

**Báo cáo NET102**

**Dotnet C# 2**

**LAB 1**

Mã số sinh viên : PS28709

Họ tên sinh viên : Quan Bích Vân

Lớp : IT18322

Giảng viên hướng dẫn : Trần Thế Bảy

MỤC LỤC

Contents

[Tạo Bảng menu Switch case và UI 3](#_Toc135847852)

[Phần Program.cs 3](#_Toc135847853)

[Phần Context.cs 4](#_Toc135847854)

[Câu 1: 6](#_Toc135847855)

[Bài 1 (2 điểm) Sử dụng static class khai báo lớp “MyCollege” gồm 2 static field là “CollegeName” và “Address”. Khởi tạo thông tin cho các field thông qua static contructor và xuất các thông tin lên màn hình console. 6](#_Toc135847856)

[Câu 2: 7](#_Toc135847857)

[Bài 2 (2 điểm) Xây dựng class Student gồm các thông tin StudentName, Course, CollegeName, CollegeAddress. Bổ sung các phần còn thiều vào đoạn code bên dưới để hoàn thiện ứng dụng: 7](#_Toc135847858)

[Câu 3: 10](#_Toc135847859)

[Viết chương trình sử dụng partial class cho các phép tính cộng trừ nhân chia. Bằng 2 cách xây dựng class partial 10](#_Toc135847860)

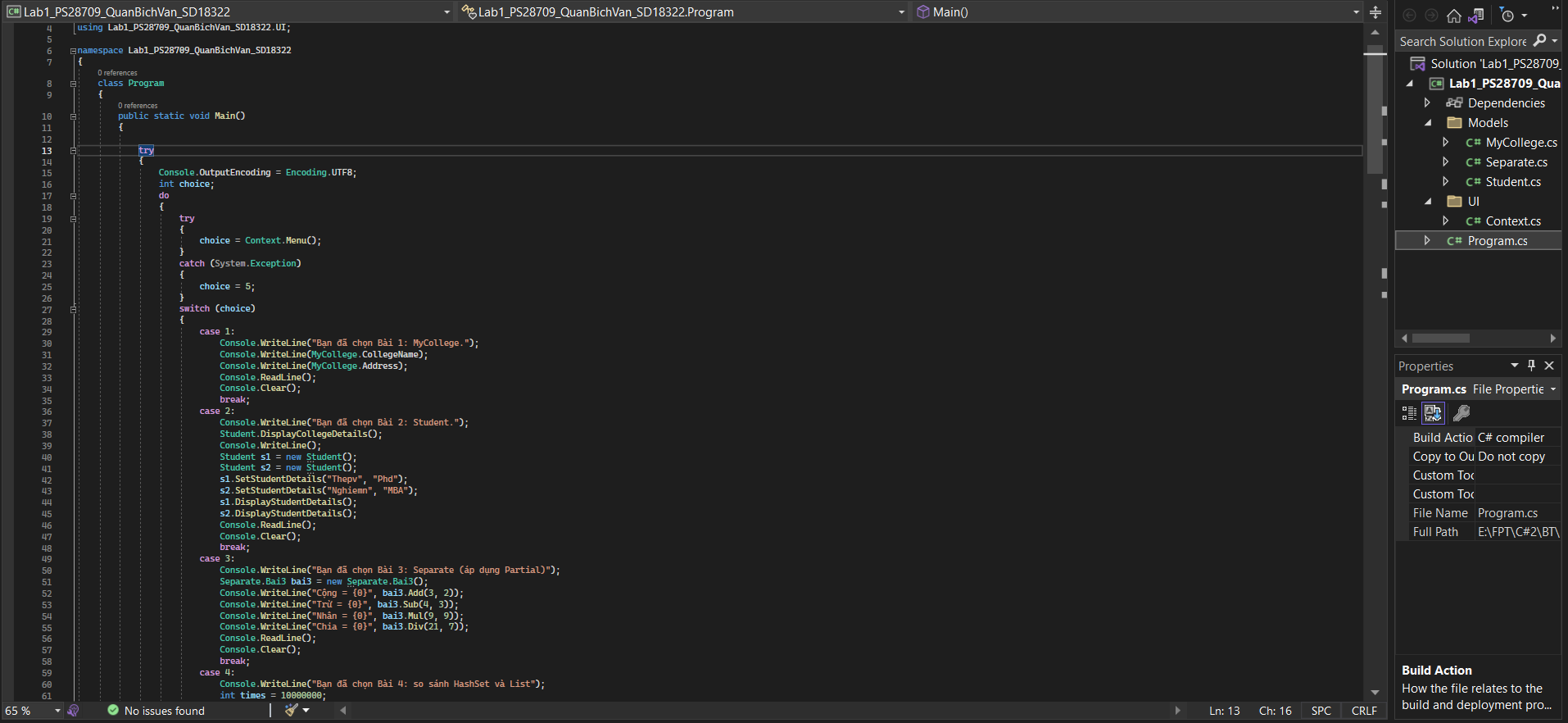
[Câu 4: 13](#_Toc135847861)

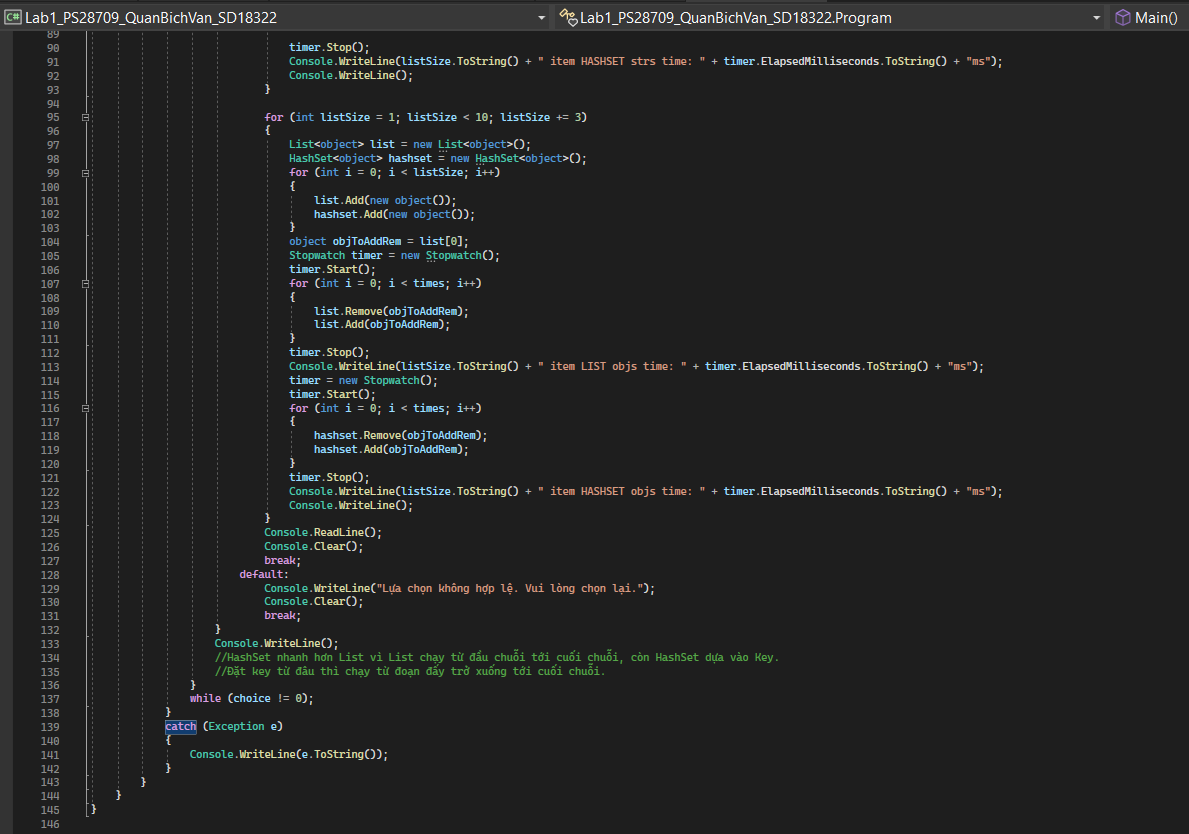
[Viết chương trình so sánh hiệu suất giữa List và HashSet 13](#_Toc135847862)

# Tạo Bảng menu Switch case và UI

## Phần Program.cs

- Bước 1: Khai báo biến Choice với kiểu dữ liệu int. Dùng try catch để hạn chế crash bắt lỗi sinh ra, dùng switch case để xuất lần lượt các chức năng từ 1 🡪 5, cuối cùng là thêm phần exit program và nhập lựa chọn.





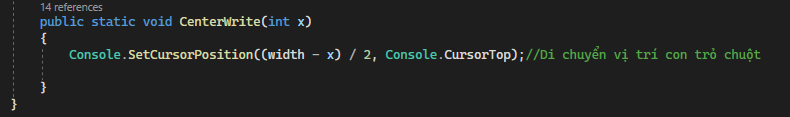
- Bước 2: Dùng switch case gọi và xuất ra từng chức năng.

# Phần Context.cs

**Giải thích về hàm context.CenterWrite(17)**

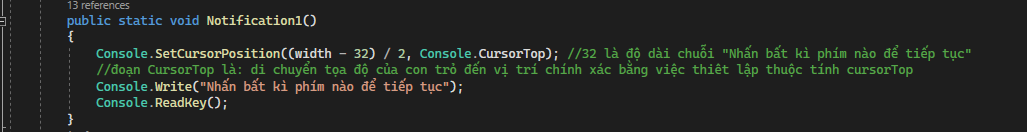
Đây là một phương thức tĩnh CenterWrite được định nghĩa trong lớp Context, có tham số là một số nguyên x. Mục đích của phương thức là di chuyển con trỏ console đến trung tâm của cửa sổ console, dựa trên chiều rộng của cửa sổ và giá trị của x.

Mục đích khi dùng là canh giữa văn bản sau nó. Giá trị 17 được truyền vào như tham số x, có thể tương ứng với độ dài của văn bản sẽ được hiển thị.

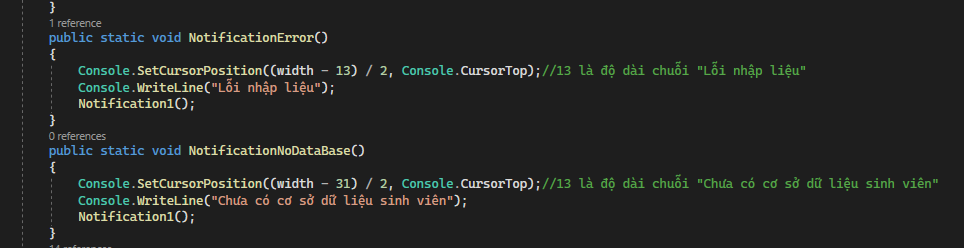


**Giải thích về context. Notification1( )**

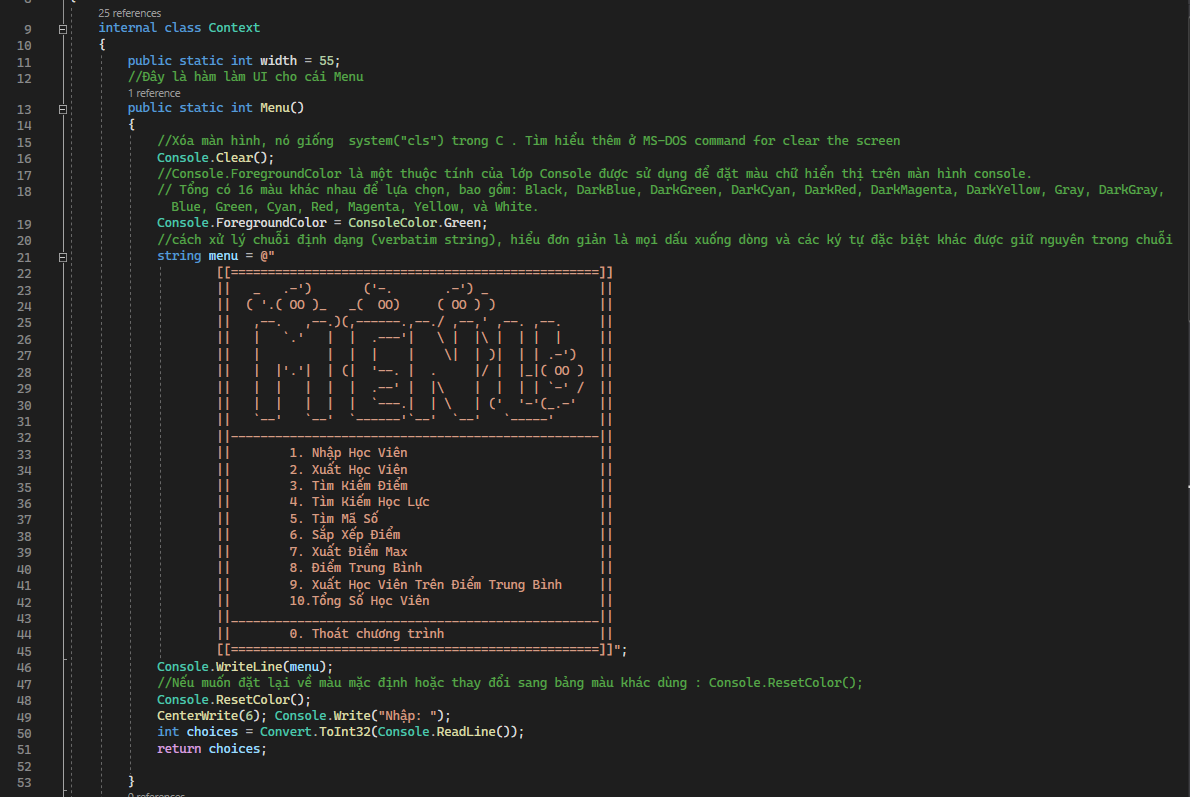
Mục đích của phương thức là hiển thị một thông báo trên console yêu cầu người dùng ấn bất kì phím nào để tiếp tục. Ở đây có thêm hàm Console.SetCursorPosition để di chuyển con trỏ console đến vị trí trung tâm của cửa sổ console



(tương tự như vậy với 2 hàm NotificationError và NotificationNoDataBase



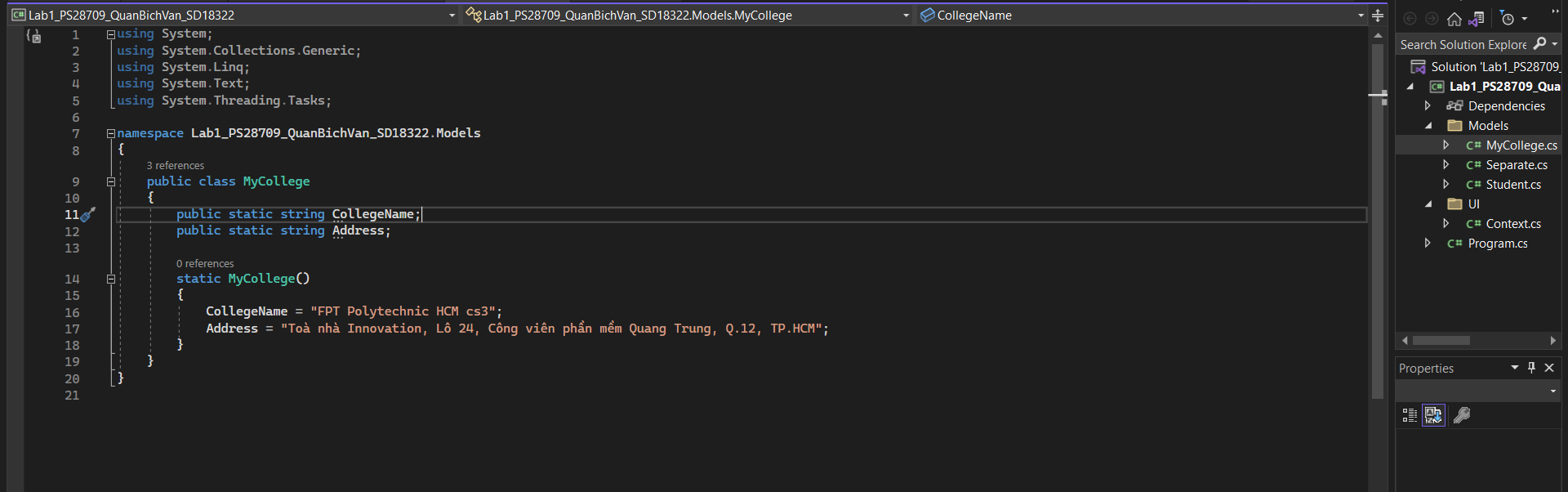
Tạo hàm context để làm UI cho cái menu

# Câu 1:

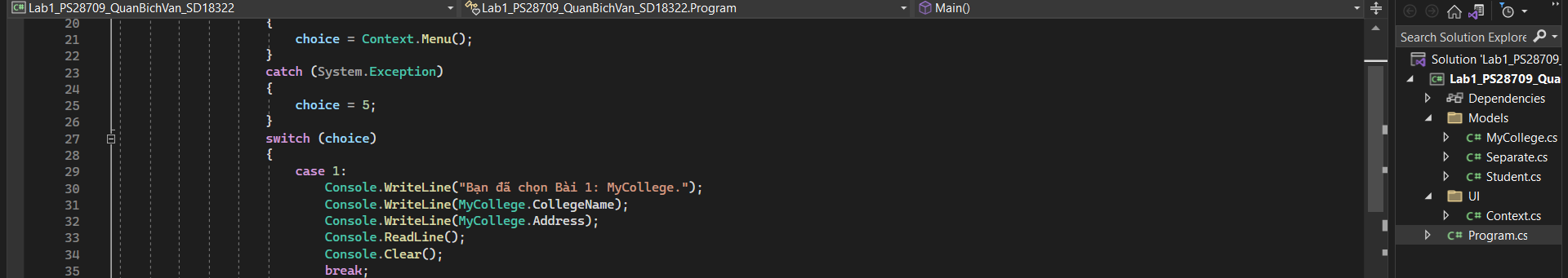
## Bài 1 (2 điểm) Sử dụng static class khai báo lớp “MyCollege” gồm 2 static field là “CollegeName” và “Address”. Khởi tạo thông tin cho các field thông qua static contructor và xuất các thông tin lên màn hình console.

Code:



khai báo một lớp có tên là "**MyCollege**" trong namespace trên. Lớp này có thuộc tính tĩnh (static) là "CollegeName" và "Address" kiểu dữ liệu là chuỗi (string).

Dòng code hàng 14 static MyCollege() bắt đầu một khối mã (block) constructor tĩnh (static constructor) cho lớp "MyCollege". Constructor tĩnh này được gọi khi lớp được khởi tạo và chỉ chạy một lần duy nhất. Trong constructor tĩnh này, giá trị của thuộc tính "CollegeName" được gán là "FPT Polytechnic HCM cs3" và giá trị của thuộc tính "Address" được gán là địa chỉ của FPT Polytechnic HCM cs3.



Trong hàm main (method "Main") của lớp "Program", có một điều kiện (case 1) được kiểm tra. Nếu điều kiện này đúng, những dòng code trong khối lệnh điều kiện này sẽ được thực thi.

Dòng code hàng thứ 30: Console.WriteLine("Bạn đã chọn Bài 1: MyCollege."); in ra màn hình dòng thông báo "Bạn đã chọn Bài 1: MyCollege.".

Tương tự dòng code Console.WriteLine(MyCollege.CollegeName);

Console.WriteLine(MyCollege.Address);

in ra màn hình giá trị của thuộc tính "CollegeName" của lớp "MyCollege". Và in ra màn hình giá trị của thuộc tính "Address" của lớp "MyCollege".

Dòng code Console.ReadLine();chờ người dùng nhấn phím để tiếp tục (đọc một dòng nhập từ bàn phím).

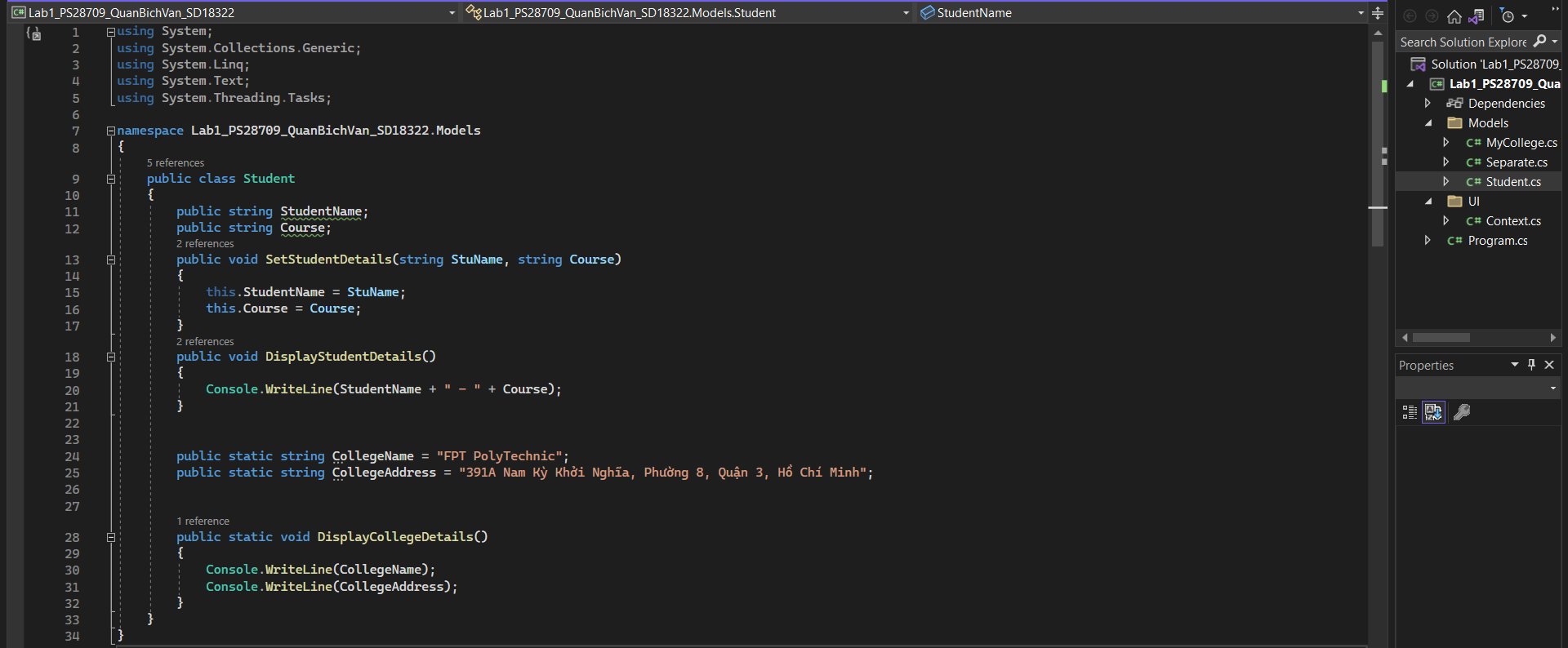
Dòng code Console.Clear(); xóa màn hình console (clear console).

Dòng code break; kết thúc khối lệnh điều kiện (break).

# Câu 2:

## Bài 2 (2 điểm) Xây dựng class Student gồm các thông tin StudentName, Course, CollegeName, CollegeAddress. Bổ sung các phần còn thiều vào đoạn code bên dưới để hoàn thiện ứng dụng:

Code:



khai báo một lớp có tên là "Student". Lớp này có ba thuộc tính công khai (public) là "StudentName" và "Course" kiểu dữ liệu là chuỗi (string), và hai phương thức công khai (public) là "SetStudentDetails" và "DisplayStudentDetails".

public string StudentName;

public string Course;

Phương thức "SetStudentDetails" nhận vào hai tham số là "StuName" và "Course", và gán giá trị của tham số vào thuộc tính tương ứng trong lớp.

public void SetStudentDetails(string StuName, string Course)

{

this.StudentName = StuName;

this.Course = Course;

}

Phương thức "DisplayStudentDetails" in ra màn hình thông tin về tên sinh viên và khóa học.

public void DisplayStudentDetails()

{

Console.WriteLine(StudentName + " - " + Course);

}

Dòng code thứ tám và thứ chín khai báo hai thuộc tính tĩnh (static) là "CollegeName" và "CollegeAddress" kiểu dữ liệu là chuỗi (string), và gán giá trị cho chúng là tên và địa chỉ của trường đại học "FPT PolyTechnic".

( public static string CollegeName = "FPT PolyTechnic";

public static string CollegeAddress = "391A Nam Kỳ Khởi Nghĩa, Phường 8, Quận 3, Hồ Chí Minh"; )

Phương thức "DisplayCollegeDetails" in ra màn hình tên và địa chỉ của trường đại học.

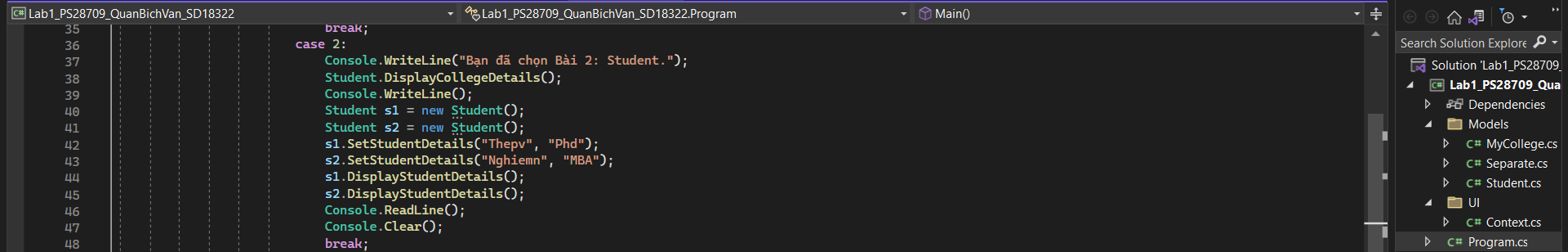
public static void DisplayCollegeDetails()

{

Console.WriteLine(CollegeName);

Console.WriteLine(CollegeAddress);

}



Trong hàm main (method "Main") của lớp "Program", có một điều kiện (case 2) được kiểm tra. Nếu điều kiện này đúng, những dòng code trong khối lệnh điều kiện này sẽ được thực thi.

Dòng code Console.WriteLine("Bạn đã chọn Bài 2: Student.");in ra màn hình dòng thông báo "Bạn đã chọn Bài 2: Student.".

Dòng code Student.DisplayCollegeDetails();gọi phương thức tĩnh "DisplayCollegeDetails" của lớp "Student" để hiển thị thông tin về trường đại học.

Dòng code thứ mười bốn và thứ mười lăm khởi tạo hai đối tượng "s1" và "s2" thuộc lớp "Student".

Dòng code s1.SetStudentDetails("Thepv", "Phd");

s2.SetStudentDetails("Nghiemn", "MBA");

gọi phương thức "SetStudentDetails" của đối tượng "s1" và "s2" để thiết lập thông tin sinh viên.

Dòng code s1.DisplayStudentDetails();

s2.DisplayStudentDetails();

gọi phương thức "DisplayStudentDetails" của đối tượng "s1" và "s2" để hiển thị thông tin sinh viên.

Dòng code Console.ReadLine(); chờ người dùng nhấn phím để tiếp tục (đọc một dòng nhập từ bàn phím).

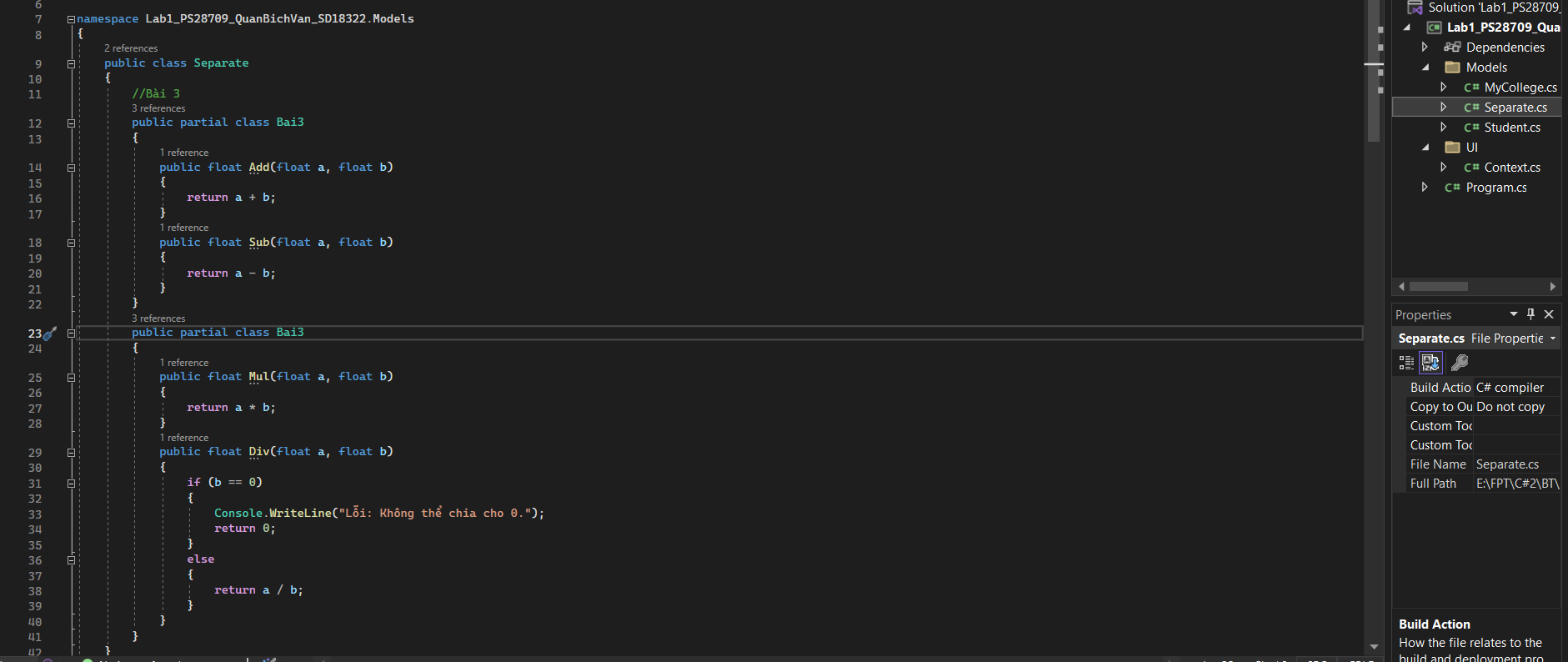
Dòng code Console.Clear(); xóa màn hình console (clear console).

Dòng code break; kết thúc khối lệnh điều kiện (break).

# Câu 3:

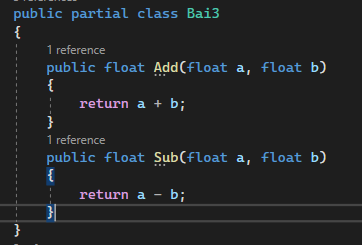
## Viết chương trình sử dụng partial class cho các phép tính cộng trừ nhân chia. Bằng 2 cách xây dựng class partial

Code:

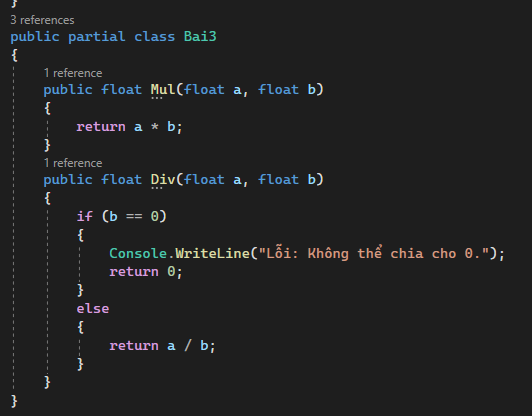


Dòng code đầu tiên khai báo một lớp có tên là "Separate". Lớp này chứa một khối mã (block) được gọi là "Bai3" và được khai báo là "partial".

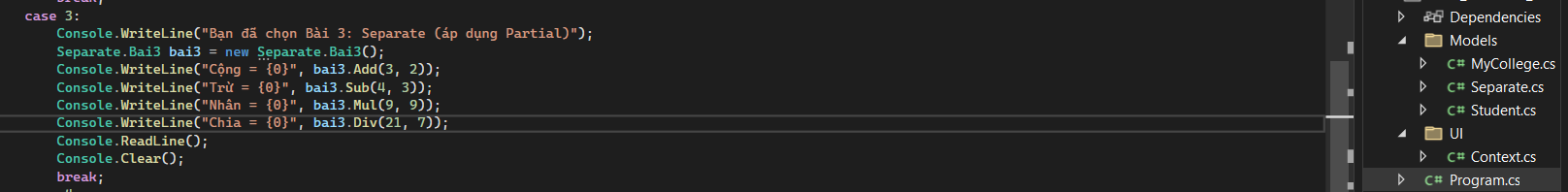
Trong khối mã "Bai3", có hai phương thức công khai (public) là "Add" và "Sub". Phương thức "Add" nhận vào hai tham số a và b, và trả về tổng của hai số đó. Phương thức "Sub" nhận vào hai tham số a và b, và trả về hiệu của hai số đó.



Sau đó, khối mã "Bai3" được khai báo lại một lần nữa để mở rộng thêm hai phương thức nữa là "Mul" và "Div". Phương thức "Mul" nhận vào hai tham số a và b, và trả về tích của hai số đó. Phương thức "Div" nhận vào hai tham số a và b, và kiểm tra nếu b là 0 thì in ra màn hình thông báo lỗi "Lỗi: Không thể chia cho 0." và trả về 0. Nếu b khác 0, phương thức trả về thương của hai số đó.



Trong hàm main:



Trong hàm main (method "Main") của lớp "Program", có một điều kiện (case 3) được kiểm tra. Nếu điều kiện này đúng, những dòng code trong khối lệnh điều kiện này sẽ được thực thi.

Dòng code Console.WriteLine("Bạn đã chọn Bài 3: Separate (áp dụng Partial)"); in ra màn hình dòng thông báo "Bạn đã chọn Bài 3: Separate (áp dụng Partial)".

Dòng code Separate.Bai3 bai3 = new Separate.Bai3();

khởi tạo một đối tượng "bai3" thuộc lớp "Separate.Bai3".

Dòng code Console.WriteLine("Cộng = {0}", bai3.Add(3, 2));

in ra màn hình giá trị của phép cộng sử dụng phương thức "Add" của đối tượng "bai3".

Dòng code Console.WriteLine("Trừ = {0}", bai3.Sub(4, 3));

in ra màn hình giá trị của phép trừ sử dụng phương thức "Sub" của đối tượng "bai3".

Dòng code Console.WriteLine("Nhân = {0}", bai3.Mul(9, 9));

in ra màn hình giá trị của phép nhân sử dụng phương thức "Mul" của đối tượng "bai3".

Dòng code Console.WriteLine("Chia = {0}", bai3.Div(21, 7));

in ra màn hình giá trị của phép chia sử dụng phương thức "Div" của đối tượng "bai3". Nếu phép chia có lỗi (chia cho 0), sẽ in ra thông báo lỗi.

Dòng code Console.ReadLine(); chờ người dùng nhấn phím để tiếp tục (đọc một dòng nhập từ bàn phím).

Dòng code Console.Clear(); xóa màn hình console (clear console).

Dòng code break; kết thúc khối lệnh điều kiện (break).

# Câu 4:

## Viết chương trình so sánh hiệu suất giữa List và HashSet

Dòng code đầu tiên in ra màn hình dòng thông báo "Bạn đã chọn Bài 4: so sánh HashSet và List".

Console.WriteLine("Bạn đã chọn Bài 4: so sánh HashSet và List");

Dòng code thứ ba khai báo biến "times" với giá trị là 10000000, đại diện cho số lần lặp trong vòng for.

int times = 10000000;

Khởi tạo một vòng lặp for để thử nghiệm và so sánh hiệu suất giữa List và HashSet với kích thước tăng dần từ 1 đến 9.

Trong khối lệnh của vòng lặp for này, tạo ra một List<string> có tên là "list" và một HashSet<string> có tên là "hashset". Tiếp theo, trong một vòng lặp khác, thêm các chuỗi "string" + số thứ tự vào cả List và HashSet.

Sau đó, tạo một đối tượng Stopwatch để đo thời gian thực thi. Bắt đầu đếm thời gian bằng cách gọi phương thức Start() của Stopwatch.

Tiếp theo, trong một vòng lặp, thực hiện việc xóa và thêm một phần tử trong List "list" với chuỗi "string0", lặp lại theo số lần đã khai báo trong biến "times".

for (int listSize = 1; listSize < 10; listSize++)

{

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

HashSet<string> hashset = new HashSet<string>();

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

{

list.Add("string" + i.ToString());

hashset.Add("string" + i.ToString());

}

Stopwatch timer = new Stopwatch();

timer.Start();

Khi vòng lặp kết thúc, gọi phương thức Stop() của Stopwatch để dừng đếm thời gian. In ra màn hình thời gian thực thi của List với kích thước listSize bằng cách sử dụng phương thức ElapsedMilliseconds của Stopwatch.

Sau đó, tạo một Stopwatch mới và lặp lại quá trình tương tự như trên, nhưng thay vì sử dụng List, sử dụng HashSet.

Sau khi in ra thời gian thực thi của cả List và HashSet, in ra một dòng trống để tách biệt giữa các lần thử nghiệm khác nhau.

Sau đó, có một vòng lặp khác để thử nghiệm và so sánh hiệu suất giữa List và HashSet với các đối tượng tùy chỉnh (object) thay vì chuỗi.

Trong khối lệnh của vòng lặp này, tạo ra một List<object> có tên là "list" và một HashSet<object> có tên là "hashset". Tiếp theo, trong một vòng lặp khác, thêm các đối tượng tùy chỉnh vào cả List và HashSet.

Sau đó, tạo một đối tượng để thêm và xóa khỏi List và HashSet, lấy đối tượng đầu tiên trong List và gán cho "objToAddRem".

Tạo một Stopwatch mới và bắt đầu đếm thời gian.

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

{

list.Remove("string0");

list.Add("string0");

}

timer.Stop();

Console.WriteLine(listSize.ToString() + " item LIST strs time: " + timer.ElapsedMilliseconds.ToString() + "ms");

timer = new Stopwatch();

timer.Start();

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

{

hashset.Remove("string0");

hashset.Add("string0");

}

timer.Stop();

Console.WriteLine(listSize.ToString() + " item HASHSET strs time: " + timer.ElapsedMilliseconds.ToString() + "ms");

Console.WriteLine();

}

Trong một vòng lặp, thực hiện việc xóa và thêm đối tượng trong List "list" và HashSet "hashset" với đối tượng "objToAddRem", lặp lại theo số lần đã khai báo trong biến "times".

Khi vòng lặp kết thúc, gọi phương thức Stop() của Stopwatch để dừng đếm thời gian. In ra màn hình thời gian thực thi của List với kích thước listSize bằng cách sử dụng phương thức ElapsedMilliseconds của Stopwatch.

Sau đó, tạo một Stopwatch mới và lặp lại quá trình tương tự như trên, nhưng thay vì sử dụng List, sử dụng HashSet.

Sau khi in ra thời gian thực thi của cả List và HashSet, in ra một dòng trống để tách biệt giữa các lần thử nghiệm khác nhau.

Cuối cùng, chờ người dùng nhấn phím để tiếp tục (đọc một dòng nhập từ bàn phím).

Sau khi người dùng nhấn phím, xóa màn hình console (clear console).

Quá trình thực hiện lặp lại cho đến khi kết thúc vòng lặp switch case trong hàm main.

for (int listSize = 1; listSize < 10; listSize += 3)

{

List<object> list = new List<object>();

HashSet<object> hashset = new HashSet<object>();

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

{

list.Add(new object());

hashset.Add(new object());

}

object objToAddRem = list[0];

Stopwatch timer = new Stopwatch();

timer.Start();

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

{

list.Remove(objToAddRem);

list.Add(objToAddRem);

}

timer.Stop();

Console.WriteLine(listSize.ToString() + " item LIST objs time: " + timer.ElapsedMilliseconds.ToString() + "ms");

timer = new Stopwatch();

timer.Start();

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

{

hashset.Remove(objToAddRem);

hashset.Add(objToAddRem);

}

timer.Stop();

Console.WriteLine(listSize.ToString() + " item HASHSET objs time: " + timer.ElapsedMilliseconds.ToString() + "ms");

Console.WriteLine();

}

Console.ReadLine();

Console.Clear();

break;