DOCTOR C# oops concepts:

-------------------------------------------------------------------------

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OopsApplication

{

internal class Doctor

{

/// <summary>

/// Initial constructor consists of default values when empty constructor is called

/// </summary>

public Doctor() {

Id = 0;

Name = string.Empty;

Age = 0;

Experience = 0;

Qualification = string.Empty;

Speciality = string.Empty;

}

/// <summary>

/// Constructor called when Id alone is defined by the user

/// </summary>

/// <param name="id">Id of Doctor</param>

public Doctor(int id){Id = id;}

/// <summary>

/// This Constructor is called when all the details about the Doctors were passed through parameters

/// </summary>

/// <param name="id"></param>

/// <param name="name"></param>

/// <param name="experience"></param>

/// <param name="age"></param>

/// <param name="qualification"></param>

/// <param name="speciality"></param>

public Doctor(int id, string name,double experience,int age,string qualification, string speciality):this(id) {

Name = name;

Experience = experience;

Age = age;

Qualification = qualification;

Speciality = speciality;

}

/// <summary>

/// Here The Getters and Setters were Implemented for secured data

/// </summary>

public int Id { get; private set; }

public string Name { get; set; }

public double Experience { get; set; }

public double Age { get; set; }

public string Qualification { get; set; }

public string Speciality { get; set; }

/// <summary>

/// Function to print the Details of Every Doctors Registered so far.

/// </summary>

public void PrintDoctorDetails()

{

Console.WriteLine($"Doctor Id\t \t \t \t:\t{this.Id}");

Console.WriteLine($"Doctor name\t \t \t \t:\t{this.Name}");

Console.WriteLine($"Doctor Qualification\t \t \t:\t{this.Qualification}");

Console.WriteLine($"Doctor Age \t \t \t \t :\t{this.Age}");

Console.WriteLine($"Doctor Experience \t \t \t :\t{this.Experience}");

Console.WriteLine($"Doctor Speciality\t \t \t:\t{this.Speciality}");

Console.WriteLine();

Console.WriteLine();

}

}

}

namespace OopsApplication

{

internal class Program

{

static void specialityFind(Doctor[] doctors,String speciality)

{

for(int i = 0; i < doctors.Length; i++)

{

if (doctors[i].Speciality == speciality)

{

doctors[i].PrintDoctorDetails();

}

}

}

Doctor CreateNewDoctorUsingConsoleData(int Id)

{

Doctor doctor = new Doctor(Id);

Console.WriteLine($"----------------Enter New Doctor Details----------------");

Console.WriteLine($"Id : {Id}");

Console.WriteLine("----------");

Console.WriteLine("Enter Your Name : ");

doctor.Name = Console.ReadLine();

Console.WriteLine("Enter your Experience in Years : ");

int experience;

while (!int.TryParse(Console.ReadLine(), out experience))

{

Console.WriteLine("Invalid Data, Please provide proper Experience in year : ");

}

doctor.Experience = experience;

Console.WriteLine("Enter your Age in Years : ");

int age;

while (!int.TryParse(Console.ReadLine(), out age))

{

Console.WriteLine("Invalid Data, Please provide proper Experience in year : ");

}

doctor.Age = age;

Console.WriteLine("Enter your Qualification : ");

doctor.Qualification = Console.ReadLine();

Console.WriteLine("Enter your Specialization : ");

doctor.Speciality = Console.ReadLine();

Console.WriteLine("---------------------------------------------------------");

Console.WriteLine();

Console.WriteLine();

return doctor;

}

static void Main(string[] args)

{

Program program = new Program();

Doctor[] doctors = new Doctor[3];

Doctor doctor1 = new Doctor

{

Name = "kavin",

Experience = 2,

Age = 20,

Qualification = "MBBS.,M.phil.",

Speciality = "Cardio"

};

for (int ind = 0; ind < doctors.Length; ind++)

{

doctors[ind] = program.CreateNewDoctorUsingConsoleData(ind + 1);

}

for(int ind = 0;ind < doctors.Length; ind++)

{

doctors[ind].PrintDoctorDetails();

}

Console.WriteLine("Enter the Speciality Type to Search : ");

string speciality = Console.ReadLine();

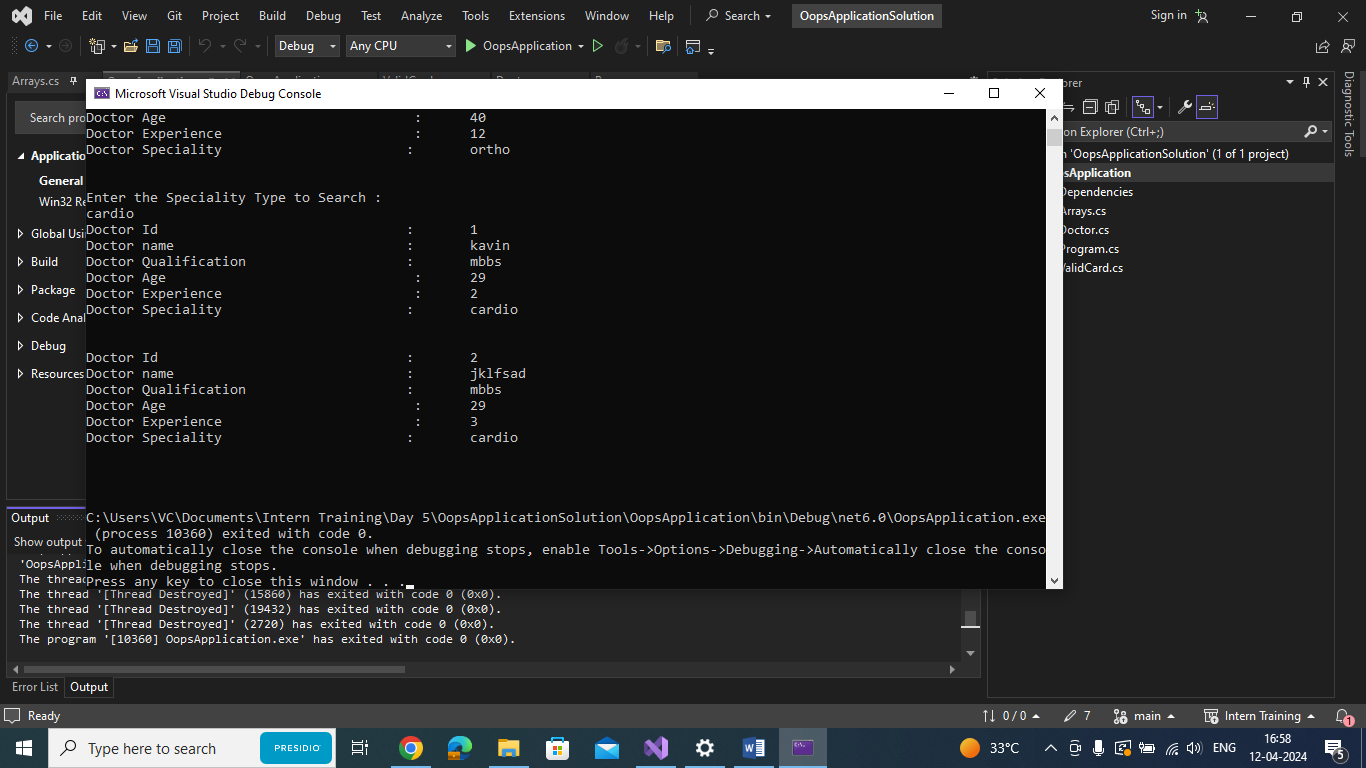
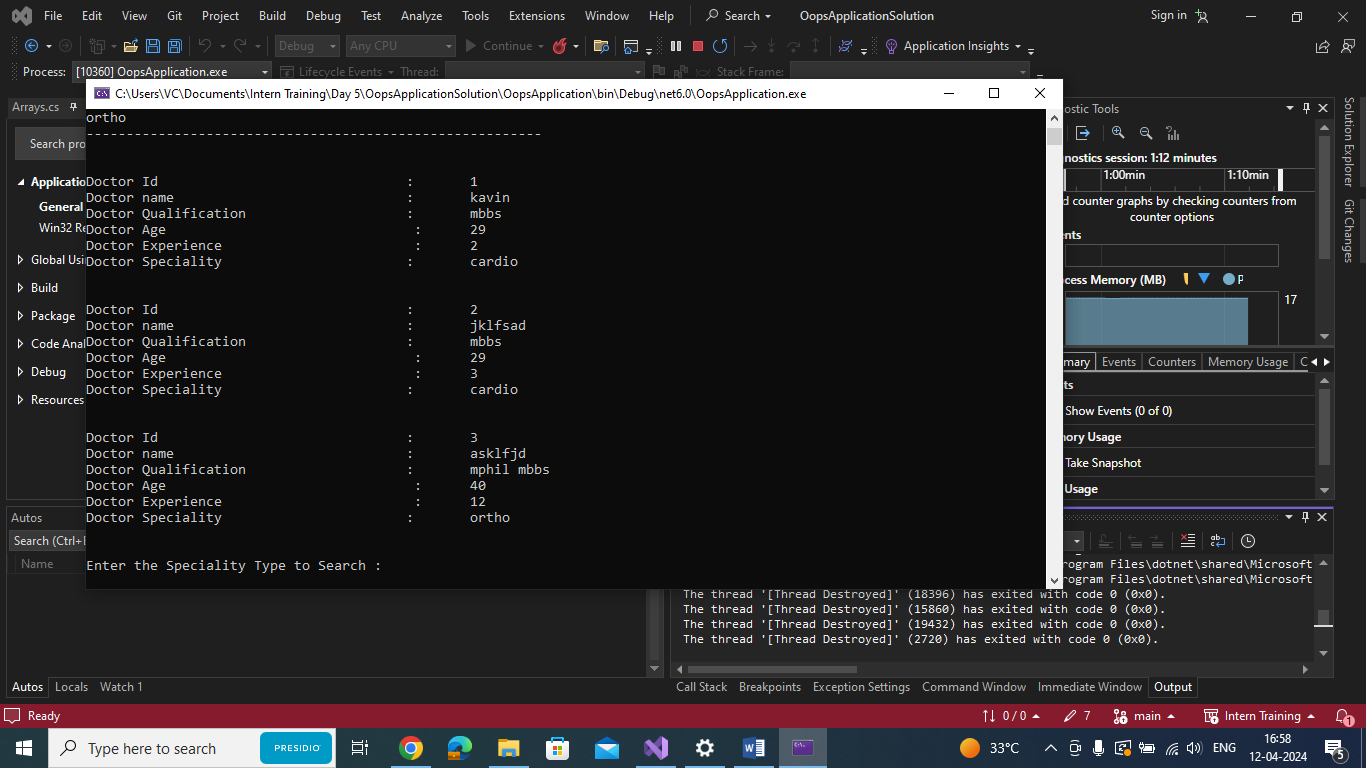
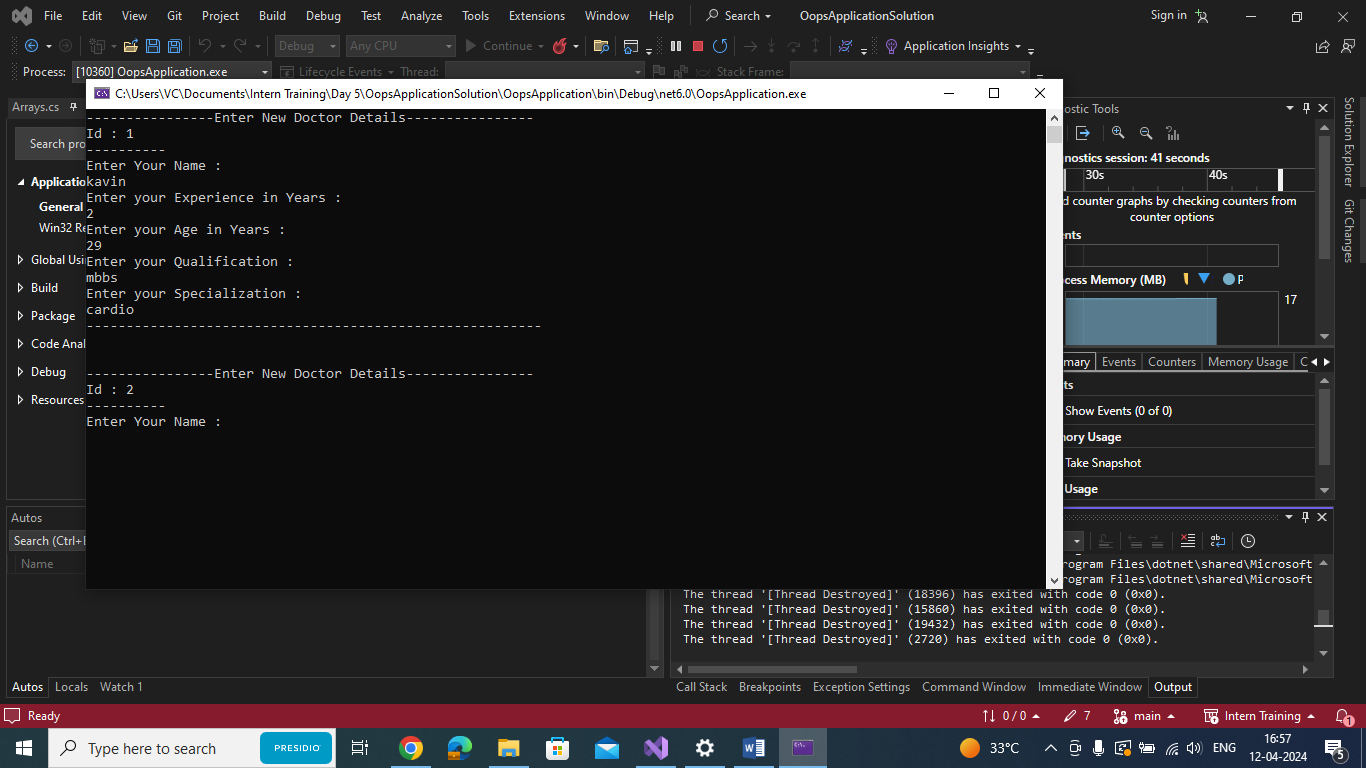
specialityFind(doctors, speciality);

Console.WriteLine();

}

}

}



VALID CARD NUMBER

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace OopsApplication

{

internal class ValidCard

{

/// <summary>

/// Reversing the given card number and converting it to an reversed integer array for further arithmatic operation

/// </summary>

/// <param name="str"></param>

/// <returns></returns>

static int[] reverseNumberToIntegerArray(string str)

{

char[] charArray = str.ToCharArray();

Array.Reverse(charArray);

int[] numbers = new int[charArray.Length];

for(int i = 0; i < charArray.Length; i++)

{

numbers[i] = charArray[i]-'0';

}

return numbers;

}

/// <summary>

/// Arithmetic operations to be handled to complete the

/// </summary>

/// <param name="str"></param>

/// <returns></returns>

static bool verificationOperations(string str)

{

if(str.Length < 15) { return false; };

int[] numbers = reverseNumberToIntegerArray(str);

int total = 0;

for(int ind = 0; ind < numbers.Length; ind++)

{

if((ind+1)%2 == 0)

{

numbers[ind] \*= 2;

}

if (numbers[ind] > 9)

{

int once = numbers[ind] % 10;

int tens = numbers[ind] / 10;

numbers[ind] = once + tens;

}

total += numbers[ind];

}

Console.WriteLine("The Final Value is " + total);

if (total % 10 == 0) return true;

else return false;

}

static void Main(string[] args)

{

Console.WriteLine("Enter your Code");

string cardNumber = Console.ReadLine();

if (verificationOperations(cardNumber)) Console.WriteLine("Valid Card Number ... !");

else Console.WriteLine("Invalid Card Number ... !");

}

}

}

