```cs
using System.Collections.Generic;
using System.Net.Mail;
using static backendDay8___Collections_non___generic_.Program;

namespace backendDay8___Collections_non___generic_
{
    1 reference
    internal class Program
    {
        14 references
        public class InputHelper
        {
            1 reference
            public static int CheckMenu(string prompt) //method pang check if tama ba yung operation na pinili
            {
                int value = 0;
                bool valid = false;
                do
                {
                    try
                    {
                        Console.WriteLine(prompt);
                        string input = Console.ReadLine();

                        if (!int.TryParse(input, out value) || value < 1 || value > 6)
                        {
                            throw new ArgumentException("Invalid operation. Must be between 1 and 6");
                        }

                        valid = true;
                    }
                    catch (ArgumentException ex)
                    {
                        Console.WriteLine(ex.Message);
                    }

                } while (!valid);

                return value;
            }
            4 references
            public static string CheckString(string prompt)
            {
                bool valid = false;
                string value = "";

                do
                {
                    try
                    {
                        Console.WriteLine(prompt);
                        value = Console.ReadLine();

                        if (string.IsNullOrWhiteSpace(value))
                        {
                            throw new ArgumentException("Invalid input. It cannot be empty.");
                        }
                        //dapat walang number sa name or sa string like sa program
                        if (value.Any(char.IsDigit))
                        {
                            throw new ArgumentException("Invalid input. Name cannot contain numbers.");
                        }

                        valid = true;
                    }
                    catch (ArgumentException ex)
                    {
                        Console.WriteLine(ex.Message);
                    }

                } while (!valid);

                return value;
            }
```

```
2 references
public static int CheckAge(string prompt)//eto naman sa age
{
    int value = 0;
    bool valid = false;
    do
    {
        try
        {
            Console.WriteLine(prompt);
            string input = Console.ReadLine();

            if (!int.TryParse(input, out value) || value < 18 || value > 40)
            {
                throw new ArgumentException("Invalid age. Must be between 18 and 40.");
            }

            valid = true;
        }
        catch (ArgumentException ex)
        {
            Console.WriteLine(ex.Message);
        }

    } while (!valid);

    return value;
}
```

```
4 references
public static string CheckStudID(string prompt)
{
    string value = "";
    bool valid = false;

    do
    {
        try
        {
            Console.WriteLine(prompt);
            value = Console.ReadLine();

            //limited to 5 digits lang and dapat numbers lnag siya.
            if (string.IsNullOrWhiteSpace(value) || value.Length != 5 || !value.All(char.IsDigit))
            {
                throw new ArgumentException("Invalid Student ID. It must be exactly 5 digits (e.g., 01234).");
            }

            valid = true;
        }
        catch (ArgumentException ex)
        {
            Console.WriteLine(ex.Message);
        }

    } while (!valid);

    return value;
}
```

```
3 references
public static string ConfirmationInput(string prompt)
{
    Console.WriteLine(prompt);
    string value = Console.ReadLine().Trim().ToLower();
    while(value != "yes" && value != "no")
    {
        Console.WriteLine($"Invalid input.{prompt}");
        value = Console.ReadLine().Trim().ToLower();
    }

    return value;
}
}
```

```
public class Student
{
    private string name;
    private string studID;
    private int age;
    private string program;

    6 references
    public string Name
    {
        get { return name; }
        set { name = value; }
    }

    8 references
    public string StudID
    {
        get { return studID; }
        set { studID = value; }
    }

    6 references
    public int Age
    {
        get { return age; }
        set { age = value; }
    }

    6 references
    public string Program
    {
        get { return program; }
        set { program = value; }
    }
}
```

```
public class ManageStudent
{
    1 reference
    public static void AddStudent(List<Student> students)
    {
        string useAgain = "no";
        do
        {
            string tempID;
            // Check if may kapareha na id
            do
            {
                tempID = InputHelper.CheckStudID("Enter student ID (ex: 00102): ");
                if (students.Any(s => s.StudID == tempID))
                {
                    Console.WriteLine("That Student ID already exists. Please enter a different ID.");
                    tempID = null;
                }
            } while (tempID == null);

            Student student = new Student();
            student.Name = InputHelper.CheckString("Enter student name: ");
            student.StudID = tempID;
            student.Age = InputHelper.CheckAge("Enter age (18 - 40): ");
            student.Program = InputHelper.CheckString("Enter course/program (ex: BSCS): ").ToUpper();

            students.Add(student);

            Console.Write("Add another student? (yes/no): ");
            useAgain = Console.ReadLine()?.Trim().ToLower();
        } while (useAgain == "yes");
    }
```

```
    1 reference
    public static void DisplayAllStudents(List<Student> students)
    {
        if (students.Count == 0)
        {
            Console.WriteLine("No student records found.");
            return;
        }

        Console.WriteLine("\n--- All Student Records ---");
        foreach (Student student in students)
        {
            Console.WriteLine($"Name: {student.Name}");
            Console.WriteLine($"Student ID: {student.StudID}");
            Console.WriteLine($"Age: {student.Age}");
            Console.WriteLine($"Program: {student.Program}");
            Console.WriteLine("--------------------------");
        }
    }
```

```
public static void EditStudent(List<Student> students)
{
    string tempID = InputHelper.CheckStudID("Please insert student ID to edit: ");
    bool found = false;

    foreach (Student student in students)
    {
        if (string.Equals(student.StudID, tempID, StringComparison.OrdinalIgnoreCase))
        {
            Console.WriteLine("\n--- Current Student Information ---");
            Console.WriteLine($"Name: {student.Name}");
            Console.WriteLine($"Student ID: {student.StudID}");
            Console.WriteLine($"Age: {student.Age}");
            Console.WriteLine($"Program: {student.Program}");

            Console.WriteLine("\n--- Enter New Information ---");

            student.Name = InputHelper.CheckString("Enter new name: ");
            student.Age = InputHelper.CheckAge("Enter new age (18-40): ");
            student.Program = InputHelper.CheckString("Enter new program: ").ToUpper();

            Console.WriteLine("Student information updated successfully.");
            found = true;
            break;
        }
    }

    if (!found)
    {
        Console.WriteLine("Student with the given ID was not found.");
    }
}
```

```
1 reference
public static void DisplayStudent(List<Student> students)
{
    string tempID = InputHelper.CheckStudID("Please insert student ID to display: ");
    bool found = false;

    foreach (Student student in students)
    {
        if (string.Equals(student.StudID, tempID, StringComparison.OrdinalIgnoreCase))
        {
            Console.WriteLine("\n--- Current Student Information ---");
            Console.WriteLine($"Name: {student.Name}");
            Console.WriteLine($"Student ID: {student.StudID}");
            Console.WriteLine($"Age: {student.Age}");
            Console.WriteLine($"Program: {student.Program}");

            found = true;
            break;
        }
    }

    if (!found)
    {
        Console.WriteLine("Student with the given ID was not found.");
    }
}
```

```
    public static void DeleteStudent(List<Student> students)
    {
        string tempID = InputHelper.CheckStudID("Please insert student ID to edit: ");
        bool found = false;
        string confirm = "";

        for (int i = 0; i < students.Count; i++)
        {
            if (string.Equals(students[i].StudID, tempID, StringComparison.OrdinalIgnoreCase))
            {
                Console.WriteLine("\n--- Current Student Information ---");
                Console.WriteLine($"Name: {students[i].Name}");
                Console.WriteLine($"Student ID: {students[i].StudID}");
                Console.WriteLine($"Age: {students[i].Age}");
                Console.WriteLine($"Program: {students[i].Program}");

                confirm = InputHelper.ConfirmationInput("\nDo you want to delete this student?(yes/no): ");

                if(confirm == "yes")
                {
                    students.RemoveAt(i);
                    Console.WriteLine("Student removed successfully.");
                }
                else
                {
                    Console.WriteLine("Operation canceled.");
                }
                found = true;
                break;
            }
        }

        if (!found)
        {
            Console.WriteLine("Student with the given ID was not found.");
        }
    }
}
```

```
        static void Main(string[] args)
        {
            string useAgain = "no";
            int operation;

            List<Student> students = new List<Student>(); //gumamit ng list since di pwedeng array kase fixed sized siya and mas madali mag or delete here

            do
            {
                Console.WriteLine("--------\t Student Management System\t--------");
                Console.WriteLine("1. Add Student Information\n2. Display All Student Records\n3. Edit Student\n4. Display a Student\n5. Remove Student\n6. Exit");
                operation = InputHelper.CheckMenu("Please select an operation (1 - 6): ");

                switch (operation)
                {
                    case 1:
                        ManageStudent.AddStudent(students);
                        break;
                    case 2:
                        ManageStudent.DisplayAllStudents(students);
                        break;
                    case 3:
                        ManageStudent.EditStudent(students);
                        break;
                    case 4:
                        ManageStudent.DisplayStudent(students);
                        break;
                    case 5:
                        ManageStudent.DeleteStudent(students);
                        break;
                    case 6:
                        useAgain = InputHelper.ConfirmationInput("Are you sure you want to exit? ('yes' / 'no'): ");
                        break;
                    default:
                        Console.WriteLine("Invalid operation selected. Please try again.");
                        break;
                }

                useAgain = InputHelper.ConfirmationInput("Do you want to perform another operation?('yes' / 'no'): ");

            } while (useAgain == "yes");
        }
    }
}
```

OUTPUTS

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
```

ADD STUDENT

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
1
Enter student ID (ex: 00102):
23293
Enter student name:
John
Enter age (18 - 40):
21
Enter course/program (ex: BSCS):
BSCS
Add another student? (yes/no): yes
Enter student ID (ex: 00102):
21211
Enter student name:
Gianna
Enter age (18 - 40):
20
Enter course/program (ex: BSCS):
BSN
Add another student? (yes/no): yes
Enter student ID (ex: 00102):
20321
Enter student name:
Rich
Enter age (18 - 40):
20
Enter course/program (ex: BSCS):
BSIT
Add another student? (yes/no): no
Do you want to perform another operation?('yes' / 'no'):
no

D:\NICO\WPH\BACKEND-codes\backendDay8 - Collections\bin\D
Press any key to close this window . . .
```

DISPLAY ALL STUDENTS

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
2

--- All Student Records ---
Name: John
Student ID: 23293
Age: 21
Program: BSCS
--------------------------
Name: Gianna
Student ID: 21211
Age: 20
Program: BSN
--------------------------
Name: Rich
Student ID: 20321
Age: 20
Program: BSIT
--------------------------
Do you want to perform another operation?('yes' / 'no'):
```

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
2
No student records found.
Do you want to perform another operation?('yes' / 'no'):
```

```
--- All Student Records ---
Name: John Rich
Student ID: 23032
Age: 21
Program: BSCS
--------------------------
```

EDIT STUDENT

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
3
Please insert student ID to edit:
23293

--- Current Student Information ---
Name: John
Student ID: 23293
Age: 21
Program: BSCS

--- Enter New Information ---
Enter new name:
Nicolas
Enter new age (18-40):
22
Enter new program:
BSCS
Student information updated successfully.
Do you want to perform another operation?('yes' / 'no'):
yes
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
2

--- All Student Records ---
Name: Nicolas
Student ID: 23293
Age: 22
Program: BSCS
---------------------------
Name: Gianna
Student ID: 21211
Age: 20
Program: BSN
---------------------------
Name: Rich
Student ID: 20321
Age: 20
Program: BSIT
---------------------------
Do you want to perform another operation?('yes' / 'no'):
```

DISPLAY A STUDENT

```
--- All Student Records ---
Name: Nicolas
Student ID: 23293
Age: 22
Program: BSCS
--------------------------
Name: Gianna
Student ID: 21211
Age: 20
Program: BSN
--------------------------
Name: Rich
Student ID: 20321
Age: 20
Program: BSIT
--------------------------
Do you want to perform another operation?('yes' / 'no'):
yes
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
4
Please insert student ID to display:
21211

--- Current Student Information ---
Name: Gianna
Student ID: 21211
Age: 20
Program: BSN
Do you want to perform another operation?('yes' / 'no'):
yes
```

DELETE A STUDENT

```
Please insert student ID to edit:
23293

--- Current Student Information ---
Name: Nicolas
Student ID: 23293
Age: 22
Program: BSCS

Do you want to delete this student?(yes/no):
yes
Student removed successfully.
Do you want to perform another operation?('yes' / 'no'):
yes
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
2

--- All Student Records ---
Name: Gianna
Student ID: 21211
Age: 20
Program: BSN
--------------------------
Name: Rich
Student ID: 20321
Age: 20
Program: BSIT
--------------------------
Do you want to perform another operation?('yes' / 'no'):
```

EXIT SYSTEM

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
6
Are you sure you want to exit? ('yes' / 'no'):
no
Do you want to perform another operation?('yes' / 'no'):
yes
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
6
Are you sure you want to exit? ('yes' / 'no'):
yes
Do you want to perform another operation?('yes' / 'no'):
no

D:\NICO\WPH\BACKEND-codes\backendDay8 - Collections\bin\Debug\
Press any key to close this window . . .
```

STUDENT VALIDATION – NO SAME STUDENT ID

```
--------          Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
1
Enter student ID (ex: 00102):
23311
Enter student name:
John
Enter age (18 - 40):
20
Enter course/program (ex: BSCS):
BSCS
Add another student? (yes/no): yes
Enter student ID (ex: 00102):
23311
That Student ID already exists. Please enter a different ID.
Enter student ID (ex: 00102):
12142
Enter student name:
John
Enter age (18 - 40):
21
Enter course/program (ex: BSCS):
BSCS
```

```
--- All Student Records ---
Name: John
Student ID: 23311
Age: 20
Program: BSCS
---------------------------
Name: John
Student ID: 12142
Age: 21
Program: BSCS
---------------------------
Name: rico
Student ID: 23121
Age: 20
Program: BSIT
---------------------------
Do you want to perform another operation?('yes' / 'no'):
```

INVALID INPUTS

```
Do you want to perform another operation?('yes' / 'no'):
asadas
Invalid input.Do you want to perform another operation?('yes' / 'no'):
sadwe
Invalid input.Do you want to perform another operation?('yes' / 'no'):
42
Invalid input.Do you want to perform another operation?('yes' / 'no'):
```

Names in NUMBER REQUIRED areas

```
Enter student ID (ex: 00102):
Rich
Invalid Student ID. It must be exactly 5 digits (e.g., 01234).
Enter student ID (ex: 00102):
23102
Enter student name:
Rich
```

EMPTY INPUTS

```
Please insert student ID to edit:

Invalid Student ID. It must be exactly 5 digits (e.g., 01234).
Please insert student ID to edit:

Invalid Student ID. It must be exactly 5 digits (e.g., 01234).
Please insert student ID to edit:
```

WRONG ID's

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
3
Please insert student ID to edit:
20232
Student with the given ID was not found.
Do you want to perform another operation?('yes' / 'no'):
```

```
--------         Student Management System      --------
1. Add Student Information
2. Display All Student Records
3. Edit Student
4. Display a Student
5. Remove Student
6. Exit
Please select an operation (1 - 6):
4
Please insert student ID to display:
21212
Student with the given ID was not found.
Do you want to perform another operation?('yes' / 'no'):
```

NAMES WITH NUMBERS

```
Enter student ID (ex: 00102):
23032
Enter student name:
johnBOXCZX123
Invalid input. Name cannot contain numbers.
Enter student name:
```

REFLECTION

While working on my Student Management System, I started by creating a Student class to store details like name, student ID, age, and program. I then made a separate class called ManageStudent where I placed all the methods for adding, editing, deleting, and displaying student records. This helped me keep everything organized and easier to work with.

I decided to use a List<Student> collection instead of arrays, stacks, or queues because using a list made it easier to add, remove, and update student records. It was more flexible and simple to manage compared to other data structures.

I also created an InputHelper class to handle input validation and error handling. I used try-catch blocks so the program wouldn't crash when the user entered something wrong, like a letter instead of a number. I added validations to make sure the user doesn't leave fields empty, that the age is between 18 to 40, and that the student ID is exactly five digits. I also made sure it checks for duplicate student IDs.

I used if and while statements to control the flow of the program and make sure only valid inputs go through. I also added user-friendly messages so the user knows if something went wrong or if the operation was successful. Additionally, I used switch case for calling the methods based on the operation they want to perform. Honestly, this project was pretty time-consuming and kind of overwhelming at times, so I had to take breaks just to reset and think clearly. But in the end, I learned a lot about organizing my code, using object-oriented concepts, and handling input properly in C#.