| **Name of Student:** Ajay Karthikesan | | | |
| --- | --- | --- | --- |
| **Roll Number:** 57 | | **Assignment Number:** 2 | |
| **Aim of Assignment:** Design Applications using Classes and Objects. | | | |
| **DOP:** 5.3.23 | | **DOS:** 19.4.23 | |
| **CO Mapped:**  CO1 | **PO Mapped:**  PO3, PO5, PSO1, PSO2 | **Faculty Signature:** | **Marks:** |

## 

## Practical No. 2

**Aim:** Design Applications using Classes and Objects.

1. Write a program to declare a class “staff” having data members as name and post. Accept this data 5 staffs and display names of staff who are HOD.
2. Define a class “salary” which will contain member variable Basic, TA, DA, HRA. Write a program using Constructor with default values for DA and HRA and calculate the salary of the employee.

**Theory:**

1. Class: A Class is like an object constructor, or a “blueprint” for creating objects. It can have fields, methods, constructors etc.
2. Object: Object is an entity that has state and behavior. Here, state means data and behavior means functionality. Object is a runtime entity, it is created at runtime. Object is an instance of a class. All the members of the class can be accessed through object.
3. Array: An array is a collection of items stored at contiguous memory locations. Arrays are used to store multiple values in a single variable, instead of declaring separate variables for each value.
4. Array of Object: In C#, arrays are actually objects, and not just addressable regions of contiguous memory as in C and C++. Object Array can store an element of various types in a single collection.
5. Constructor: Constructor is a special method which is invoked automatically at the time of object creation. It is used to initialize the data members of new object. The constructor has the same name as class. There can be two types of constructors in C#.
   1. Default constructor.
   2. Parameterized constructor
   3. Copy constructor
   4. Static constructor
6. Types of Constructor:
   1. Default Constructor: A constructor without any parameters called default constructor. In this constructor every instance of the class will be initialized without any parameter values.
   2. Parameterized constructor: A constructor with at least one parameter is called as parameterized constructor. In parameterized constructor we can initialize each instance of the class to different values.
   3. Copy constructor: A parameterized constructor that contains a parameter of same class type is called as copy constructor. Main purpose of copy constructor is to initialize new instance to the values of an existing instance.
   4. Static constructor: When we declared constructor as static it will be invoked only once for any number of instances of the class and it’s during the creation of first instance of the class or the first reference to a static member in the class. Static constructor is used to initialize static fields of the class and to write the code that needs to be executed only once.

**Code:**

File: 1problem/StaffRegistrationForm.cs

﻿using System;

using System.Collections;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_2APractical.\_1problem

{

public partial class StaffRegistrationForm : Form

{

private uint \_noOfStaffs = 0;

private List<Staff> staffs = new();

public StaffRegistrationForm()

{

InitializeComponent();

postCbb.DataSource = Enum.GetValues(typeof(Post));

}

private void IncrementNoOfStaffs()

{

++\_noOfStaffs;

noOfStaffs.Text = \_noOfStaffs.ToString();

}

private void AddStaff()

{

Post post = (Post)postCbb.SelectedIndex;

Staff staff = new Staff(nameTxt.Text, post);

staffs.Add(staff);

IncrementNoOfStaffs();

}

private void addStaffBtn\_Click(object sender, EventArgs e)

{

AddStaff();

nameTxt.Text = "";

}

private void ShowHODS(List<Staff> hods)

{

List<string> hodsNames = new();

foreach (Staff staff in hods)

{

hodsNames.Add(staff.Name);

}

HODListForm hODListForm = new(hodsNames);

hODListForm.Show();

}

private void showHodsBtn\_Click(object sender, EventArgs e)

{

List<Staff> hods = staffs.FindAll(staff => staff.Post == Post.HOD);

if (hods.Count == 0)

{

MessageBox.Show("No HODS are there.");

}

else

{

ShowHODS(hods);

}

}

}

}

File: 1problem/Staff.cs

﻿using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2APractical.\_1problem

{

internal class Staff

{

public string Name { get; private set; }

public Post Post { get; private set; }

internal Staff(string name,Post post)

{

this.Name = name;

this.Post = post;

}

}

}

File: 1problem/Post.cs

﻿using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2APractical.\_1problem

{

enum Post

{

Poen,

HOD,

Principal

}

}

File: 1problem/HODListForm.cs

﻿using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_2APractical.\_1problem

{

public partial class HODListForm : Form

{

internal HODListForm(List<string> hods)

{

InitializeComponent();

hodsLbx.DataSource = hods;

}

}

}

File: 2prblem/Salary.cs

﻿using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace \_2APractical.\_2problem

{

internal class Salary

{

decimal basic;

byte ta;

byte da;

byte hra;

internal Salary(decimal basic,byte ta, byte da=20, byte hra=50)

{

this.basic = basic;

this.ta = ta;

this.da = da;

this.hra=hra;

}

private decimal PercentOfBasic(byte rate)

{

return (basic\*rate)/100;

}

public decimal get()

{

return basic + PercentOfBasic(ta) + PercentOfBasic(da) + PercentOfBasic(hra);

}

}

}

File: 2problem/SalaryCalculatorForm.cs

﻿using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_2APractical.\_2problem

{

public partial class SalaryCalculatorForm : Form

{

public SalaryCalculatorForm()

{

InitializeComponent();

}

private void ShowSalary()

{

Salary salary;

if(daTxt.Text=="" || hraTxt.Text == "")

{

salary = new(Convert.ToDecimal(basicTxt.Text), Convert.ToByte(taTxt.Text));

}

else

{

salary = new(Convert.ToDecimal(basicTxt.Text), Convert.ToByte(taTxt.Text),

Convert.ToByte(daTxt.Text), Convert.ToByte(hraTxt.Text));

}

MessageBox.Show($"Your salary is {salary.get().ToString("0.##")}");

}

private void findSalBtn\_Click(object sender, EventArgs e)

{

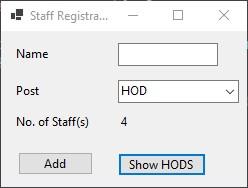
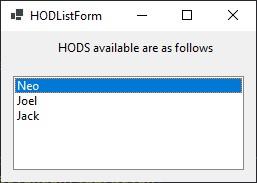
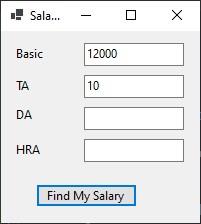
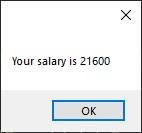
ShowSalary();

}

}

}

**Output:**

1.  
2.  

**Conclusion:** I learnt how to design Applications using Classes and Objects.