Oops Concepts

1. Class : User defined type , Blueprint /template based on which multiple variables(object)

Could be declared.

Int data type indicates , what we can store & functions or operations we can perform

Int x, y , z;

1. Object : Instance / Variable of class

Int **x**;

Student **student**;

1. Polymorphism : One name different forms

Add(int , int)

Add(int , int , int)

Add(float , int)

Polymorphism : It is of 2 types

1. Compile Time > Achieved thru Function or Method Overloading
2. Run Time > Achieved thru Abstract Classes & Virtual Functions
3. Abstraction : Showing details which are needed by user
4. Encapsulation : Hiding the details which are not needed by the user

How do we achieve them by using **Access Sepecifier**

**Both of them go side by side**

1. Inheritance > Deriving Features of one class to other class

Advantage : Reusability of Code

1. Data Hiding > Data of a class is hidden outside the class making data secure

**Inheritance** : > Deriving Features of one class to other class

Advantage : Reusability of Code

**The class from which the features are inherited is known as**

**Parent Class / Super Class /Base Class**

**The class into which the features are inherited is known as**

**Child Class / Sub Class / Derived Class**

**Syntax of Inheritance**

**Class A {}**

**Class B : A**

**{**

**}**

We created a class Employee

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

class Employee

{

int id;

string name;

string manager;

int salary;

public Employee()

{

}

public Employee(int id, string name, string manager,

int salary)

{

this.id = id;

this.name = name;

this.manager = manager;

this.salary = salary;

}

public void GetDetails()

{

Console.WriteLine("Enter ID");

id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

name = Console.ReadLine();

Console.WriteLine("Enter Manager Name");

manager = Console.ReadLine();

Console.WriteLine("Enter Salary");

salary = int.Parse(Console.ReadLine());

}

public void DisplayDetails()

{

Console.WriteLine("ID is " + id);

Console.WriteLine("Name is " + name);

Console.WriteLine("Manager Name is " + manager);

Console.WriteLine("Salary is " + salary);

}

}

Employees could be 2 types

Parttime

Fulltime

**Common attributes**

ID

Name

Manager

**Parttime Employees**

Commercials

Duration

**FullTime Employees**

Salary

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

class PartTimeEmployee

{

int id;

string name;

string manager;

int commercials;

string Duration;

public PartTimeEmployee()

{

}

public PartTimeEmployee(int id, string name, string manager,

int commercials, string duration)

{

this.id = id;

this.name = name;

this.manager = manager;

this.commercials = commercials;

this.Duration = duration;

}

public void GetDetails()

{

Console.WriteLine("Enter ID");

id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

name = Console.ReadLine();

Console.WriteLine("Enter Manager Name");

manager = Console.ReadLine();

Console.WriteLine("Enter commercials");

commercials = int.Parse(Console.ReadLine());

}

public void DisplayDetails()

{

Console.WriteLine("ID is " + id);

Console.WriteLine("Name is " + name);

Console.WriteLine("Manager Name is " + manager);

Console.WriteLine("commercials is " + commercials);

}

}

class FullTimeEmployee

{

int id;

string name;

string manager;

int salary ;

public FullTimeEmployee()

{

}

public FullTimeEmployee(int id, string name, string manager,

int salary)

{

this.id = id;

this.name = name;

this.manager = manager;

this.salary = salary;

}

public void GetDetails()

{

Console.WriteLine("Enter ID");

id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

name = Console.ReadLine();

Console.WriteLine("Enter Manager Name");

manager = Console.ReadLine();

Console.WriteLine("Enter salary");

salary = int.Parse(Console.ReadLine());

}

public void DisplayDetails()

{

Console.WriteLine("ID is " + id);

Console.WriteLine("Name is " + name);

Console.WriteLine("Manager Name is " + manager);

Console.WriteLine("Salary is " + salary);

}}

**The problem is that there is lot of duplicacy**

We can use Inheritance , as we saw that many attributes & methods are common in different type of Employees

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

class Employee

{

int id;

string name;

string manager;

public Employee()

{

}

public Employee(int id, string name, string manager

)

{

this.id = id;

this.name = name;

this.manager = manager;

}

public void GetDetails()

{

Console.WriteLine("Enter ID");

id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

name = Console.ReadLine();

Console.WriteLine("Enter Manager Name");

manager = Console.ReadLine();

}

public void DisplayDetails()

{

Console.WriteLine("ID is " + id);

Console.WriteLine("Name is " + name);

Console.WriteLine("Manager Name is " + manager);

}

}

class PartTimeEmployee : Employee

{

int commercials;

string Duration;

//public PartTimeEmployee()

//{

//}

//public PartTimeEmployee(int id, string name, string manager,

// int commercials, string duration)

//{

// this.commercials = commercials;

// this.Duration = duration;

//}

public void GetParttimeEmployeeDetails()

{

Console.WriteLine("Enter commercials");

commercials = int.Parse(Console.ReadLine());

}

public void DisplayParttimeEmployeeDetails()

{

Console.WriteLine("commercials is " + commercials);

}

}

class FullTimeEmployee : Employee

{

int salary ;

//public FullTimeEmployee()

//{

//}

//public FullTimeEmployee(int id, string name, string manager,

// int salary)

//{

// this.id = id;

// this.name = name;

// this.manager = manager;

// this.salary = salary;

//}

public void GetFullTimeEmployeeDetails()

{

Console.WriteLine("Enter salary");

salary = int.Parse(Console.ReadLine());

}

public void DisplayFullTimeEmployeeDetails()

{

Console.WriteLine("Salary is " + salary);

}}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

class Employee

{

int id;

string name;

string manager;

public Employee()

{

}

public Employee(int id, string name, string manager

)

{

this.id = id;

this.name = name;

this.manager = manager;

}

public void GetDetails()

{

Console.WriteLine("Enter ID");

id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

name = Console.ReadLine();

Console.WriteLine("Enter Manager Name");

manager = Console.ReadLine();

}

public void DisplayDetails()

{

Console.WriteLine("ID is " + id);

Console.WriteLine("Name is " + name);

Console.WriteLine("Manager Name is " + manager);

}

}

class PartTimeEmployee : Employee

{

int commercials;

string Duration;

//public PartTimeEmployee()

//{

//}

//public PartTimeEmployee(int id, string name, string manager,

// int commercials, string duration)

//{

// this.commercials = commercials;

// this.Duration = duration;

//}

public void GetParttimeEmployeeDetails()

{

Console.WriteLine("Enter commercials");

commercials = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Duration");

Duration = Console.ReadLine();

}

public void DisplayParttimeEmployeeDetails()

{

Console.WriteLine("commercials is " + commercials);

Console.WriteLine("Duration is "+ Duration);

}

}

class FullTimeEmployee : Employee

{

int salary ;

//public FullTimeEmployee()

//{

//}

//public FullTimeEmployee(int id, string name, string manager,

// int salary)

//{

// this.id = id;

// this.name = name;

// this.manager = manager;

// this.salary = salary;

//}

public void GetFullTimeEmployeeDetails()

{

Console.WriteLine("Enter salary");

salary = int.Parse(Console.ReadLine());

}

public void DisplayFullTimeEmployeeDetails()

{

Console.WriteLine("Salary is " + salary);

}

}

class Program2

{

static void Main()

{

//Console.WriteLine("Employee Objet");

//Employee employee = new Employee();

//employee.GetDetails();

//employee.DisplayDetails();

Console.WriteLine("Parttime Employee Object");

PartTimeEmployee partTimeEmployee = new PartTimeEmployee();

partTimeEmployee.GetDetails();

partTimeEmployee.GetParttimeEmployeeDetails();

partTimeEmployee.DisplayDetails();

partTimeEmployee.DisplayParttimeEmployeeDetails();

FullTimeEmployee fullTimeEmployee = new FullTimeEmployee();

fullTimeEmployee.GetDetails();

fullTimeEmployee.GetFullTimeEmployeeDetails();

fullTimeEmployee.DisplayDetails();

fullTimeEmployee.DisplayFullTimeEmployeeDetails();

}

}

Method Overriding : Redefining method of parent class in child class

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace A1

{

class Employee

{

int id;

string name;

string manager;

public Employee()

{

}

public Employee(int id, string name, string manager

)

{

this.id = id;

this.name = name;

this.manager = manager;

}

public void GetDetails()

{

Console.WriteLine("Enter ID");

id = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Name");

name = Console.ReadLine();

Console.WriteLine("Enter Manager Name");

manager = Console.ReadLine();

}

public void DisplayDetails()

{

Console.WriteLine("ID is " + id);

Console.WriteLine("Name is " + name);

Console.WriteLine("Manager Name is " + manager);

}

}

class PartTimeEmployee : Employee

{

int commercials;

string Duration;

//public PartTimeEmployee()

//{

//}

//public PartTimeEmployee(int id, string name, string manager,

// int commercials, string duration)

//{

// this.commercials = commercials;

// this.Duration = duration;

//}

// METHOD OVERRIDING

public void GetDetails()

{

**base.GetDetails(); It will call GetDetails() method of Parent tlass**

Console.WriteLine("Enter commercials");

commercials = int.Parse(Console.ReadLine());

Console.WriteLine("Enter Duration");

Duration = Console.ReadLine();

}

public void DisplayDetails()

{

base.DisplayDetails();

Console.WriteLine("commercials is " + commercials);

Console.WriteLine("Duration is " + Duration);

}

}

class FullTimeEmployee : Employee

{

int salary;

//public FullTimeEmployee()

//{

//}

//public FullTimeEmployee(int id, string name, string manager,

// int salary)

//{

// this.id = id;

// this.name = name;

// this.manager = manager;

// this.salary = salary;

//}

public void GetDetails()

{

base.GetDetails();

Console.WriteLine("Enter salary");

salary = int.Parse(Console.ReadLine());

}

public void DisplayDetails()

{

base.DisplayDetails();

Console.WriteLine("Salary is " + salary);

}

}

class Program2

{

static void Main()

{

//Console.WriteLine("Employee Objet");

//Employee employee = new Employee();

//employee.GetDetails();

//employee.DisplayDetails();

Console.WriteLine("Parttime Employee Object");

PartTimeEmployee partTimeEmployee = new PartTimeEmployee();

partTimeEmployee.GetDetails();

partTimeEmployee.DisplayDetails();

FullTimeEmployee fullTimeEmployee = new FullTimeEmployee();

fullTimeEmployee.GetDetails();

fullTimeEmployee.DisplayDetails();

}

}

}

OVERRIDING vs OVERLOADING

Overriding is only allowed in inheritance

Overloading can be in same class or in derived class too

In Overriding, the signature same, if you change signature , it will become overloading

**Constructors in Inheritance**