Lab Assignment #5

Problem DP1 Create a class Session, which can store the details of user into session while user get logged into the application. The session must have implement an indexer to fulfill the requirement.

Problem DP2 User can retrieve the details by providing the index name into the session object.

```
using System;

namespace Lab1
{
   public class Session{
      public string[] key=new string[10];
      public string[] values=new string[10];
      public string this[string index]{
        get
        {
          int i=0;
          foreach(string item in key)
          {
            if(item==index)
              return values[i];
            i++;
          }
            return null;
        }
        set{
          int i=0;
          foreach(string item in key)
          {
            if(item==index && item!=null)
            {
            values[i]=value;
            i++;
            }
            else if(item==null)
            {
               key[i]=index;
               values[i]=value;
            }
            else
              i++;
          }
        }
      }
      static void Main(string[] args)
      {
      Session s=new Session();
```

```
      s["username"]="Nikita";
      s["password"]="123";
      s["email"]="nikita@gmail.com";
      s["email"]="n@gmail.com";
      s["username"]="NK";
      s["password"]="123456";
      Console.WriteLine("User Logged In");
      Console.WriteLine("Username:"+s["username"]+"\t\tPassword:"+s["password"]+"\t\tE
mail:"+s["email"]);
      }

   }
}
```

OUTPUT

```
PS C:\Users\user\Desktop\SEM-3\C#\C-sharp programs\D> dotnet run
User Logged In
Username:NK             Password:123456         Email:n@gmail.com
PS C:\Users\user\Desktop\SEM-3\C#\C-sharp programs\D>
```

Problem DA1 The stock (also capital stock) of a corporation constitutes the equity stake of its owners. It represents the residual assets of the company that would be due to stockholders after discharge of all senior claims such as secured and unsecured debt. Stock market prediction is the act of trying to determine the future value of a company stock or other financial instrument traded on an exchange. The successful prediction of a stock's future price could yield significant profit. The Stock Market prediction task is interesting as well as divides researchers and academics into two groups those who believe that we can devise mechanisms to predict the market and those who believe that the market is efficient and whenever new information comes up the market absorbs it by correcting itself, thus there is no space for prediction.

Problem 1: How to Predict Stock Value? You need to identify all the classes and explicitly invoke the method for desired prediction.

Method Momentum

1. Divide today's close by the close a certain number of days ago. For example, you can look back five days.

2. Multiply that number by 100.

M = (Price Today/Price Five Days Ago) x100

M = (15/10) x 100 = 150

Method Mean Reversion (Basic)

Find out the mean of last X days then evaluate the correlation between today's price and mean price to predict the values of stock for the next day.

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$$

$$\sigma_x = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \qquad \sigma_y = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(y_i - \bar{y})^2}$$

$$\sigma_{x,y} = \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

$$\rho_{x,y} = \frac{\sigma_{x,y}}{\sigma_x \sigma_y}$$

```
using System;


namespace DA
{
   class Stock{
      double [] price=new double[10];
      public double this[int index]{
         get{return price[index];}
         set{price[index]=value;}
      }
      public double MethodMomentum(int days){
         double M=(price[0]/price[days])*100;
         return M;
      }
   }
   class Exchange{
      double [] price=new double[10];
      public double this[int index]{
         get{return price[index];}
         set{price[index]=value;}
      }
      public double MethodMomentum(int days){
         double M=(price[0]/price[days])*100;
         return M;
      }
   }
   class Program
   {
       public static double mean(Stock a,int n){
         double sum_x=0;
         for(int i=0;i<n;i++)
            sum_x+=a[i];
         //MEAN
        return sum_x/n;
      }
      public static double mean(Exchange a,int n){
```

```
        double sum_x=0;
        for(int i=0;i<n;i++)
            sum_x+=a[i];
        //MEAN
       return sum_x/n;
    }
    public static double StandardDeviation(Stock a,int n){
        double[] a_new=new double[10];
        double aa_sum=0;
         for(int i=0;i<n;i++)
            a_new[i]=Math.Pow((a[i]-mean(a,n)),2);
         for(int i=0;i<n;i++)
            aa_sum+=a_new[i];
        return Math.Sqrt((aa_sum)/(n-1));
    }
    public static double StandardDeviation(Exchange a,int n){
        double[] a_new=new double[10];
        double aa_sum=0;
         for(int i=0;i<n;i++)
            a_new[i]=Math.Pow((a[i]-mean(a,n)),2);
         for(int i=0;i<n;i++)
            aa_sum+=a_new[i];
        return Math.Sqrt((aa_sum)/(n-1));
    }
     public static double Covariance(Stock a,Exchange b,int n){
        //FORMULA (ai-amean)*(bi=bmean)
        int i;
        double[] a_new=new double[10];
        double[] b_new=new double[10];
        double[] ab_new=new double[10];
        double ab_sum=0;
         for(i=0;i<n;i++){
            a_new[i]=a[i]-mean(a,n);
            b_new[i]=b[i]-mean(b,n);
            ab_new[i]=a_new[i]*b_new[i];
         }

         for(i=0;i<n;i++)
            ab_sum+= ab_new[i];
        return ab_sum/(n-1);
    }
    public static double Corelation(Stock a,Exchange b,int n){
        return Covariance(a,b,n)/(StandardDeviation(a,n)*StandardDeviation(b,n));
    }
    static void Main(string[] args)
    {
        int n=10;
        Stock MRF=new Stock();
```

```
        MRF[0]=16980;
        MRF[1]=16960;
        MRF[2]=16970;
        MRF[3]=16975;
        MRF[4]=16972;
        MRF[5]=16970;
        MRF[6]=16950;
        MRF[7]=16945;
        MRF[8]=16950;
        MRF[9]=16930;
        Exchange NSE=new Exchange();
        NSE[0]=20180;
        NSE[1]=20175;
        NSE[2]=20160;
        NSE[3]=20155;
        NSE[4]=20165;
        NSE[5]=20145;
        NSE[6]=20130;
        NSE[7]=20120;
        NSE[8]=20130;
        NSE[9]=20125;
        Console.WriteLine("\n*****Method Momentum*****");
        Console.WriteLine("How may Days ago you want to Check?");
        int day=Convert.ToInt32(Console.ReadLine());
        Console.WriteLine("Method Momentum(MRF)="+Math.Round(MRF.MethodMome
ntum(day),2));
        Console.WriteLine("Method Momentum(NSE)= "+Math.Round(NSE.MethodMomen
tum(day),2));
        Console.WriteLine("\n*****Method Mean Reversion*****");
        Console.WriteLine("Corelation(MRF,NSE)= "+Math.Round(Corelation(MRF,NSE,n)
,2));
      }
   }
}
```

OUTPUT

```
PS C:\Users\user\Desktop\SEM-3\C#\C-sharp programs\DA> dotnet run

*****Method Momentum*****
How may Days ago you want to Check?
2
Method Momentum(MRF)=100.06
Method Momentum(NSE)= 100.1

*****Method Mean Reversion*****
Corelation(MRF,NSE)= 0.81
PS C:\Users\user\Desktop\SEM-3\C#\C-sharp programs\DA>
```