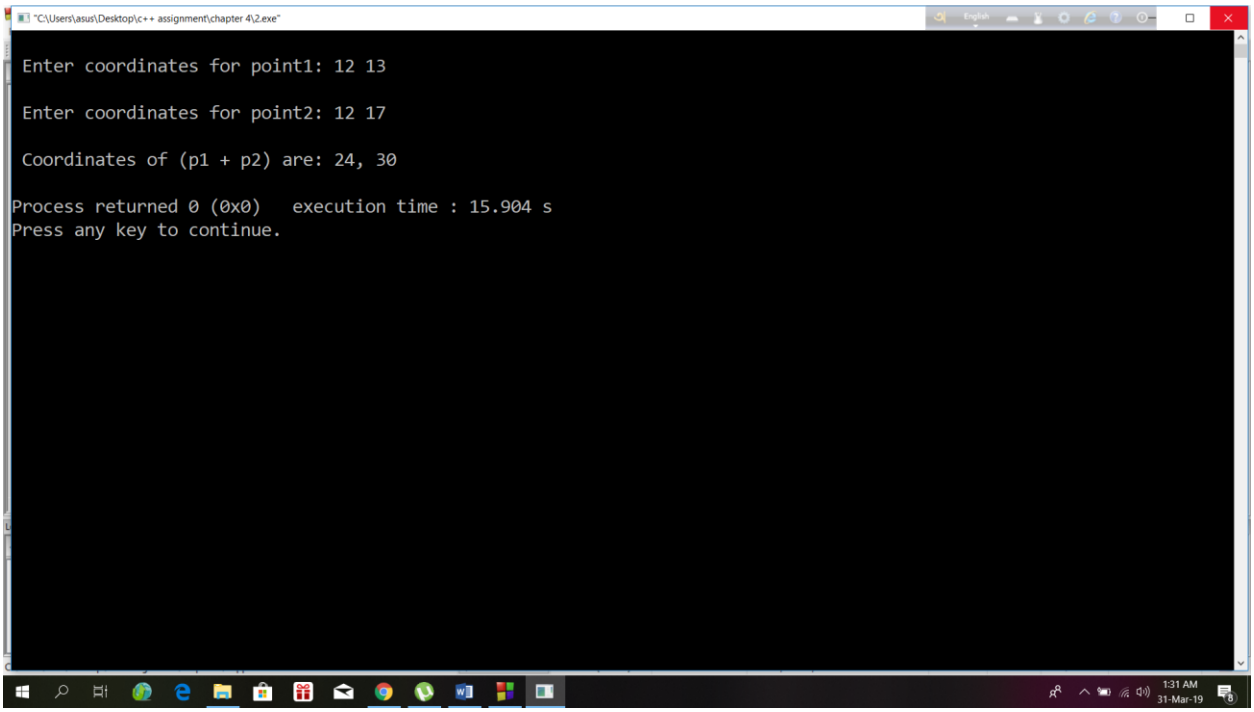
Chapter 4

Exercise 2:

```
#include <iostream>
#include<iomanip>
using namespace std;
struct point
{
   int xCo;
   int yCo;
};

int main()
{
   point p1, p2, p3;
   cout << " \n Enter coordinates for point1: ";
   cin >> p1.xCo >> p1.yCo;
   cout << "\n Enter coordinates for point2: ";
   cin >> p2.xCo >> p2.yCo;
   p3.xCo = p1.xCo + p2.xCo;
   p3.yCo = p1.yCo + p2.yCo;
   cout << "\n Coordinates of (p1 + p2) are: "
        << p3.xCo << ", " << p3.yCo << endl;
   return 0;
```

}

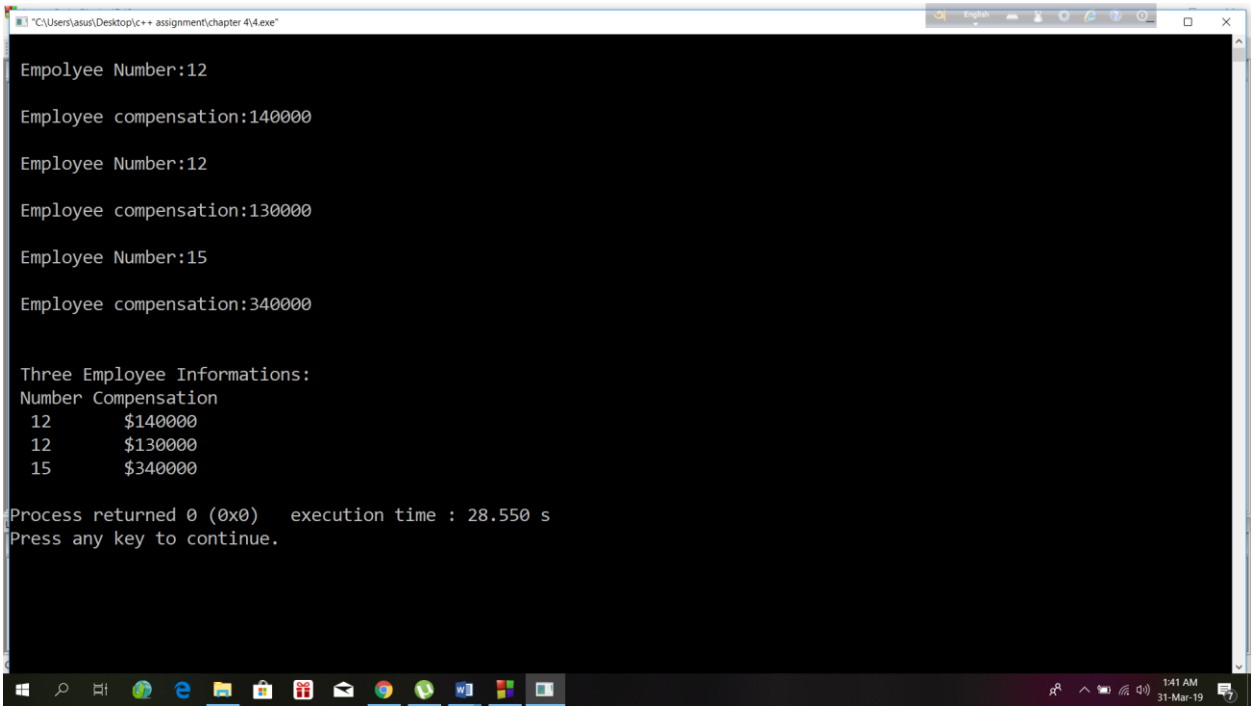


Exercise 4:

```
#include<iostream>
#include<iomanip>
struct emplyee
{
   int emplyee_number;
   float compensation;
};
using namespace std;
int main()
{
   emplyee emplyee1,emplyee2,emplyee3;
   cout<< "\n Empolyee Number:";
```

```
cin>>emplyee1.emplyee_number;
cout<< "\n Employee compensation:";
cin>>emplyee1.compensation;
cout<< "\n Employee Number:";
cin>>emplyee2.emplyee_number;
cout<< "\n Employee compensation:";
cin>>emplyee2.compensation;
cout<< "\n Employee Number:";
cin>>emplyee3.emplyee_number;
cout<< "\n Employee compensation:";
cin>>emplyee3.compensation;

cout<<setw(6)<< "\n\n Three Employee Informations:"<<endl;
cout<<setw(4)<< " Number"<<" "<<setw(4)<< "Compensation"<<endl;

cout<<setw(4)<<emplyee1.emplyee_number<< " "
   <<setw(7)<< "$"<<emplyee1.compensation<<endl;
cout<<setw(4)<<emplyee2.emplyee_number<< " "
   <<setw(7)<<"$"<<emplyee2.compensation<<endl;
cout<<setw(4)<<emplyee3.emplyee_number<< " "
   <<setw(7)<<"$"<<emplyee3.compensation<<endl;
return 0;
```

}

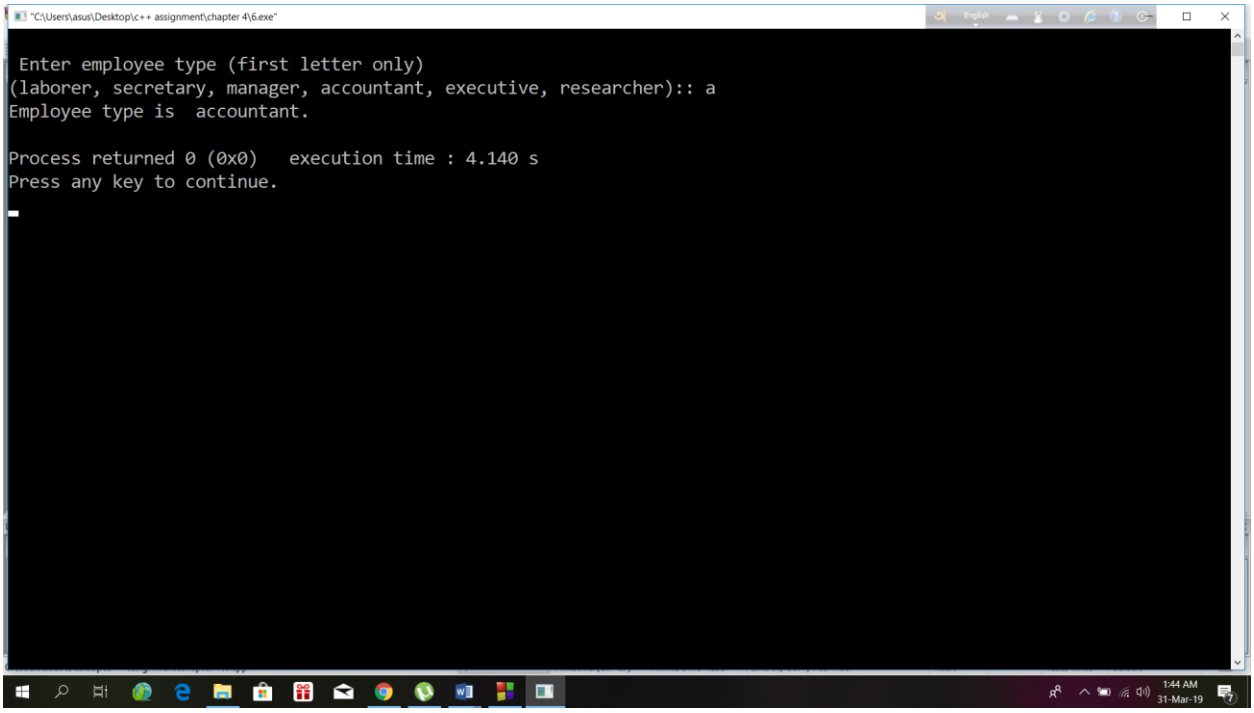


Exercise 6:

```
#include<iostream>
#include<iomanip>
#include<conio.h>
enum etype {laborer, secretary, manager, accountant, executive, researcher};
using namespace std;
int main()
{
   etype x;
   char *ret;

   cout<<"\n Enter employee type (first letter only)"<<endl
```

```
   <<"(laborer, secretary, manager, accountant, executive, researcher):: ";

switch(getche())
{
case 'l':
   x=laborer   ;
   break;
case 's':
   x=secretary ;
   break;
case 'm':
   x=manager   ;
   break;
case 'a':
   x=accountant;
   break;
case 'e':
   x=executive ;
   break;
case 'r':
   x=researcher;
}

switch(x)
```

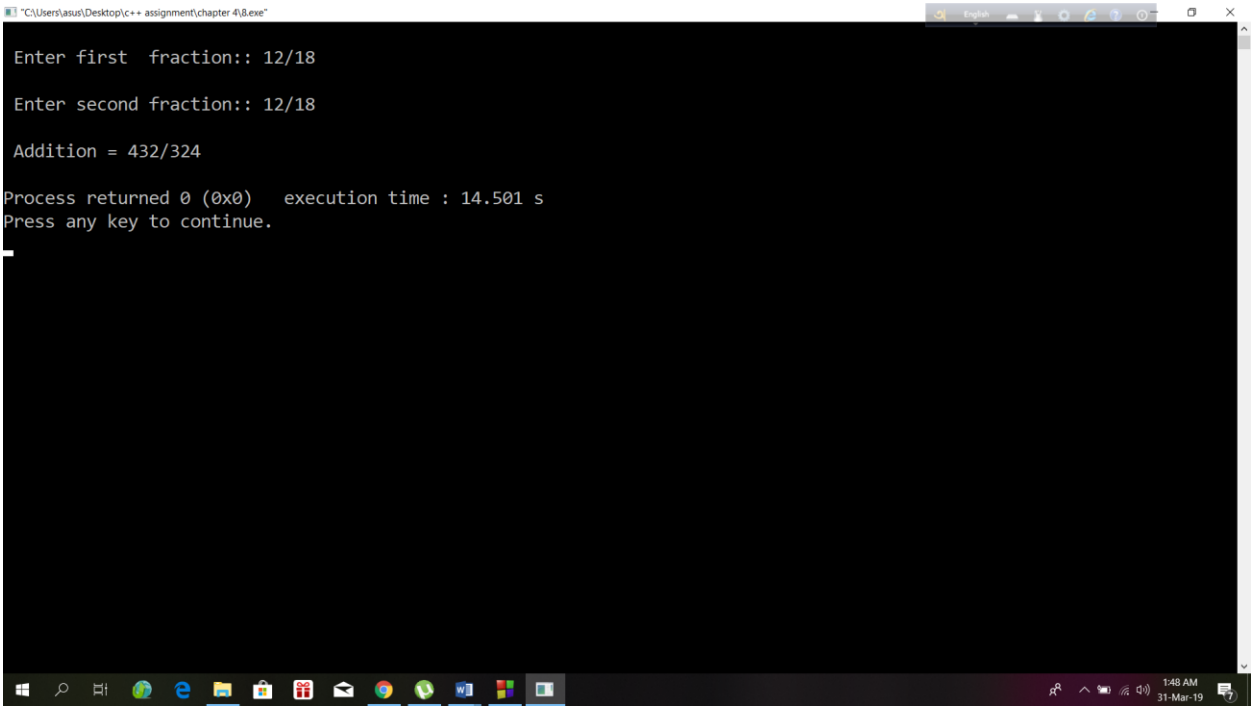
```
    {
    case 0:
       ret = " laborer"   ;
       break;
    case 1:
       ret = " secretary" ;
       break;
    case 2:
       ret = " manager"   ;
       break;
    case 3:
       ret = " accountant";
       break;
    case 4:
       ret = " executive" ;
       break;
    case 5:
       ret = " researcher";
    }
    cout<<"\nEmployee type is "<<ret<<"."<<endl;

    return 0;
}
```


"C:\Users\asus\Desktop\c++ assignment\chapter 4\6.exe"
Enter employee type (first letter only)
(laborer, secretary, manager, accountant, executive, researcher):: a
Employee type is accountant.
Process returned 0 (0x0)   execution time : 4.140 s
Press any key to continue.
1:44 AM
31-Mar-19

Exercise 8:

```
#include<iostream>
#include<conio.h>
struct fraction
{int numerator;
 int denominator;
 };

using namespace std;
int main()
{

 fraction equation[2];
 char Operator;

 cout<<"\n Enter first  fraction:: ";
 cin >>equation[0].numerator>>Operator>>equation[0].denominator;

 cout<<"\n Enter second fraction:: ";
 cin >>equation[1].numerator>>Operator>>equation[1].denominator;

 cout<<"\n Addition = "
   <<(equation[0].numerator*equation[1].denominator +
equation[0].denominator*equation[1].numerator)
   <<Operator<<(equation[0].denominator*equation[1].denominator)<<endl;
```

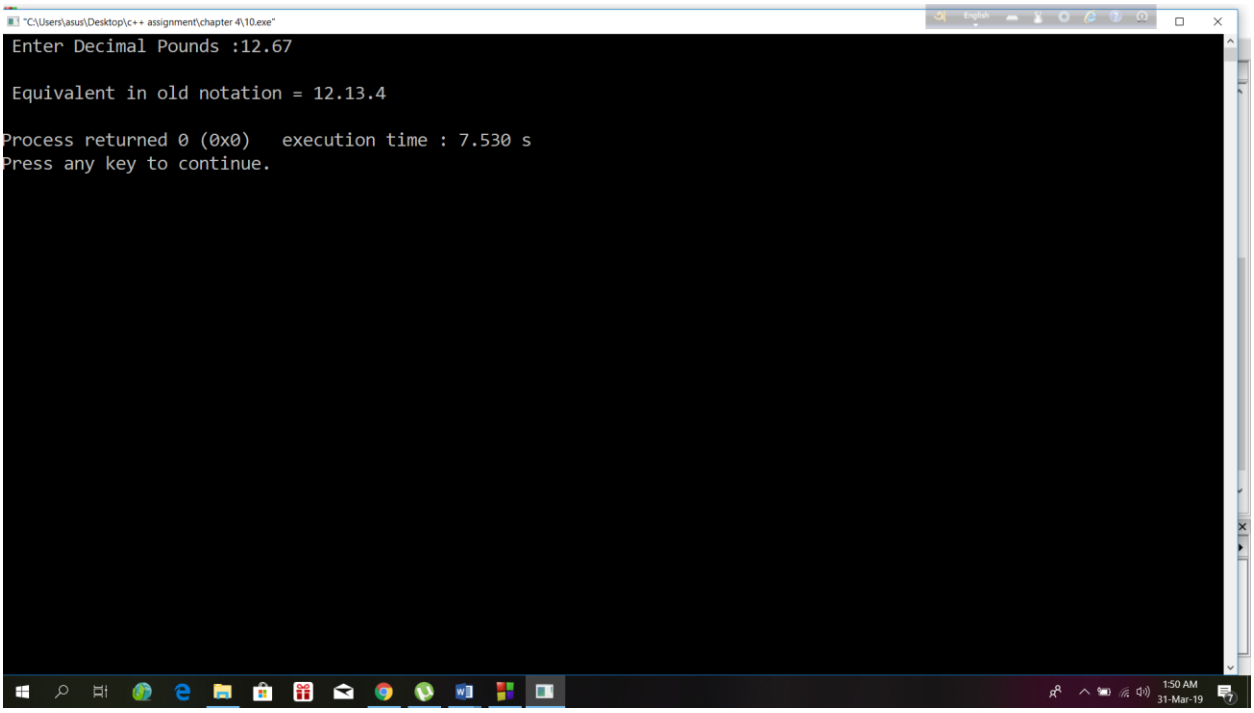
```
 return 0;

}
```



Exercise 10:

```
#include<iostream>
#include<iomanip>
struct sterling
{
   int pounds;
```

```
    int sillings;
    int pence;
};

using namespace std;
int main()
{
    sterling a;
    double decPounds,decFraction,temp;
    cout<< " Enter Decimal Pounds :";
    cin>> decPounds;
    cout<<endl;

    a.pounds = static_cast<int>(decPounds);
    decFraction = decPounds - a.pounds;
    temp =decFraction*20;
    a.sillings = decFraction*20;
    a.pence = (temp-a.sillings)*12;
    cout<< " Equivalent in old notation = "
        <<a.pounds<< "."<<a.sillings<< "."<<a.pence<<endl ;
    return 0;
}
```



Exercise 12:

```
#include<iostream>
#include<iomanip>
struct fraction
{
   int numerator;
   int denominator;
};

using namespace std;
int main()
{
```

```
fraction f[2];
char sign, Operator;
char ch='y';
cout<<endl;
cout<< "    ========================================"<<endl;
cout<< "         Enter The Inputs Like Below<"<<endl;
cout<< "    ========================================"<<endl;
cout<< "       Addition      : a/b + c/d    "<<endl;
cout<< "       Subtraction   : a/b - c/d    "<<endl;
cout<< "       Multiplication: a/b * c/d    "<<endl;
cout<< "       Division      : a/b / c/d    "<<endl;
cout<< "    ========================================"<<endl;
cout<< "    ========================================"<<endl;
do
{


   cout<<" Write your Expression : ";


   cin
>>f[0].numerator>>sign>>f[0].denominator>>Operator>>f[1].numerator>>sign>>f
[1].denominator;


   if(Operator=='+')
   {
```

```
            cout<<"Addition = "<<(f[0].numerator*f[1].denominator +
f[0].denominator*f[1].numerator)<<sign<<(f[0].denominator*f[1].denominator);
            }


            if(Operator=='-')
            {
               cout<<"Subtraction = "<<(f[0].numerator*f[1].denominator -
f[0].denominator*f[1].numerator)<<sign<<(f[0].denominator*f[1].denominator);
            }
            if(Operator=='*')
            {
               cout<<"Multip[lication =
"<<(f[0].numerator*f[1].denominator)<<sign<<
(f[0].denominator*f[1].denominator);
            }


            if(Operator=='/')
            {
               if(f[0].denominator != 0 &&f[1].numerator!=0 )
               {
                  cout<<"Division = "<<(f[0].numerator*f[1].denominator)<<sign<<
(f[0].denominator*f[1].numerator);
               }


               else
               {
```

```
            cout<<"Math error !"<<endl;
          }
        }

        cout<< "\n Do you wish to continue y/n ::";
        cin>>ch;
    }
    while(ch!='n');
    return 0;
}
```