|  |  |
| --- | --- |
| using System;  namespace ClassLibrary  {  public abstract class Course  {  string cName;  string[] topic = new string[10];  public Course() { }    public Course(string cName)  {  this.cName = cName;  }  public string CName { get => cName; set => cName = value; }  public abstract void AddTopic(params string[] a);    public virtual void Print()  {  Console.WriteLine(" Course:"+cName);    }  }  } | using System;  namespace ClassLibrary  {  public class OffsiteCourse:Course  {  string town;  string[] topic = new string[10];  public OffsiteCourse() { }  public OffsiteCourse(string name, string tw):base(name)  {  town = tw;  }  public override void AddTopic( params string[] a)  {  for (int i = 0; i < a.Length; i++)  {  if (topic[i] == null)  topic[i] = a[i];  }  }  public override void Print()  {  Console.WriteLine("\nOffsiteCourse-------------->");  Console.WriteLine(" Course:" + base.CName);  Console.WriteLine(" Town:" + town);  Console.Write(" Topic:");  Console.Write("\n ");  for (int i = 0; i < topic.Length; i++)  {  if (topic[i] != null)  {  Console.Write(topic[i] + ",");  }  }  }  }  } |

|  |  |
| --- | --- |
| using System;  namespace ClassLibrary  {  public class LocalCourse : Course  {  string labNumber;  string[] topic = new string[10];  public LocalCourse()  { }  public LocalCourse(string name, string lab) : base(name)  {  labNumber = lab;  }  public override void AddTopic( params string[] a)  {  for (int i = 0; i < a.Length; i++)  {  if (topic[i] == null)  topic[i] = a[i];  }  }  public override void Print()  {  Console.WriteLine("\nCampusCourse-------------->");  Console.WriteLine(" Course:" + base.CName);  Console.WriteLine(" LabNumber:" + labNumber);  Console.Write(" Topic:");  Console.Write("\n ");  for (int i = 0; i < topic.Length; i++)  {  if (topic[i] != null)  {  Console.Write(topic[i]+",");  }  }  }  }  } | using System;  namespace ClassLibrary  {  public class Person  {  string pName;  string id;  public string Id { get => id; set => id = value; }  public string PName { get => pName; set => pName = value; }  public Person() { }  public Person(string name,string id)  {  this.pName = name;  this.id = id;  }  }  } |

|  |  |
| --- | --- |
| using System;  namespace ClassLibrary  {  public class Teacher:Person  {  Course[] course = new Course[5];  public Teacher() { }  public Teacher(string nm,string id):base(nm,id) { }  public string TName  {  get { return base.PName; }  }  public string TId  {  get { return base.Id; }  }  public void AddCourse(params Course[] a)  {  for (int i = 0; i < a.Length; i++)  {  if (course[i] == null)  {  course[i] = a[i];  }  }  }  public Course GetCourse(int i)  {  return course[i];  }  public void Print()  {  Console.WriteLine("Teacher:"+base.PName);  Console.WriteLine("Teacher Id:"+base.Id);  }  }  } | using System;  namespace ClassLibrary  {  public class Student:Person  {  Teacher[] teachers = new Teacher[4];  public Student(string name,string id):base(name,id)  { }  public string SName  {  get { return base.PName; }  }  public string SId  {  get { return base.Id; }  }  public void AddTeacher(Teacher t)  {  for(int i=0;i<teachers.Length;i++)  {  if (teachers[i] == null)  {  teachers[i] = t;  break;  }  }  }  public Teacher GetTeacher(int i)  {  return teachers[i];  }  }  } |

This Class contains Main Mathod:

using ClassLibrary;

using System;

namespace \_18\_36449\_1

{

class Start

{

static void Main(string[] args)

{

Course c1 = new LocalCourse("OOP2", "1234");

c1.AddTopic("Using Classes and Objects", "Defining Classes", "OOP Principles", "Property");

Course c2 = new LocalCourse("OOAD", "5678");

c2.AddTopic("Introduction to Modelling", "Introduction to UML", "Use Case Diagram", "Class Diagram");

Course c3 = new OffsiteCourse("Cyber Security", "Sylhet");

c3.AddTopic("Security", "Encryption", "Algorithm", "Cyber Ethics");

Teacher t1 = new Teacher("Victor Stany Rozario", "15-34939-1");

t1.AddCourse(c1, c2);

Teacher t2 = new Teacher("SUPTA RICHARD PHILIP", "14-54463-2");

t2.AddCourse(c3);

Student s1 = new Student("Borno", "18-36449-1");

s1.AddTeacher(t2);

s1.AddTeacher(t1);

TeacherDetails(t1);

TeacherDetails(t2);

StudentDetails(s1);

}

public static void TeacherDetails (Teacher t)

{

if (t.TName != null)

{

Console.WriteLine("Teacher:" + t.TName);

Console.WriteLine("Teacher Id:" + t.Id);

}

for (int i = 0; i < 2; i++)

{

if (t.GetCourse(i) != null)

t.GetCourse(i).Print();

}

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

}

public static void StudentDetails(Student s)

{

if (s.SName != null)

{

Console.WriteLine("Student:" + s.SName);

Console.WriteLine("Student Id:" + s.SId);

}

for (int i = 0; i < 2; i++)

{

if (s.GetTeacher(i) != null)

s.GetTeacher(i).Print();

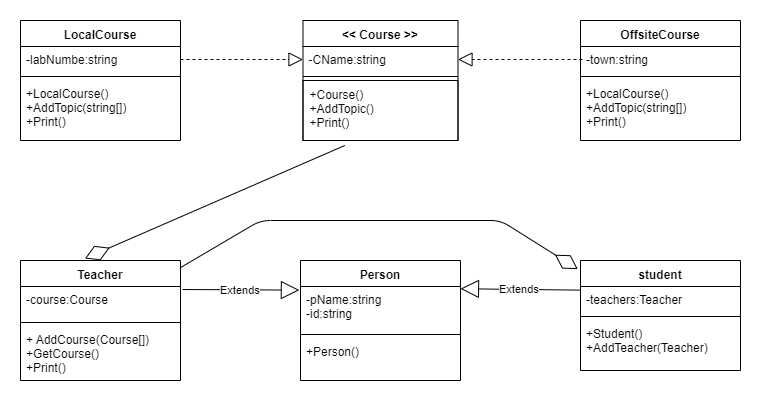
}

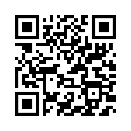
}

}

}

Class Diagram



GitHub Profile: <https://github.com/Born0> or scan this: Assignment path: <https://github.com/Born0/SE-assignment>