Module Code: COS1512

Assessment: Assignment 3

Student Number: 69234175

Name: Jaymeen Patel

Unique Number: 170001

Due Date: 22/08/2022

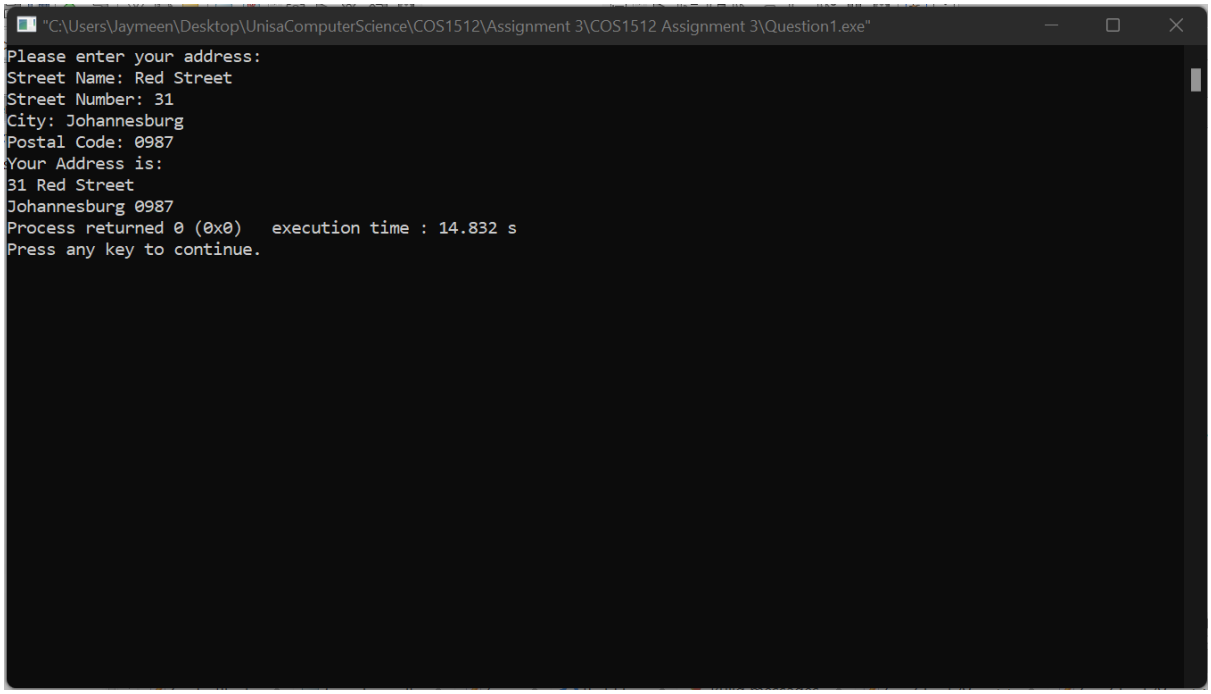
Question 1

```
#include <iostream>

using namespace std;

class Address
{
   public:

      Address()
      {
        streetName = " ";
        streetNr = 0;
        city = " ";
        postalCode = "0000";
      }

      void setStreetname(string sName)
      {
        streetName = sName;
      }
      void setStreetnum(int iNum)
```

```
    {
      streetNr = iNum;
    }
    void setCity(string sCity)
    {
      city = sCity;
    }
    void setPostalcode(string sCode)
    {
      postalCode = sCode;
    }

    string getStreetname()
    {
       return streetName;
    }
    int getStreetnum()
    {
       return streetNr;
    }
    string getCity()
    {
       return city;
    }
    string getPostalcode()
    {
       return postalCode;
    }

private:
    string streetName;
```

```
        int streetNr;
        string city;
        string postalCode;

};

int main()
{
    Address Address1;
    string strName;
    int strNr;
    string sCity;
    string pCode;

    cout << "Please enter your address: " << endl;

    cout << "Street Name: ";
    getline(cin, strName, '\n');
    Address1.setStreetname(strName);

    cout << "Street Number: ";
    cin >> strNr;
    Address1.setStreetnum(strNr);

    cout << "City: ";
    cin >> sCity;
    Address1.setCity(sCity);

    cout << "Postal Code: ";
    cin >> pCode;
    Address1.setPostalcode(pCode);
```

```
    cout << "Your Address is: " << endl;
    cout << Address1.getStreetnum() << " " << Address1.getStreetname() << endl;
    cout << Address1.getCity() << " " << Address1.getPostalcode();



    return 0;
}
```



Question 2

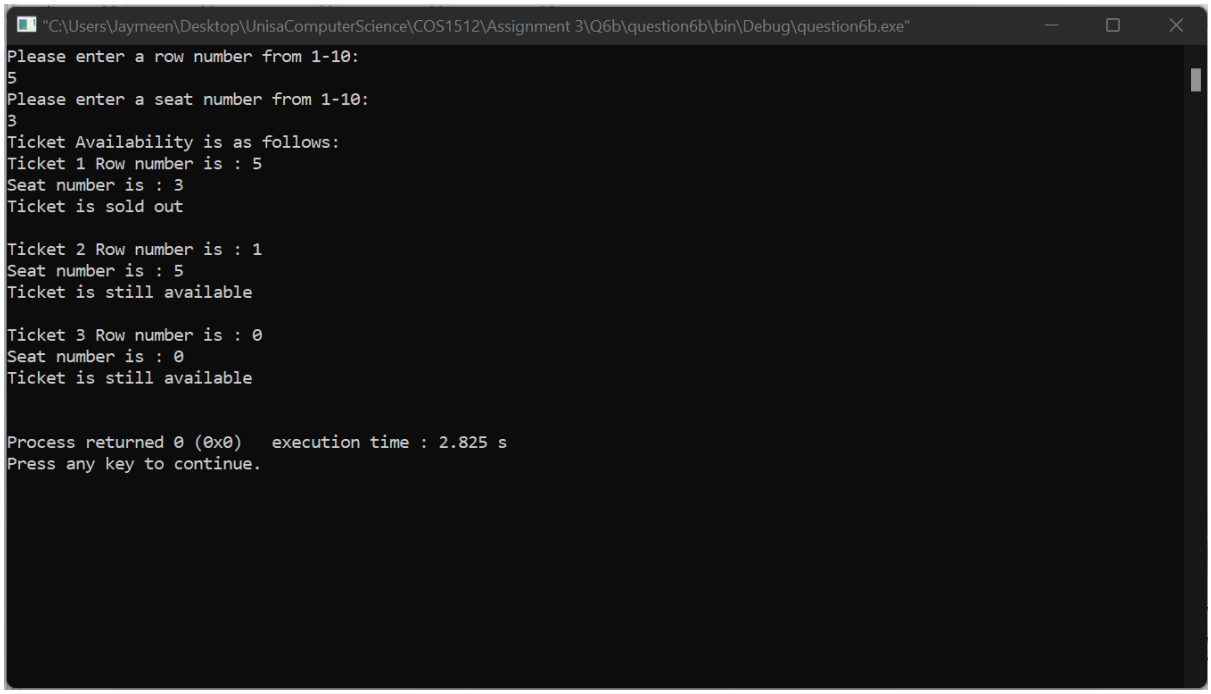
```
#include <iostream>

using namespace std;
```

```
class ShowTicket
{
   public:

      ShowTicket()
      {
        row = 0;
        seatNr = 0;
        ticketSold = false;
      }

      ShowTicket(int aRow, int aSeatNr)
      {
         row = aRow;
         seatNr = aSeatNr;
         ticketSold = false;
      }

       bool getTicketSold()
      {
         return ticketSold;
      }


      int getRow()
      {
         return row;
      }

      int getSeatNr()
      {
```

```
    return seatNr;
}



void setTicketSold(bool bTsold)
{
    ticketSold = bTsold;
}

void setRow(int arow)
{
    row = arow;
}

void setSeatNr(int aSeatNr)
{
    seatNr = aSeatNr;
}

void showOutput()
{
    string sold;
    if (ticketSold == 0)
        sold = "still available";
    else
        sold = "sold out";

    cout << "Row number is : " << row << endl;
    cout << "Seat number is : " << seatNr << endl;
    cout << "Ticket is " << sold << endl << endl;
```

```
        }

        ~ShowTicket()
        {

        }

    private:
        int row;
        int seatNr;
        bool ticketSold;

};

int main()
{
    ShowTicket showTicket1;
    ShowTicket showTicket2(1,5) ;
    ShowTicket showTicket3 ;
        int iRow;
        int iSeat;









    cout << "Please enter a row number from 1-10: " << endl;
```

```
    cin >> iRow;
    showTicket1.setRow(iRow);
    cout << "Please enter a seat number from 1-10: " << endl;
    cin >> iSeat;
    showTicket1.setSeatNr(iSeat);
    showTicket1.setTicketSold(true);
    cout << "Ticket Availability is as follows: " << endl;
    cout << "Ticket 1 ";
    showTicket1.showOutput();
    cout << "Ticket 2 ";
    showTicket2.showOutput();
    cout << "Ticket 3 ";
    showTicket3.showOutput();



    return 0;
}
```



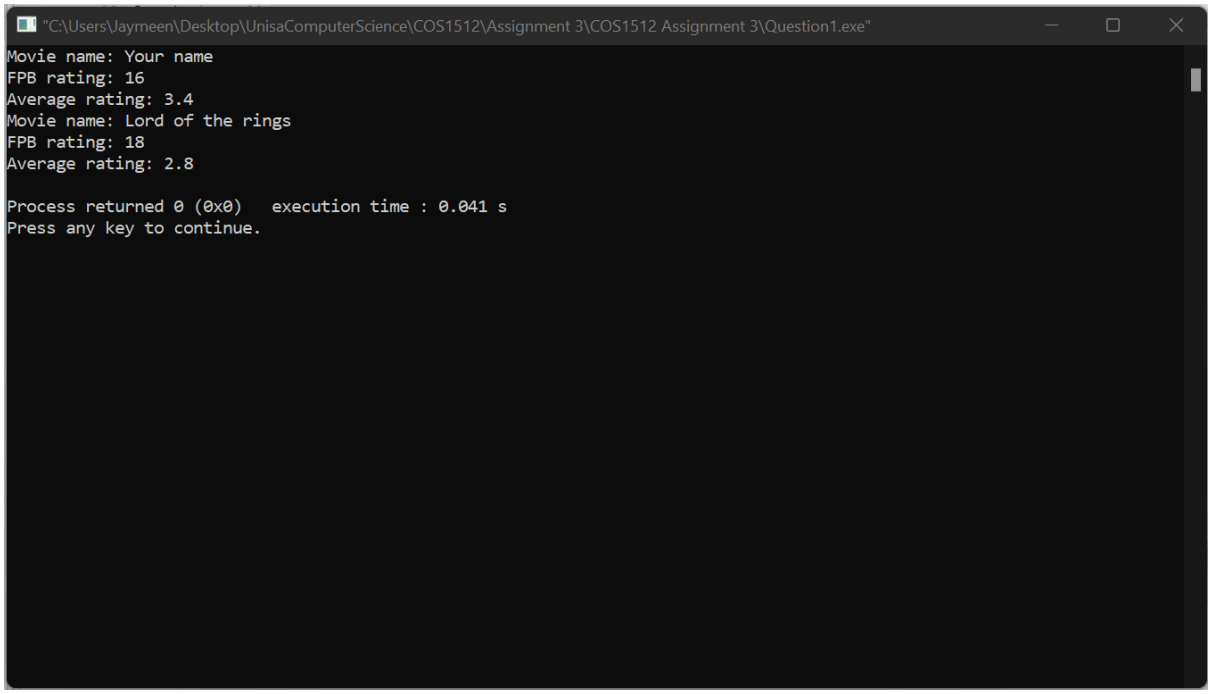
Question 3

```
#include <iostream>

using namespace std;

class Movie
{
public:

   Movie(string mName, string fpbRating)
   {
      name = mName;
      rating = fpbRating;
   }

   void setmName(string sname)
   {
      name = sname;
   }

   void setfpbRating(string srating)
   {
      rating = srating;
   }

   string getmName()
   {
      return name;
   }
```

```
string getfpbRating()
{
   return rating;
}

void addRating(int irating)
{


   if (irating>0 && irating<6)
   {
      switch(irating)
      {
         case 1: Terrible++;
               avrating = avrating + 1;
               break;
         case 2: Bad++;
               avrating = avrating + 2;
               break;
         case 3: OK++;
               avrating = avrating + 3;
               break;
         case 4: Good++;
               avrating = avrating + 4;
               break;
         case 5: Great++;
               avrating = avrating + 5;
               break;
      }

   }
```

```
        else cout << "Please enter a rating from 1-5 " << endl;



    }

    double getAverage()
    {
        double total;
        double average;



        total = avrating;
        average = total/5;



        return average;
    }



private:
    string name;
    string rating;
    int Terrible = 0;
    int Bad = 0;
    int OK = 0;
    int Good = 0;
    int Great = 0;
    double avrating = 0;
};
```

```
int main()
{



    Movie Movie1("Your name", "16");
     Movie1.addRating(4);
     Movie1.addRating(3);
     Movie1.addRating(5);
     Movie1.addRating(3);
     Movie1.addRating(2);

    Movie Movie2("Lord of the rings", "18");
     Movie2.addRating(5);
     Movie2.addRating(2);
     Movie2.addRating(4);
     Movie2.addRating(2);
     Movie2.addRating(1);

     cout << "Movie name: " << Movie1.getmName() << endl;
     cout << "FPB rating: " << Movie1.getfpbRating() << endl;
     cout << "Average rating: " << Movie1.getAverage() << endl;

     cout << "Movie name: " << Movie2.getmName() << endl;
     cout << "FPB rating: " << Movie2.getfpbRating() << endl;
     cout << "Average rating: " << Movie2.getAverage() << endl;



   return 0;
}
```



Question 4

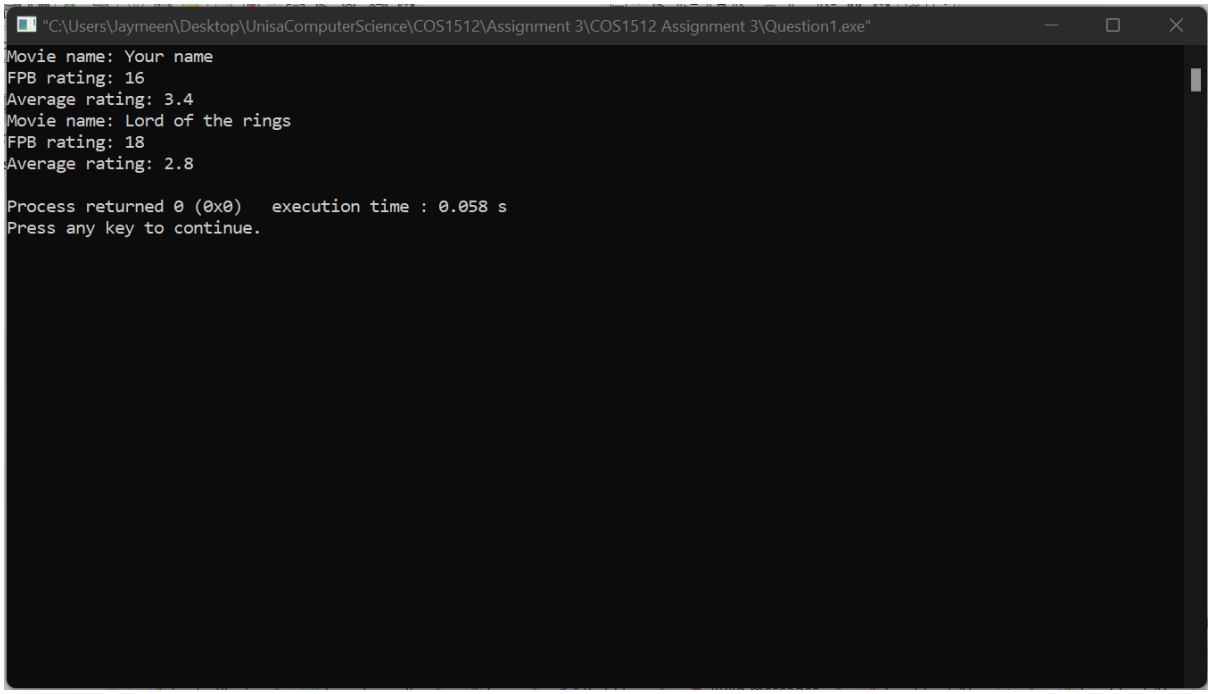
```
#include <iostream>

using namespace std;

class Movie
{
public:

   Movie(string mName, string fpbRating)
   {
      name = mName;
      rating = fpbRating;
      ratings[0] = ratings[1] = ratings[2] = ratings[3] = ratings[4] = 0;
   }
```

```
void setmName(string sname)
{
   name = sname;
}

void setfpbRating(string srating)
{
   rating = srating;
}

string getmName()
{
   return name;
}

string getfpbRating()
{
   return rating;
}

void addRating(int irating)
{
   if (irating>0 && irating<6)
   {
      switch(irating)
      {
         case 1: ratings[0]++;
                 avratings = avratings + 1;
                 break;
         case 2: ratings[1]++;
```

```
                avratings = avratings + 2;
                break;
            case 3: ratings[2]++;
                avratings = avratings + 3;
                break;
            case 4: ratings[3]++;
                avratings = avratings + 4;
                break;
            case 5: ratings[4]++;
                avratings = avratings + 5;
                break;
          }
       }
       else cout << "Please enter a rating from 1-5 " << endl;

   }

   double getAverage()
   {

       double average;


       average = avratings/5;

       return average;
   }


private:
   string name;
```

```
    string rating;
    int ratings[5];
    double avratings = 0;

};

int main()
{




      Movie Movie1("Your name", "16");
       Movie1.addRating(4);
       Movie1.addRating(3);
       Movie1.addRating(5);
       Movie1.addRating(3);
       Movie1.addRating(2);

      Movie Movie2("Lord of the rings", "18");
       Movie2.addRating(5);
       Movie2.addRating(2);
       Movie2.addRating(4);
       Movie2.addRating(2);
       Movie2.addRating(1);

       cout << "Movie name: " << Movie1.getmName() << endl;
       cout << "FPB rating: " << Movie1.getfpbRating() << endl;
       cout << "Average rating: " << Movie1.getAverage() << endl;

       cout << "Movie name: " << Movie2.getmName() << endl;
```

```
    cout << "FPB rating: " << Movie2.getfpbRating() << endl;
    cout << "Average rating: " << Movie2.getAverage() << endl;


  return 0;
}
```



Question 5

a) The keyword **public** that the variables and functions inside are accessible outside the class too, while for **private** it is only accessible in the class.

b) The purpose of a constructor is to create an object and assign values to its members.

c)
```
Money::Money()
{
    rands = 0;
    cents = 0;
}
```

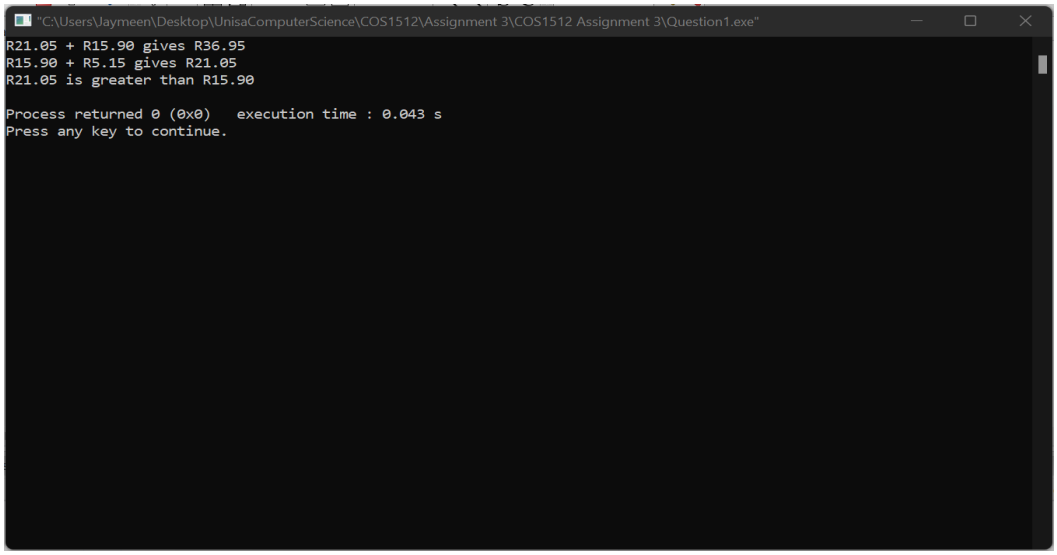
```
Money::Money(int r, int c)
{
        rands = r;
        cents = c;
}

Money::~Money()
{

}
```

d)
```
int Money::theRands() const
  {
    return rands;
  }

  int Money:: theCents() const
  {
    return cents;
  }
```

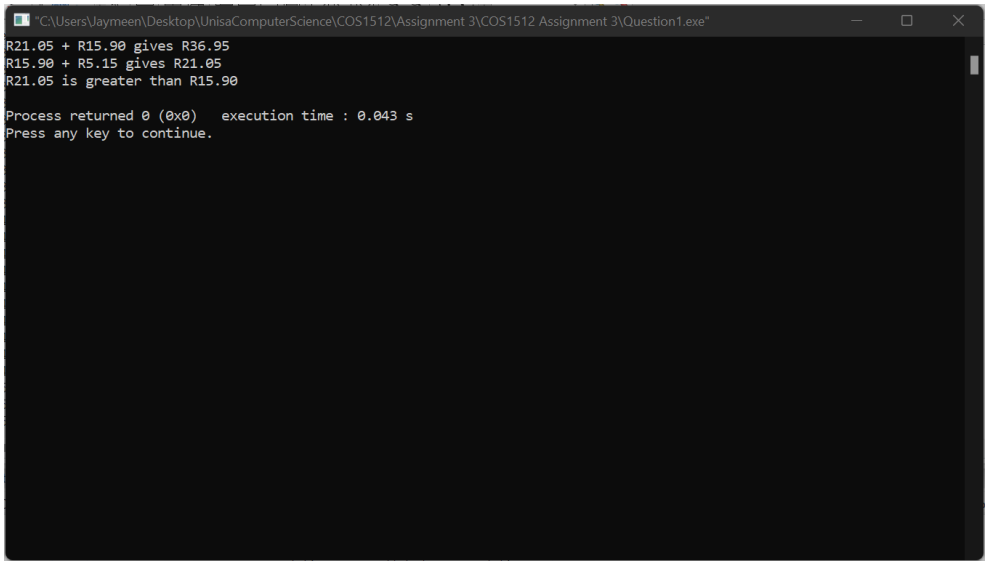
e)
```
Money Money::Plus(Money m)
{
  Money sum;
  int i;

  i = (cents + m.cents) + (100 * (rands + m.rands ));
  sum.rands = i / 100;
  sum.cents = i % 100;
  return sum;
```

```
}

f) Money Money::operator+ (Money &m)
{
   Money sum;
   int i;

   i = (cents + m.cents) + (100 * (rands + m.rands ));
   sum.rands = i / 100;
   sum.cents = i % 100;
   return sum;
}

g) bool Money::GreaterThan(Money m)
{
    ((100*rands + cents) > (100*m.theRands() + m.theCents()));

}

h) friend ostream &operator<<(ostream &ocout, Money &m) // friend functiom
{
   if (m.theCents() < 10)
 ocout << "R" << m.theRands() << ".0" << m.theCents() ;
 else
 ocout << "R" << m.theRands() << "." << m.theCents() ;
 return ocout;
}

i) //5i) method 1
bool Money::GreaterThan(Money m)
{
```

```
  ((100*rands + cents) > (100*m.theRands() + m.theCents()));


}
```




```
//5i) method 2
friend bool operator >(Money m1,Money m2)
{
   return ((100*m1.rands + m1.cents) > (100*m2.rands + m2.cents));
}
```

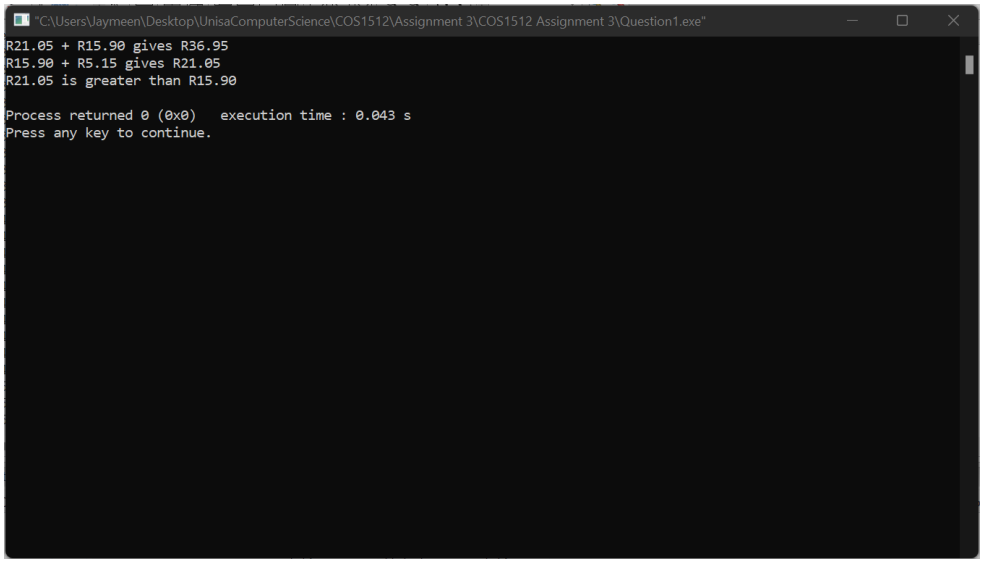



```
//5i) method 3
bool Money::operator >(Money &m) const
```

{

((100*rands + cents) > (100*m.theRands() + m.theCents()));

}



Question 6

a) Main.cpp

```
#include <iostream>
#include<fstream>
#include "Movie.h"

using namespace std;



int main()
{
```

```
    Movie Movie1("Your name", "16");
    Movie1.addRating(4);
    Movie1.addRating(3);
    Movie1.addRating(5);
    Movie1.addRating(3);
    Movie1.addRating(2);

    Movie Movie2("Lord of the rings", "18");
    Movie2.addRating(5);
    Movie2.addRating(2);
    Movie2.addRating(4);
    Movie2.addRating(2);
    Movie2.addRating(1);

    cout << "Movie name: " << Movie1.getmName() << endl;
    cout << "FPB rating: " << Movie1.getfpbRating() << endl;
    cout << "Average rating: " << Movie1.getAverage() << endl;

    cout << "Movie name: " << Movie2.getmName() << endl;
    cout << "FPB rating: " << Movie2.getfpbRating() << endl;
    cout << "Average rating: " << Movie2.getAverage() << endl;


  return 0;
}
```

Movie.cpp

```
#include "Movie.h"
```

```
#include <iostream>

Movie::Movie(string mName, string fpbRating)
{
	name = mName;
	rating = fpbRating;
	ratings[0] = ratings[1] = ratings[2] = ratings[3] = ratings[4] = 0;
}



void Movie::setmName(string sname)
{
   name = sname;
}

void Movie::setfpbRating(string srating)
{
   rating = srating;
}

string Movie::getmName()
{
   return name;
}

string Movie::getfpbRating()
{
   return rating;
}

void Movie::addRating(int irating)
```

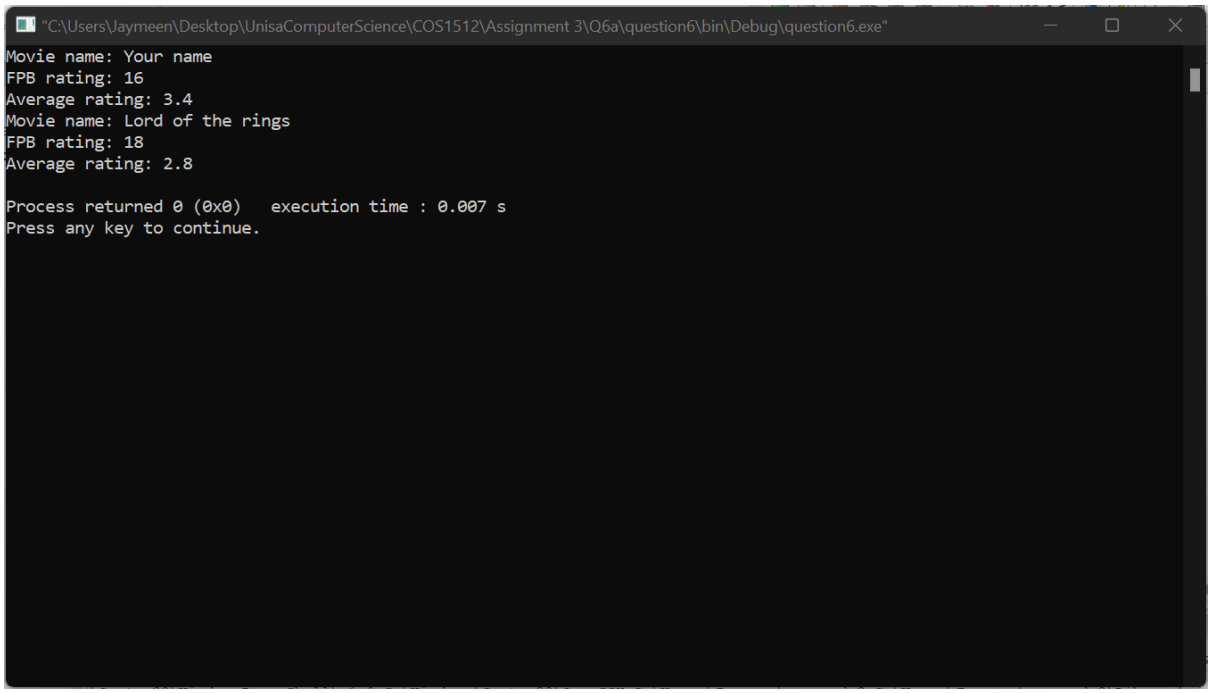
```
{
   if (irating>0 && irating<6)
      {
         switch(irating)
         {
            case 1: ratings[0]++;
                  avratings = avratings + 1;
                  break;
            case 2: ratings[1]++;
                  avratings = avratings + 2;
                  break;
            case 3: ratings[2]++;
                  avratings = avratings + 3;
                  break;
            case 4: ratings[3]++;
                  avratings = avratings + 4;
                  break;
            case 5: ratings[4]++;
                  avratings = avratings + 5;
                  break;
         }
      }
      else cout << "Please enter a rating from 1-5 " << endl;
}

double Movie::getAverage()
{
      double average;

      average = avratings/5;
```

```
        return average;
}




Movie::~Movie()
{
   //dtor
}
```

Movie.h

```
#ifndef MOVIE_H
#define MOVIE_H
#include <iostream>
#include<string>
#include "Movie.h"

using namespace std;
class Movie
{
   public:
       Movie(string mName, string fpbRating);
       void setmName(string sname);
       void setfpbRating(string srating);
       string getmName();
       string getfpbRating();
       void addRating(int irating);
       double getAverage();
       ~Movie();
```

```
    private:
    string name;
    string rating;
    int ratings[5];
    double avratings = 0;
};

#endif // MOVIE_H
```



b) Main.cpp

```
#include <iostream>
#include "Money.h"
#include<string>
using namespace std;
```

```
int main()
{
Money m1;
Money m2(15,90);
Money m3(5,15);
m1 = m2.Plus(m3);
Money mi = m1.Plus(m2);

cout << m1 << " + " << m2 << " gives " << mi << endl;

m1 = m2 + m3;

cout << m2 << " + " << m3 << " gives " << m1 << endl;


if (m2.GreaterThan(m1))

cout << m2 << " is greater than " << m1 << endl;

else

cout << m2 << " is less than " << m1 << endl;




return 0;
}

Money.h
#ifndef MONEY_H
```

```
#define MONEY_H
#include <iostream>


using namespace std;
class Money
{
   public:
       Money();
       Money(int r, int c);
      ~Money();
       int theRands() const;
       int theCents() const;
       Money Plus(Money m);
       Money operator+ (Money & m);
       bool GreaterThan(Money m);
       friend ostream &operator<<(ostream &ocout, const Money &m);
       friend bool operator > (const Money m1, const Money m2);



   private:
      int rands;
      int cents;

};

#endif // MONEY_H
```

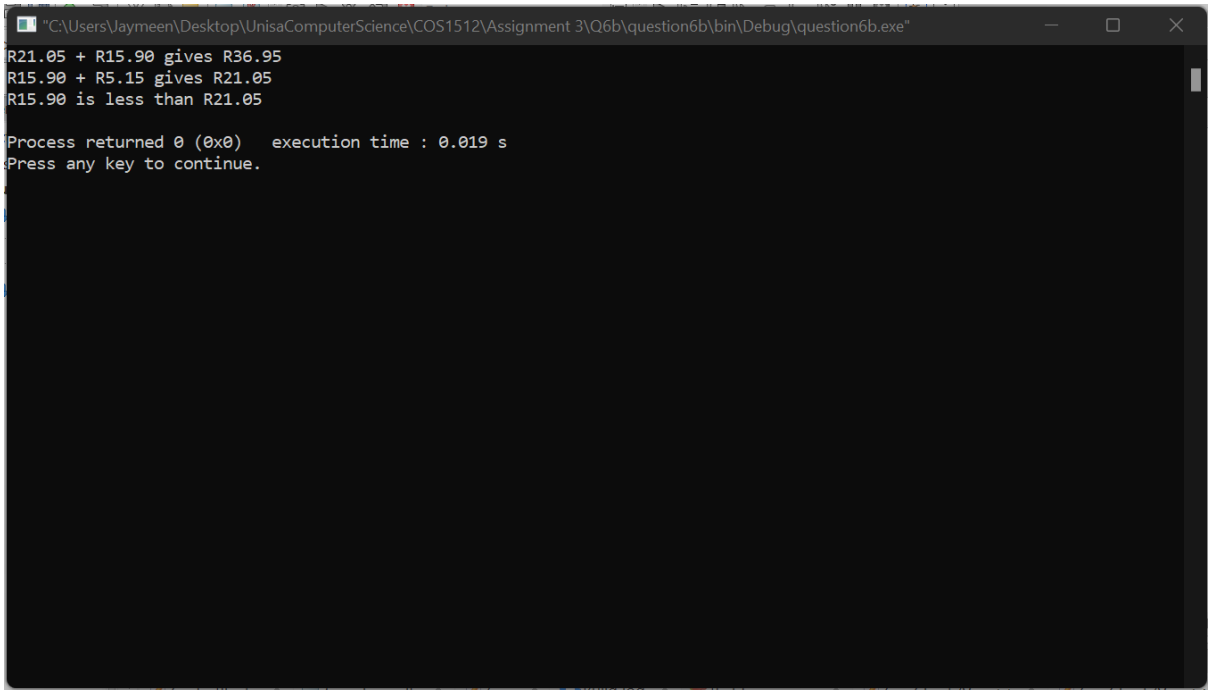
Money.cpp

```
#include "Money.h"
```

```
Money::Money()
{
 rands = 0;
 cents = 0;
}

Money::Money(int r, int c)
{
 rands = r;
 cents = c;
}

int Money::theRands() const
{
 return rands;
}

int Money::theCents() const
{
 return cents;
}

Money Money::Plus(Money m)
{
   Money sum;
   int i;

   i = (cents + m.cents) + (100 * (rands + m.rands ));
   sum.rands = i / 100;
   sum.cents = i % 100;
```

```
    return sum;
}



Money Money::operator+ (Money &m)
{
    Money sum;
    int i;

    i = (cents + m.cents) + (100 * (rands + m.rands ));
    sum.rands = i / 100;
    sum.cents = i % 100;
    return sum;
}

bool Money::GreaterThan(Money m)
{
    return ((100*rands + cents) > (100*m.theRands() + m.theCents()));

}

ostream &operator<<(ostream &ocout, const Money &m)
{
    if (m.theCents() < 10)
        ocout << "R" << m.theRands() << ".0" << m.theCents() ;
    else
        ocout << "R" << m.theRands() << "." << m.theCents() ;
    return ocout;
}

Money::~Money()
```

```
{
   //dtor
}
```



Question 7

Main.cpp

```
#include <iostream>
#include<fstream>
#include "Address.h"
#include <string>
#include<array>

using namespace std;

int main()
```

```
{


    int icount = 0;
    int icount2 = 0;
    int icount3 = 0;

    string sInput;
    string sCode;
    Address Addressobs[20];
    Address Addresscode[20];




    cout << "Please enter postal code: " << endl;
    cin >> sCode;
    cout << endl;


    ifstream infile("Address.dat");


    if (infile.fail())
    {
        cout << "File opening failed!";
        exit(1);
    }

    while (getline(infile,sInput))
    {
```

```
if(icount == 4)
{
   icount2 = icount2 + 1;
   icount = 0;
}


if (icount == 0)
{

   Addressobs[icount2].setStreetname(sInput);



}

else if (icount == 1)
{

   Addressobs[icount2].setStreetnum(stoi(sInput));


}

else if (icount == 2)
{
   Addressobs[icount2].setCity(sInput);
}

else
```

```
    {
        Addressobs[icount2].setPostalcode(sInput);


    }



    icount = icount + 1;


}


    for (int j = 0; j < (icount2 - 1) ; j++)
    {
        if (sCode == Addressobs[j].getPostalcode())
        {
            Addresscode[icount3] = Addressobs[j];
            icount3 = icount3 + 1;

            cout << Addressobs[j].getStreetname() << endl;
            cout << Addressobs[j].getStreetnum() << endl;
            cout << Addressobs[j].getCity() << endl;
            cout << Addressobs[j].getPostalcode() << endl << endl;
        }

    }
```

```
    return 0;
}
```

Address.h

```
#ifndef ADDRESS_H
#define ADDRESS_H
#include <iostream>

using namespace std;

class Address
{
    public:
        Address();
        ~Address();
        void setStreetname(string sName);
        void setStreetnum(int iNum);
        void setCity(string sCity);
        void setPostalcode(string sCode);
        string getStreetname();
        int getStreetnum();
        string getCity();
        string getPostalcode();
```

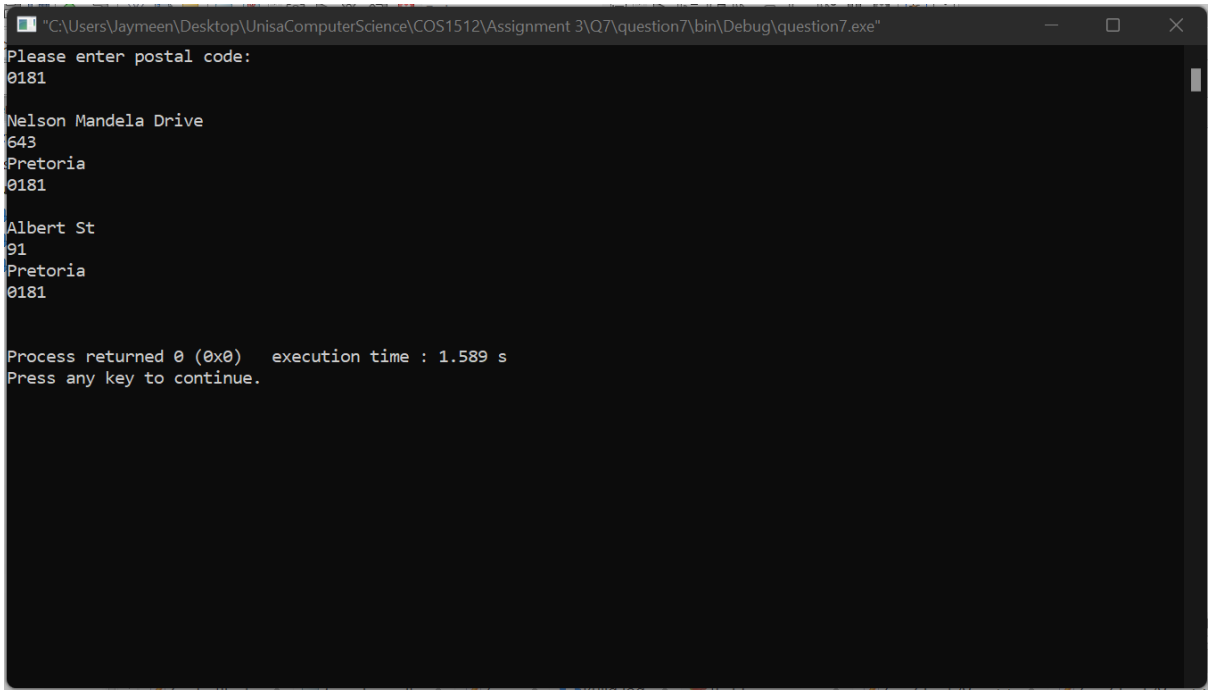
```
    friend istream &operator>>(istream &inp, Address &add);
    friend ostream &operator<<(ostream &outp, const Address &add);


  private:
    string streetName;
    int streetNr;
    string city;
    string postalCode;

};

#endif // ADDRESS_H
```

Address.cpp

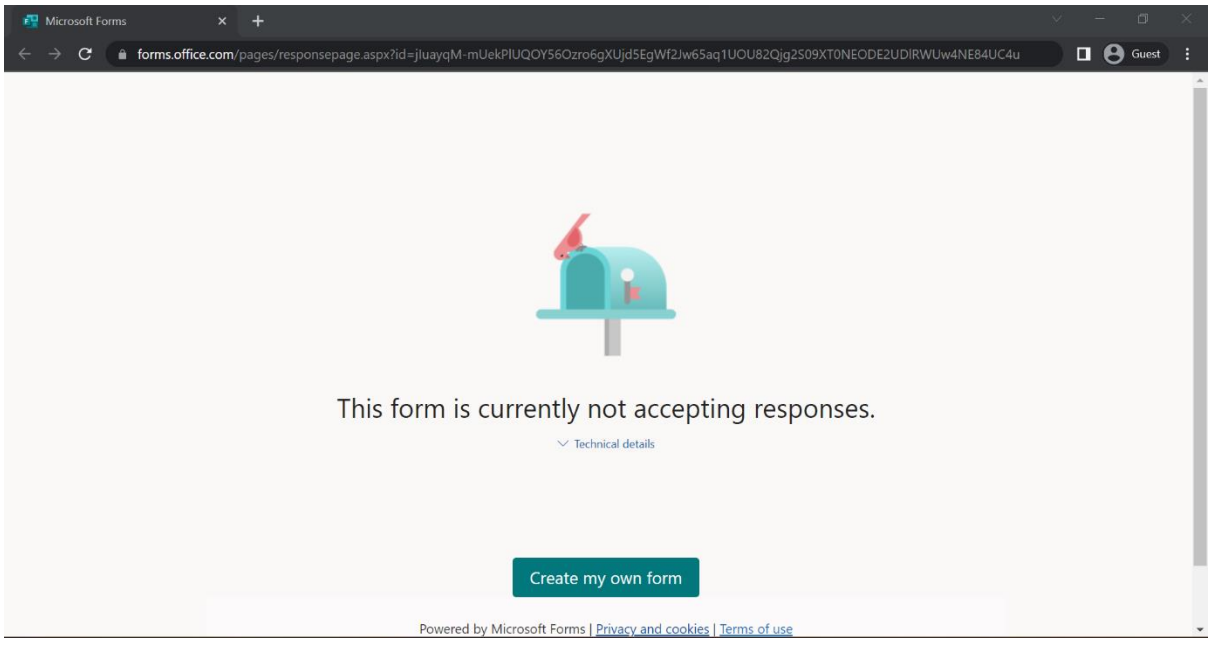
```
#include "Address.h"

Address::Address()
    {
      streetName = " ";
      streetNr = 0;
      city = " ";
      postalCode = "0000";
    }


Address::~Address()
    {
  //dtor
    }
```

```
void Address::setStreetname(string sName)
	{
	  streetName = sName;
	}
void Address::setStreetnum(int iNum)
	{
	  streetNr = iNum;
	}
void Address::setCity(string sCity)
	{
	  city = sCity;
	}
void Address::setPostalcode(string sCode)
	{
	  postalCode = sCode;
	}


string Address::getStreetname()
	{
	   return streetName;
	}
int Address::getStreetnum()
	{
	   return streetNr;
	}
string Address::getCity()
	{
	   return city;
	}
string Address::getPostalcode()
```

```
    {
        return postalCode;
    }

istream &operator>>(istream &inp, Address &add)
    {
        getline(inp, add.streetName);
        inp >> add.streetNr;
        inp >> add.city;
        inp >> add.postalCode;

        return inp;
    }

ostream &operator<<(ostream &outp, const Address &add)
    {
        outp << "Street Name: " << add.streetName << endl;
        outp << "Street Number: " << add.streetNr << endl;
        outp << "City: " << add.city << endl;
        outp << "Postal code: " << add.postalCode << endl;

        return outp;
    }
```


"C:\Users\Jaymeen\Desktop\UnisaComputerScience\COS1512\Assignment 3\Q7\question7\bin\Debug\question7.exe"
Please enter postal code:
0181
Nelson Mandela Drive
643
Pretoria
0181
Albert St
91
Pretoria
0181
Process returned 0 (0x0) execution time : 1.589 s
Press any key to continue.

Question 8


Microsoft Forms
forms.office.com/pages/responsepage.aspx?id=jluayqM-mUekPlUQOY56Ozro6gXUjd5EgWf2Jw65aq1UOU82Qjg2S09XT0NEODE2UDlRWUw4NE84UC4u
Guest
This form is currently not accepting responses.
Technical details
Create my own form
Powered by Microsoft Forms | Privacy and cookies | Terms of use