C# OOPS

Static Functions

```
//class function code
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
   internal class Employee
   {

      public static int empid;
      public static string empname;
      public static string design;
      public static decimal salary;

      public static void getempinfo()
      {
         Console.WriteLine("enter emp details");
         empid = Convert.ToInt32(Console.ReadLine());
         empname = Console.ReadLine();
         design = Console.ReadLine();
         salary = Convert.ToDecimal(Console.ReadLine());

      }
      public static void displayempinfo()
      {
         Console.WriteLine($" empid {empid} name {empname}" +
            $" design {design} salary {salary}");
```

```
        }
    }

}
//main function
using System;
using System.Data.SqlClient;

namespace ConsoleApp1
{1
    internal class Program
    {

        static void Main(string[] args)
        {

            //--------------------static
            //On static func declare we can acess without create the obj or instance
            Employee.getempinfo();
            Employee.displayempinfo();
        }

    }
}
```

constructor

//constructor name should as same as class , class name

//class always as default constructor
// one class contain one or more constructor called constructor overloading
//always be public
//types -default , parameterized , copy constructor

// otherwise take the priority

// max use for initialize the value for any variable

```
//classs program
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
{
   internal class Employee
   {
      int empcomp;
      static Employee()
      {
         Console.WriteLine("static constructor called");
      } // static doesn't need public  specifier
      //static take higher priority

      public Employee()
      {
         Console.WriteLine("parameter less or default  constructor");
      }
      public Employee(int empid)
      {
         this.empcomp = empid;
         Console.WriteLine($"parameterised constructor {empid}");
      }
      public Employee(int empcomp , string name)
      {
         this.empcomp = empcomp;
         Console.WriteLine($"parametrised invooked \n companyid {empcomp} na
      }
      public Employee(Employee employee)
      {
         this.empcomp = employee.empcomp;
```

```
            Console.WriteLine($"copy construtor {empcomp}");
        }
    }

}
//main program
using System;
using System.Data.SqlClient;

namespace ConsoleApp1
{
    internal class Program
    {
        static void Main(string[] args)
        {
            Employee employee = new Employee();
            Employee employee1 = new Employee(20);
            Employee employee2 = new Employee(1, "poornima");
            Employee emp3 = new Employee(employee2);
            Console.ReadLine();

        }
    }
}
```

Inheritance

```
//class porgram
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleApp1
```

```
{
    public class Employee
    {
        public int empid;
        public string empname;
        public string design;
        public decimal salary;

        //constructor
        public Employee(int id, string name, string design, decimal salary)
        {
            this.empid = id;
            this.empname = name;
            this.design = design;
            this.salary = salary;
        }
        //method
        public void displayemp()
        {
            Console.WriteLine($"id {empid} name {empname} " +
                $"design {design} salary{salary}");
        }
    }
    //inheritace
    public class Manager : Employee
    {
        string deprt;
        public Manager(int id, string name, string design, decimal salary, string depr
            base(id, name, design, salary)
        {
            this.deprt = deprt;
        }

        public void displaymanager()
        {
            displayemp();
```

```
            Console.WriteLine($"Manager is Managing {deprt}");
        }
    }


}
//main program
using System;
using System.Data.SqlClient;

namespace ConsoleApp1
{
    public class Program
    {
        static void Main(string[] args)
        {
            Manager manager = new Manager(101, "latha", "product engineer", 3324
            manager.displaymanager();


        }
    }
}
```

AccessSpecifier

//public , private , protected

```
// class program

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
namespace ConsoleApp1
{
   public class BankAccount
   {
      private string accnum;
      public string accholdername;
      protected double balance;
      public BankAccount(string accnum, string accholdername)
      {
         this.accnum = accnum;
         this.accholdername = accholdername;
         balance = 0.0;
      }
      public void Deposit(double amount)
      {
         this.balance += amount;
      }
      private void ShowAccountNum()
      {
         Console.WriteLine($"Account number ={accnum}");
      }
      protected void ShowBalance()
      {
         Console.WriteLine($"Balance ={balance}");
      }
   }
   class RecurringDeposit : BankAccount
   {
      public int tenureMonth;
      public int amountPerMonth;
      public double maturityAmount;
     public RecurringDeposit(string accnum, string accholdername,  int tenureMo
         base(accnum, accholdername)
      {

         this.tenureMonth = tenureMonth;
```

```
            this.amountPerMonth = amountPerMonth;
            this.maturityAmount = 0;



        }

        public void GetRDreturn()
        {
            ShowBalance();
            Deposit(34000);
            // ShowAccountNumber();
            double interest = (tenureMonth * amountPerMonth * (tenureMonth + 1) * 5
            maturityAmount = (amountPerMonth * tenureMonth) + interest;
            Console.WriteLine($"Total Amount After RD Completed {maturityAmount}

        }

    }
}
//main program
using System;
using System.Data.SqlClient;

namespace ConsoleApp1
{
    public class Program
    {
        static void Main(string[] args)
        {
            RecurringDeposit recurringDeposit = new RecurringDeposit("001", "Geeth

            recurringDeposit.Deposit(70000);
            recurringDeposit.GetRDreturn();
            Console.ReadLine();

        }
```

```
    }
}
```