**.NET Assignment - 6**

Github link : <https://github.com/Aishwarya01-github/.NET-Assignments/tree/main/Lab%206>

1. **Code :**

**using System;**

**using System.Collections.Generic;**

**using System.Text;**

**namespace ConsoleApp18jan2022**

**{**

**class SelectionSort {**

**static public void Sort<T>(IList<T> sortArray, Func<T,T,bool> comparision) {**

**bool swapped = true;**

**do {**

**int position , i , j;**

**swapped = false;**

**for(i=0; i < sortArray.Count - 1; i++) {**

**position = i;**

**for(j = i; j < sortArray.Count; j++) {**

**if(comparision(sortArray[position],sortArray[j])) {**

**position = j;**

**}**

**}**

**if(position!=i){**

**T temp = sortArray[i];**

**sortArray[i] = sortArray[i+1];**

**sortArray[i+1] = temp;**

**swapped=true;**

**}**

**}**

**} while(swapped);**

**}**

**}**

**public enum Designations**

**{**

**CEO=4,**

**CFO=10,**

**sde=5,**

**ba=2,**

**pm=6**

**}**

**class Employee**

**{**

**int empid;**

**float salary;**

**public string name;**

**Designations designation;**

**public Employee(int EmpId, float Salary, string NAme, Designations design)**

**{**

**this.empid = EmpId;**

**this.salary = Salary;**

**this.name = NAme;**

**this.designation = design;**

**}**

**internal static bool CompareSalary(Employee e1, Employee e2)**

**{**

**return e1.salary < e2.salary;**

**}**

**internal static bool CompareNames(Employee e1, Employee e2)**

**{**

**if(e1.name.CompareTo(e2.name)== 0)**

**{**

**return true;**

**}**

**else**

**{**

**return false;**

**}**

**}**

**internal static bool CompareDesignations(Employee e1, Employee e2)**

**{**

**return e1.designation < e2.designation;**

**}**

**}**

**public class TestGenericMethods**

**{**

**public static void Main(string[] args)**

**{**

**List<Employee> emplist = new List<Employee>(30);**

**emplist.Add(new Employee(1111, 250000, "Harsh", Designations.CFO));**

**emplist.Add(new Employee(0011, 12000000, "Jay", Designations.CEO));**

**emplist.Add(new Employee(2001, 501000, "Aish", Designations.sde));**

**emplist.Add(new Employee(2004,20999,"Kezal",Designations.CEO));**

**emplist.Add(new Employee(4001,78000,"Rushita",Designations.sde));**

**emplist.Add(new Employee(6006,900000,"Yash",Designations.CFO));**

**SelectionSort.Sort<Employee>(emplist, Employee.CompareSalary);**

**foreach(Employee e1 in emplist)**

**{**

**Console.WriteLine(e1.name);**

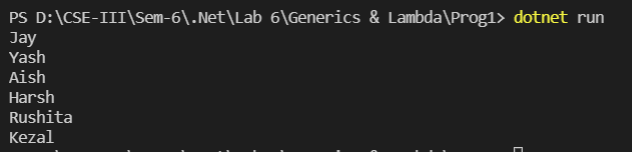
**}**

**}**

**}**

**}**

**Output :**

****

1. **Code :**

**class ATM {**

**public int account\_no;**

**public String account\_holder\_name;**

**public double avaliable\_balance;**

**public ATM(int acc\_no, String acc\_name, double ava\_bal) {**

**this.account\_no = acc\_no;**

**this.account\_holder\_name = acc\_name;**

**this.avaliable\_balance = ava\_bal;**

**}**

**}**

**public delegate void WatchBalance(double n);**

**class Program {**

**public static void checkBalance(double bal) {**

**if(bal < 0) {**

**Console.WriteLine("You are overdrawn");**

**} else if (bal < 10) {**

**Console.WriteLine("Your account balance is very low");**

**} else if (bal < 100) {**

**Console.WriteLine("Watch your spending carefully.");**

**} else {**

**Console.WriteLine("You have over $100 in your account");**

**}**

**}**

**static void Main() {**

**WatchBalance watch = checkBalance;**

**ATM ac1 = new ATM(200012232,"Rahul",-100);**

**watch(ac1.avaliable\_balance);**

**Console.WriteLine("Mr. " + ac1.account\_holder\_name + " your current balance is " + ac1.avaliable\_balance);**

**Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**

**ATM ac2 = new ATM(200012255,"Jay",400);**

**watch(ac2.avaliable\_balance);**

**Console.WriteLine("Mr. " + ac2.account\_holder\_name + " your current balance is " + ac2.avaliable\_balance);**

**Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**

**ATM ac3 = new ATM(200012222,"Kezal",10);**

**watch(ac3.avaliable\_balance);**

**Console.WriteLine("Ms. " + ac3.account\_holder\_name + " your current balance is " + ac3.avaliable\_balance);**

**Console.WriteLine("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*");**

**ATM ac4 = new ATM(200012291,"Aish",50);**

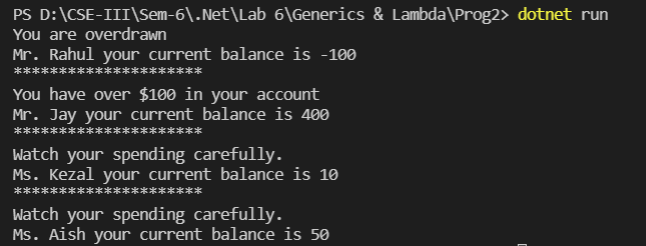
**watch(ac4.avaliable\_balance);**

**Console.WriteLine("Ms. " + ac4.account\_holder\_name + " your current balance is " + ac4.avaliable\_balance);**

**}**

**}**

**Output :**

****

1. **Code :**

**using System;**

**namespace Code\_3**

**{**

**public delegate double Maximum(double a, double b);**

**class Program**

**{**

**public static void CallAnonymousMethod()**

**{**

**bool positive = new Func<int, bool>(delegate (int int32) { return int32 > 0; })(1);**

**new Action<bool>(delegate (bool value) { Console.WriteLine(value); })(positive);**

**}**

**public static void CallLambda()**

**{**

**bool positive = new Func<int, bool>(int32 => int32 > 0)(1);**

**new Action<bool>(value => Console.WriteLine(value))(positive);**

**}**

**static void Main(string[] args)**

**{**

**Console.WriteLine("\n1) Anonymous Function and Action Delegate Without Lamda Expression : ");**

**Program.CallAnonymousMethod();**

**Console.WriteLine("\n2) Anonymous Function and Action Delegate With Lamda Expression : ");**

**Program.CallLambda();**

**//delegate with lamda expression**

**Console.WriteLine("\n3) Delegate with Lamda Expression : ");**

**Maximum parse = (double x, double y) => (x > y ? x : y);**

**Console.WriteLine("\nMax(11.5,1.1) = {0}", parse(11.5, 1.1));**

**Console.WriteLine("\n4) Function Delegate with Lamda Expression : ");**

**Func<double, double, double> f = (x, y) => { if (x > y) return x; return y; };**

**double a1 = f(40, 30);**

**Console.WriteLine("\nMax(40,30) = {0}", a1);**

**Console.WriteLine("\n5) Function Delegate with Diffrent Lamda Expressions : ");**

**double a2;**

**Func<double, double, double> f2;**

**f2 = (x, y) =>**

**{**

**if (x > y)**

**return x;**

**return y;**

**};**

**a2 = f2(10, 20);**

**Console.WriteLine("\nMax(40,30) = {0}", a2);**

**f2 = (x, y) =>**

**{**

**if (x < y)**

**return x;**

**return y;**

**};**

**a2 = f2(40, 20);**

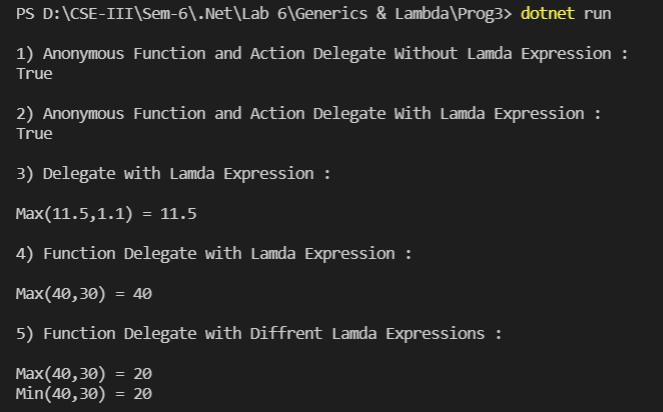
**Console.WriteLine("Min(40,30) = {0}\n", a2);**

**}**

**}**

**}**

**Output :**

****