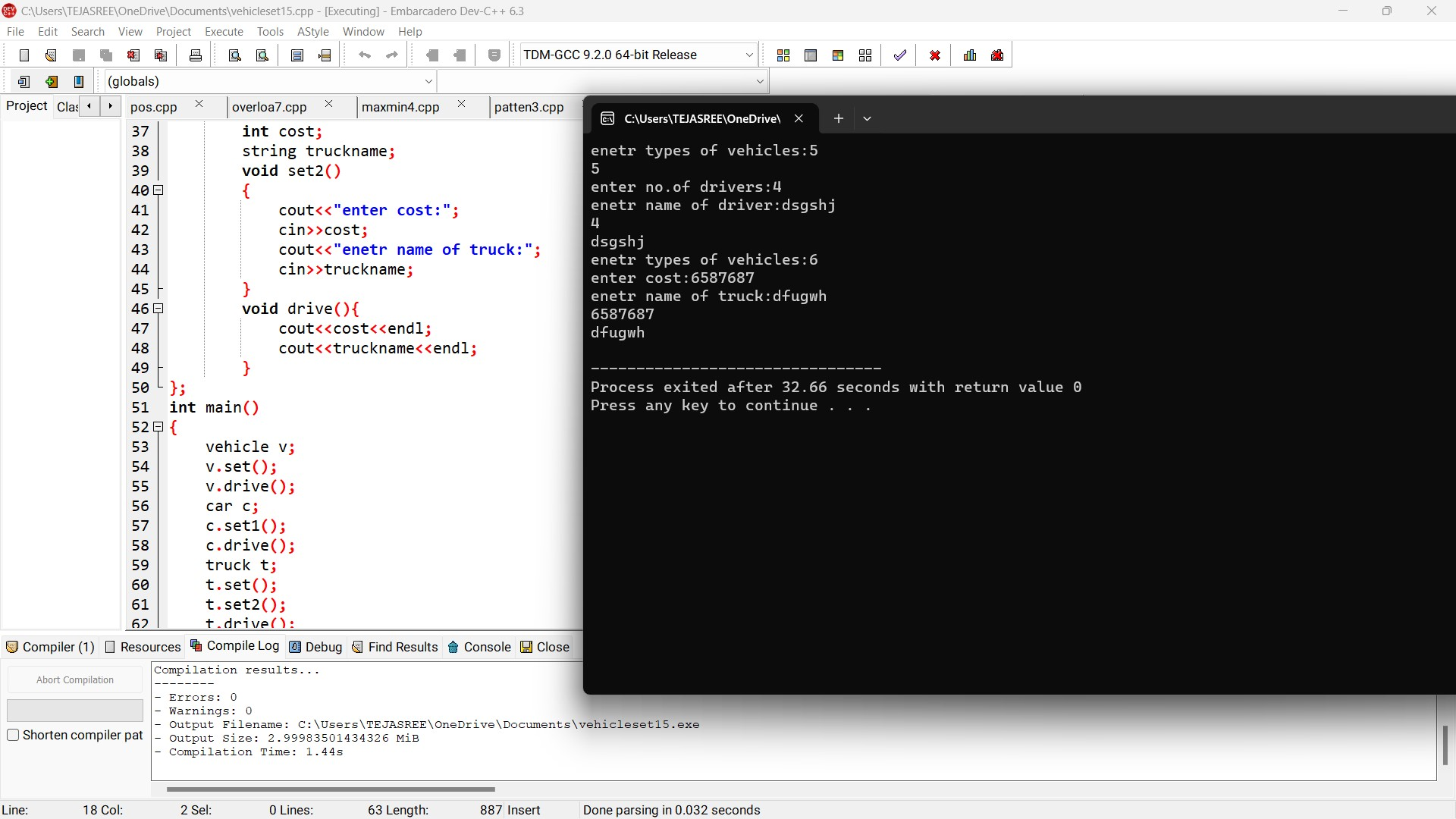
truck t;

t.set();

t.set2();

t.drive();

}



6)person ,student and teacher class:

#include<iostream>

using namespace std;

class person

{

public:

int age;

string gender,name;

void set()

{

cout<<"enetr age:";

cin>>age;

cout<<"enetr gender:";

cin>>gender;

cout<<"enetr name:";

cin>>name;

}

void display()

{

cout<<age<<endl;

cout<<gender<<endl;

cout<<name<<endl;

}

};

class student:public person

{

public:

int roll\_no,cls;

void set1()

{

cout<<"enter roll number:";

cin>>roll\_no;

cout<<"enetr class:";

cin>>cls;

}

void display1(){

cout<<roll\_no<<endl;

cout<<cls<<endl;

}

};

class teacher:public person

{

public:

int sal;

string subject;

void set2()

{

cout<<"enter salary:";

cin>>sal;

cout<<"enetr subject:";

cin>>subject;

}

void display2(){

cout<<sal<<endl;

cout<<subject<<endl;

}

};

int main()

{

person p;

p.set();

p.display();

student s;

s.set();

s.display();

s.set1();

s.display1();

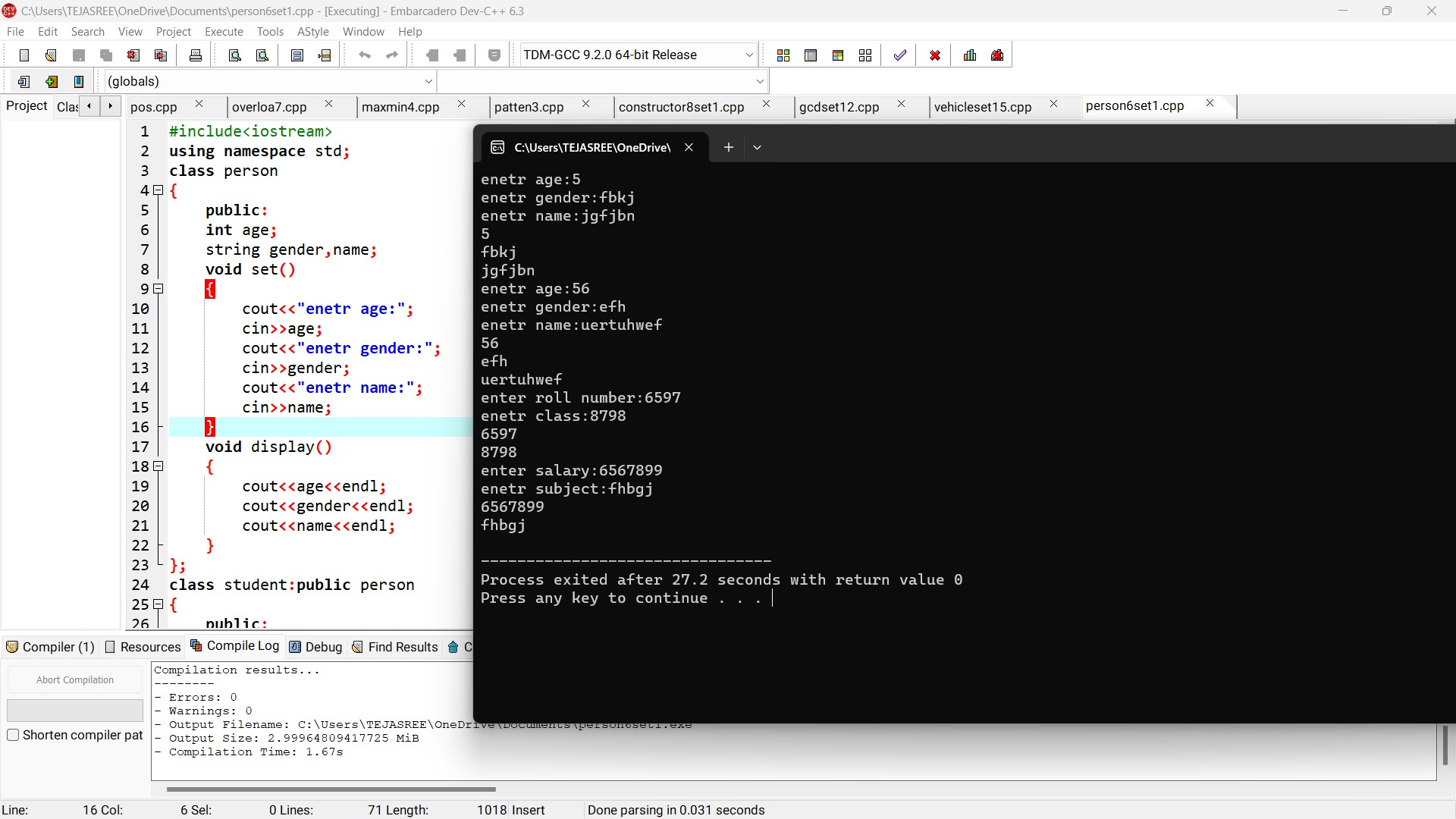
teacher t;

t.set2();

t.display2();

return 0;

}



7)operator overload +:

#include<iostream>

using namespace std;

class A

{

public:

int a,b;

void set()

{

cout<<"enetr num1";

cin>>a;

cout<<"enter num2";

cout<<b;

}

void display()

{

cout<<a<<endl;

cout<<b<<endl;

}

A operator+()

{

cout<<a+b;

}

};

int main()

{

A a1;

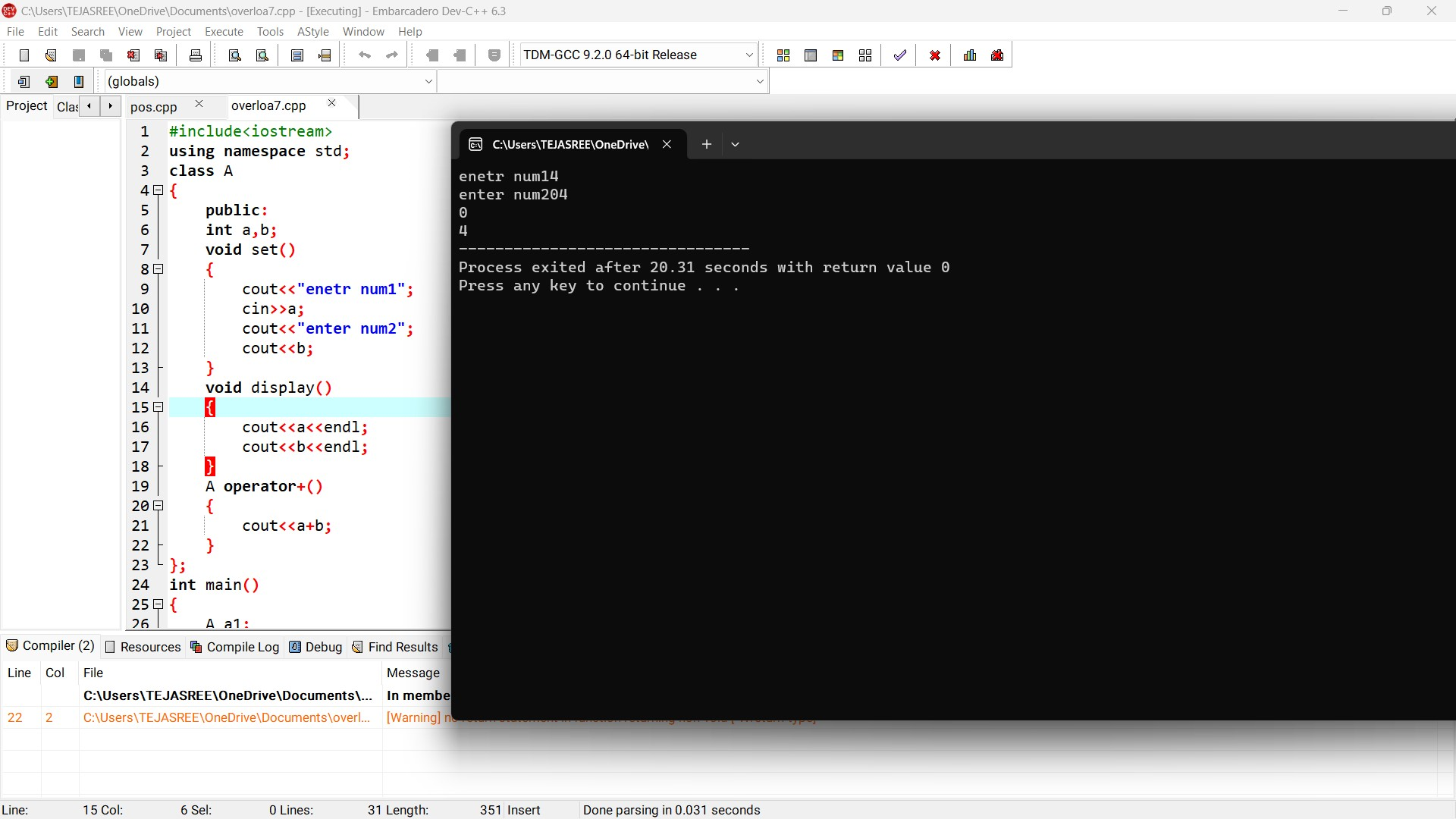
a1.set();

a1.display();

cout<<a1.a+a1.b;

return 0;

}



8)book constructor and destructor:

#include<iostream>

using namespace std;

class book

{

public:

int types,cost;

string name,author;

book()

{

cout<<"enter no.of types:";

cin>>types;

cout<<"enetr cost:";

cin>>cost;

cout<<"enetr name:";

cin>>name;

cout<<"enetr author name:";

cin>>author;

}

void display(){

cout<<types<<endl;

cout<<cost<<endl;

cout<<name<<endl;

cout<<author<<endl;

}

~book()

{

cout<<"operator destructor";

}

};

int main()

{

book b;

b.display();

return 0;

}

