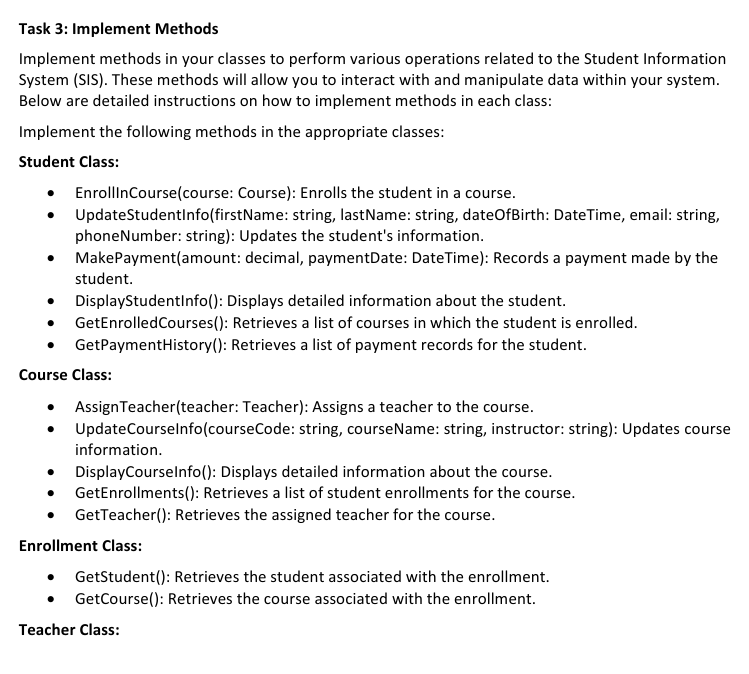
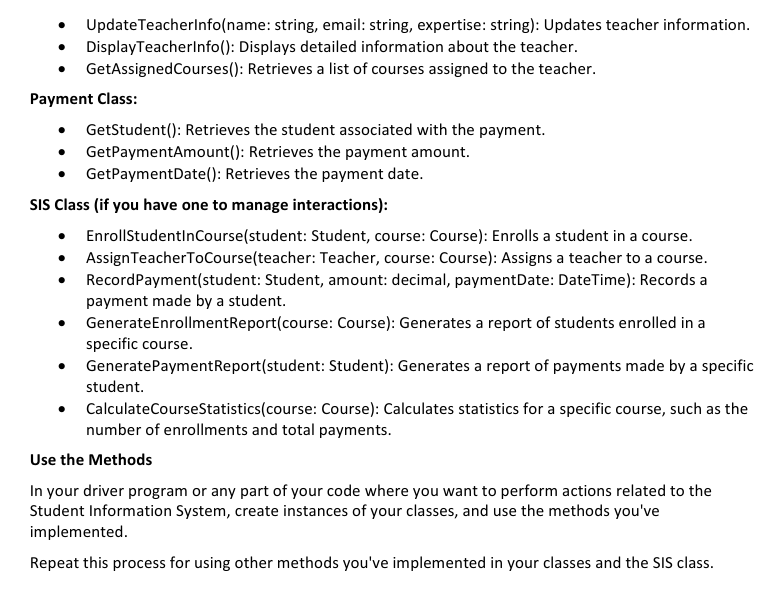
**Task 3 Question**





Made a few changes from the previous task as required concepts have been introduced in the sessions.

**Output screenshots are at the end.**

**The Student Class**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SIS

{

internal class Student

{

public int StudentID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public DateTime DateOfBirth { get; set; }

public string Email { get; set; }

public string PhoneNumber { get; set; }

public List<Enrollment> EnrolledCourses { get; set; }

public List<Payment> PaymentHistory { get; set; }

private static int enroll\_id = 0; *//For ID increment*

private static int payment\_id = 0;

public Student(int studentID, string firstName, string lastName, DateTime dateOfBirth, string email, string phoneNumber)

{

StudentID = studentID;

FirstName = firstName;

LastName = lastName;

DateOfBirth = dateOfBirth;

Email = email;

PhoneNumber = phoneNumber;

EnrolledCourses = new List<Enrollment>();

PaymentHistory = new List<Payment>();

}

public void EnrollInCourse(Course course)

{

Enrollment = new Enrollment(++enroll\_id, this, course, DateTime.Now);

EnrolledCourses.Add(enrollment); *//Updating the list in student class*

course.EnrolledCourses.Add(enrollment); *//Updating the list in course class*

}

public void UpdateStudentInfo(string firstName, string lastName, DateTime dateOfBirth, string email, string phoneNumber)

{

FirstName = firstName;

LastName = lastName;

DateOfBirth = dateOfBirth;

Email = email;

PhoneNumber = phoneNumber;

}

public void MakePayment(decimal amount, DateTime paymentDate)

{

Payment = new Payment(payment\_id++, this, amount, paymentDate);

PaymentHistory.Add(payment); *//Updating the payment list*

}

public void DisplayStudentInfo()

{

Console.WriteLine($"Student ID: {StudentID}\nName: {FirstName} {LastName}\nDOB: {DateOfBirth.ToShortDateString()}\nEmail: {Email}\nPhone: {PhoneNumber}\n");

}

public List<string> GetEnrolledCourses()

{

List<string> courses = new List<string>();

foreach (var enrollment in EnrolledCourses)

{

courses.Add(enrollment.CourseID.CourseName);

}

return courses;

}

public List<Payment> GetPaymentHistory()

{

return PaymentHistory;

}

}

}

**The Course Class**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SIS

{

internal class Course

{

public int CourseID { get; set; }

public string CourseName { get; set; }

public string CourseCode { get; set; }

public string InstructorName { get; set; }

public List<Enrollment> EnrolledCourses { get; set; }

public Course(int courseID, string courseName, string courseCode, string instructorName)

{

CourseID = courseID;

CourseName = courseName;

CourseCode = courseCode;

InstructorName = instructorName;

EnrolledCourses = new List<Enrollment>();

}

public void AssignTeacher(Teacher teacher)

{

InstructorName = teacher.FirstName + " " + teacher.LastName;

teacher.AssignedCourses.Add(this); *//Updates list in Teacher class*

}

public void UpdateCourseInfo(string courseCode, string courseName, string instructor)

{

CourseName = courseName;

CourseCode = courseCode;

InstructorName = instructor;

}

public void DisplayCourseInfo()

{

Console.WriteLine($"Course ID: {CourseID}\nName: {CourseName}\nCode: {CourseCode}\nInstructor: {InstructorName}\n\n");

}

public List<Enrollment> GetEnrollments()

{

return EnrolledCourses;

}

public string GetTeacher()

{

return InstructorName;

}

}

}

**The Enrollment Class**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SIS

{

internal class Enrollment

{

public int EnrollmentID { get; set; }

public Student StudentID { get; set; }

public Course CourseID { get; set; }

public DateTime EnrollmentDate { get; set; }

public Enrollment(int enrollmentID, Student studentID, Course courseID, DateTime enrollmentDate)

{

EnrollmentID = enrollmentID;

StudentID = studentID;

CourseID = courseID;

EnrollmentDate = enrollmentDate;

}

public Student GetStudent()

{

return StudentID;

}

public Course GetCourse()

{

return CourseID;

}

}

}

**The Teacher Class**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SIS

{

internal class Teacher

{

public int TeacherID { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public string Email { get; set; }

public string Expertise { get; set; }

public List<Course> AssignedCourses { get; set; }

public Teacher(int teacherID, string firstName, string lastName, string email, string expertise)

{

TeacherID = teacherID;

FirstName = firstName;

LastName = lastName;

Email = email;

Expertise = expertise;

AssignedCourses = new List<Course>();

}

public void UpdateTeacherInfo(string name, string email, string expertise)

{

FirstName = name.Split(' ')[0];

LastName = name.Split(' ')[1];

Email = email;

Expertise = expertise;

}

public void DisplayTeacherInfo()

{

Console.WriteLine($"Teacher ID: {TeacherID}\nName: {FirstName} {LastName}\nEmail: {Email}\n\n");

}

public List<Course> GetAssignedCourse()

{

return AssignedCourses;

}

}

}

**The Payment Class**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SIS

{

internal class Payment

{

public int PaymentID { get; set; }

public Student StudentID { get; set; }

public decimal Amount { get; set; }

public DateTime PaymentDate { get; set; }

public Payment(int paymentID, Student studentID, decimal amount, DateTime paymentDate)

{

PaymentID = paymentID;

StudentID = studentID;

Amount = amount;

PaymentDate = paymentDate;

}

public Student GetStudent()

{

return StudentID;

}

public decimal GetPaymentAmount()

{

return Amount;

}

public DateTime GetPaymentDate()

{

return PaymentDate;

}

}

}

**The Main Class**

namespace SIS

{

internal class Program

{

static void Main(string[] args)

{

Task\_3 tc = new Task\_3();

tc.TestStudent();

tc.TestCourse();

tc.TestEnroll();

tc.TestTeacher();

tc.TestPayment();

}

}

}

**The Sis Class**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using SIS.CoustomExceptions;

namespace SIS

{

internal class Sis

{

public void EnrollStudentInCourse(Student student, Course course)

{

student.EnrollInCourse(course);

}

public void AssignTeacherToCourse(Teacher teacher, Course course)

{

course.AssignTeacher(teacher);

}

public void RecordPayment(Student student, decimal amount, DateTime paymentDate)

{

student.MakePayment(amount, paymentDate);

}

public List<string> GenerateEnrollmentReport(Course course)

{

List<string> enrolledStudents = new List<string>();

foreach (var enrollment in course.GetEnrollments())

{

enrolledStudents.Add(enrollment.StudentID.FirstName + " " + enrollment.StudentID.LastName);

}

return enrolledStudents;

}

public List<string> GeneratePaymentReport(Student student)

{

List<string> paymentReport = new List<string>();

foreach (var payment in student.GetPaymentHistory())

{

paymentReport.Add($"Amount: {payment.Amount}, Date: {payment.PaymentDate}");

}

return paymentReport;

}

public (int, decimal) CalculateCourseStatistics(Course course)

{

int enrollmentCount = 0;

decimal totalPayments = 0;

foreach (var enrollment in course.GetEnrollments())

{

enrollmentCount++;

foreach (var payment in enrollment.StudentID.GetPaymentHistory())

{

totalPayments += payment.Amount;

}

}

return (enrollmentCount, totalPayments);

}

}

}

**The Task\_3 Class [Tester class to get the outputs]**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SIS

{

internal class Task\_3

{

Sis system = new Sis();

Student student1 = new Student(101, "Caitlyn", "Kiraman", new DateTime(2000, 5, 15), "cait@mail.com", "1234567890");

Student student2 = new Student(102, "David", "Johnson", new DateTime(1999, 8, 22), "dave@mail.com", "9876543210");

Student student3 = new Student(107, "April", "Smith", new DateTime(2000, 10, 1), "april@mail.com", "9591529583");

Student student4 = new Student(100, "Beth", "Michel", new DateTime(1999, 5, 12), "beth@mail.com", "9754283655");

Course math = new Course(201, "Algebra", "MATH101", "");

Course bio = new Course(201, "Biology", "BIO401", "");

Course history = new Course(201, "History", "HIS74", "");

Teacher teacher1 = new Teacher(1, "Jayce", "Norten", "jay@mail.com", "Algebra");

Teacher teacher2 = new Teacher(1, "Haley", "Fields", "haley@mail.com", "History");

Teacher teacher3 = new Teacher(1, "Roy", "Wells", "roy@mail.com", "Biology");

public void TestStudent()

{

student1.EnrollInCourse(bio);

student2.EnrollInCourse(history);

student3.EnrollInCourse(math);

student3.EnrollInCourse(history);

student4.EnrollInCourse(math);

student1.MakePayment(60000, new DateTime(2025, 1, 5));

student2.MakePayment(50000, new DateTime(2025, 1, 24));

student3.MakePayment(65000, new DateTime(2025, 2, 15));

student3.MakePayment(70000, new DateTime(2025, 2, 15));

student4.MakePayment(65000, new DateTime(2025, 2, 3));

Console.WriteLine($"Student Info : \n");

student2.DisplayStudentInfo();

Console.WriteLine($"Student Info : \n");

student4.DisplayStudentInfo();

List<string> enrolledcourses = student3.GetEnrolledCourses();

Console.WriteLine("April's Enrolled courses :\n");

foreach( string enrolledCourse in enrolledcourses )

{

Console.WriteLine(enrolledCourse);

}

List<Payment> paymenthistory = student3.GetPaymentHistory();

Console.WriteLine("\nApril's Payment history :\n");

foreach (Payment paymentHistory in paymenthistory)

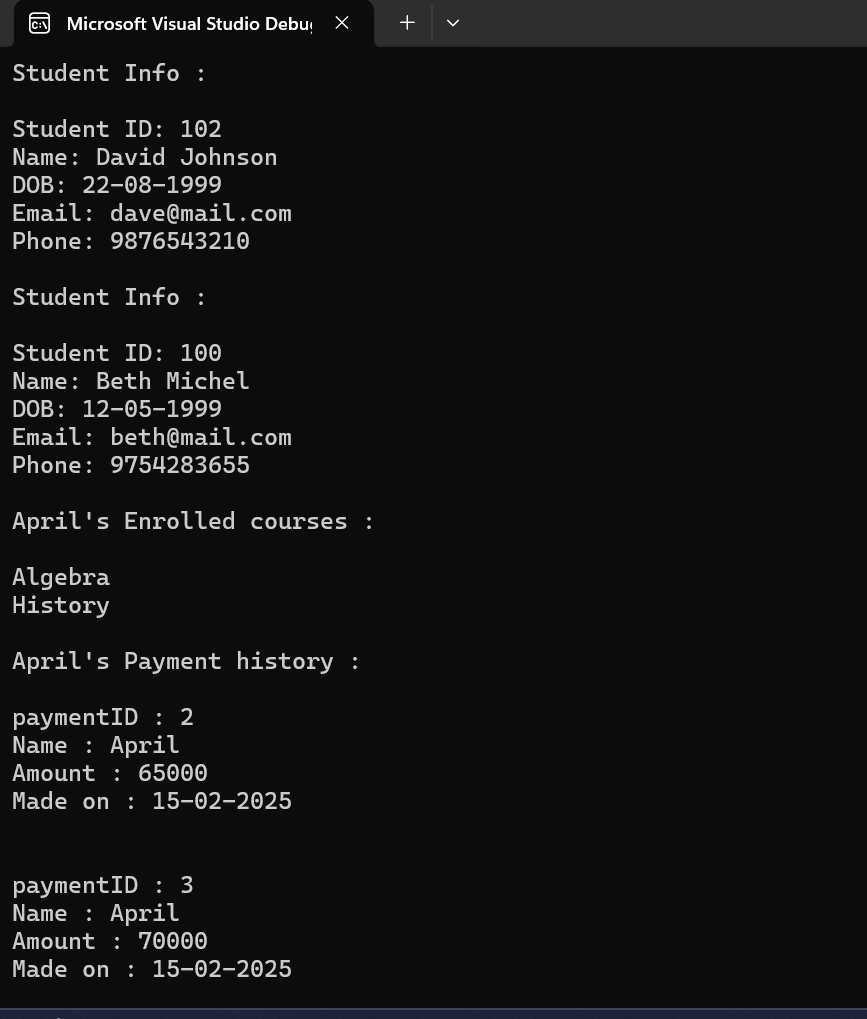
{

Console.WriteLine($"paymentID : {paymentHistory.PaymentID}\nName : {paymentHistory.StudentID.FirstName}\nAmount : {paymentHistory.Amount}\nMade on : {paymentHistory.PaymentDate.ToShortDateString()}\n\n");

}

}

*The output for methods from student class*



public void TestCourse()

{

math.AssignTeacher(teacher2);

history.AssignTeacher(teacher3);

bio.AssignTeacher(teacher1);

bio.DisplayCourseInfo();

student1.EnrollInCourse(bio);

student2.EnrollInCourse(history);

student3.EnrollInCourse(math);

student3.EnrollInCourse(history);

student4.EnrollInCourse(math);

List<Enrollment> courseEnrolled = math.GetEnrollments();

Console.WriteLine("The enrollments for the course Math :\n");

foreach( Enrollment in courseEnrolled )

{

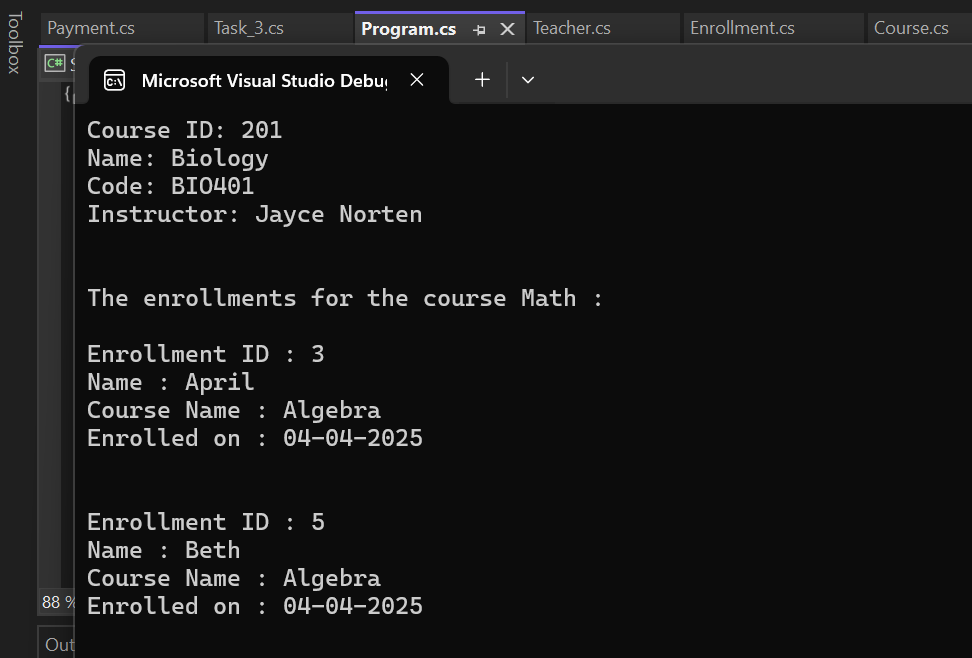
Console.WriteLine($"Enrollment ID : {enrollment.EnrollmentID}\nName : {enrollment.StudentID.FirstName}\nCourse Name : {enrollment.CourseID.CourseName}\nEnrolled on : {enrollment.EnrollmentDate.ToShortDateString()}\n\n");

}

Console.WriteLine($"The instructor for the course History is : {history.GetTeacher()}");

}

*The output for methods from course class*



public void TestEnroll()

{

student1.EnrollInCourse(bio);

student2.EnrollInCourse(history);

student3.EnrollInCourse(math);

student3.EnrollInCourse(history);

student4.EnrollInCourse(math);

student1.MakePayment(60000, new DateTime(2025, 1, 5));

student2.MakePayment(50000, new DateTime(2025, 1, 24));

student3.MakePayment(65000, new DateTime(2025, 2, 15));

student3.MakePayment(70000, new DateTime(2025, 2, 15));

student4.MakePayment(65000, new DateTime(2025, 2, 3));

foreach (var enrolls in student3.EnrolledCourses)

{

Console.WriteLine($"The enrollment number {enrolls.EnrollmentID} : made by student {enrolls.StudentID.FirstName}");

}

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

Console.WriteLine($"The enrollments made by student {student3.FirstName} :");

foreach(var enrolls in student3.EnrolledCourses )

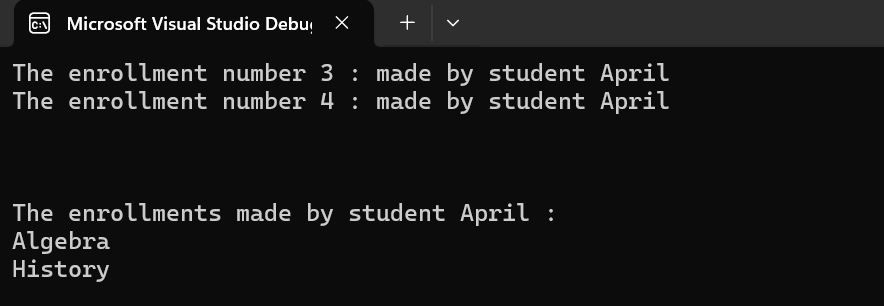
{

Console.WriteLine(enrolls.CourseID.CourseName);

}

}

*The output for methods from enrollment class*



public void TestTeacher()

{

teacher2.DisplayTeacherInfo();

math.AssignTeacher(teacher2);

history.AssignTeacher(teacher3);

bio.AssignTeacher(teacher3);

Console.WriteLine($"The courses handled by {teacher3.FirstName} :");

List<Course> course = teacher3.GetAssignedCourse();

foreach (var assigned in course)

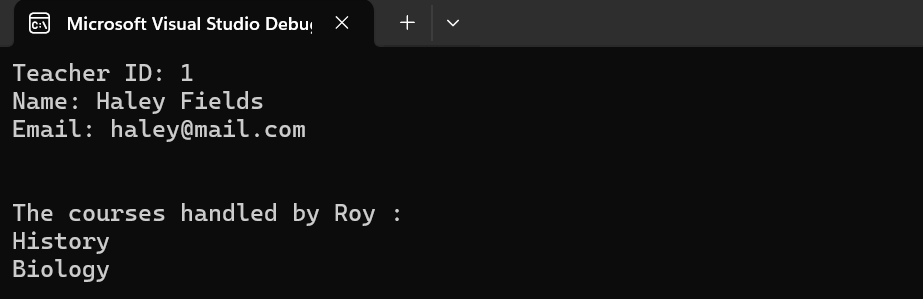
{

Console.WriteLine(assigned.CourseName);

}

}

*The output for methods from teacher class*



public void TestPayment()

{

student1.EnrollInCourse(bio);

student2.EnrollInCourse(history);

student3.EnrollInCourse(math);

student3.EnrollInCourse(history);

student4.EnrollInCourse(math);

student1.MakePayment(60000, new DateTime(2025, 1, 5));

student2.MakePayment(50000, new DateTime(2025, 1, 24));

student3.MakePayment(65000, new DateTime(2025, 2, 15));

student3.MakePayment(70000, new DateTime(2025, 2, 15));

student4.MakePayment(65000, new DateTime(2025, 2, 3));

foreach (var pay in student3.PaymentHistory)

{

Console.WriteLine($"The payment id {pay.PaymentID} was made by {pay.StudentID.FirstName}");

}

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

foreach (var pay in student3.PaymentHistory)

{

Console.WriteLine($"The student {pay.StudentID.FirstName} has paid {pay.Amount}");

}

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

foreach (var pay in student3.PaymentHistory)

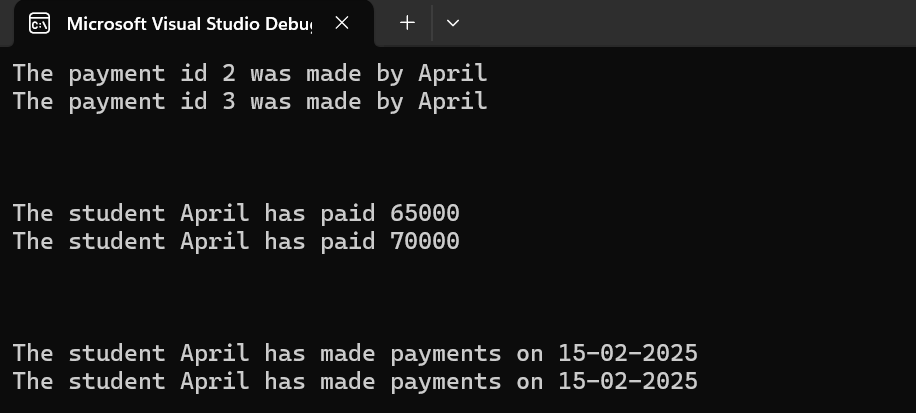
{

Console.WriteLine($"The student {pay.StudentID.FirstName} has made payments on {pay.PaymentDate.ToShortDateString()}");

}

}

*The output for methods from payment class*



public void TestSis()

{

student1.EnrollInCourse(bio);

student2.EnrollInCourse(history);

student3.EnrollInCourse(math);

student3.EnrollInCourse(history);

student4.EnrollInCourse(math);

student1.MakePayment(60000, new DateTime(2025, 1, 5));

student2.MakePayment(50000, new DateTime(2025, 1, 24));

student3.MakePayment(65000, new DateTime(2025, 2, 15));

student3.MakePayment(70000, new DateTime(2025, 2, 15));

student4.MakePayment(65000, new DateTime(2025, 2, 3));

Sis info = new Sis();

Console.WriteLine($"The students enrolled in {math.CourseName} are :");

List<string> report = info.GenerateEnrollmentReport(math);

foreach(string name in report)

{

Console.WriteLine(name);

}

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

Console.WriteLine($"The student {student2.FirstName} has made the payments : ");

List<string> payments = info.GeneratePaymentReport(student2);

foreach (string rec in payments)

{

Console.WriteLine(rec);

}

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

(int enrollmentCount, decimal totalPayments) = info.CalculateCourseStatistics(math);

Console.WriteLine($"The stats for the course {math.CourseName}:");

Console.WriteLine($"Total Enrollments: {enrollmentCount}");

Console.WriteLine($"Total Payments Collected: {totalPayments}");

}

*The output from Sis class*

