1) Create a program that will take a 16 digit number from user

the card number

4477 4683 4311 3002

Reverse the number

2003 1134 3864 7744

Even position number multiply by 2

2+0+0+6+1+2+3+8+3+16+6+8+7+14+4+8

sum up the 2 digit numbers

2+0+0+6+1+2+3+8+3+7+6+8+7+5+4+8

70 % 10 = 0

public bool AuthenticateCardNumber(long digits)

{

int[] reversed = new int[16];

string strNum = digits.ToString();

int revCount = 0;

for (int i = 15; i >= 0; i--)

{

//reversed

int c = Int32.Parse(strNum[i].ToString());

reversed[revCount++] = c;

}

for (int i = 1; i <= 16; i++)

{

if (i % 2 == 0)

{

reversed[i-1] = reversed[i-1] \* 2;

}

}

int count = 0, sum=0, m;

for (int i = 0; i < 16; i++)

{

count = getDigits(reversed[i], count);

if (count > 1)

{

while (reversed[i] > 0)

{

m = reversed[i] % 10;

sum = sum + m;

reversed[i] = reversed[i] / 10;

}

reversed[i] = sum;

}

count = 0;

sum = 0;

}

//sum of all numbers

for (int i = 0; i < 16; i++)

{

sum += reversed[i];

}

if (sum % 10 == 0)

return true;

else

return false;

}

public int getDigits(int n1, int nodigits)

{

if (n1 == 0)

return nodigits;

return getDigits(n1 / 10, ++nodigits);

}

2) Take 11 numbers from user and find that one number which is not repeating

example

2,3,4,5,1,10,3,2,5,4,1

10

public void FindNonRepeatingNumber()

{

Console.WriteLine("Please key in 11 numbers, one by one separated by comma");

int[] arr = new int[11];

int i = 0;

string strNumbers = Console.ReadLine();

foreach (var sn in strNumbers.Split(','))

{

int n = Convert.ToInt32(sn);

// work with n

arr[i++] = n;

}

int ii, j;

for (ii = 0; ii < 11; ii++)

{

for (j = 0; j < 11; j++)

{

if (arr[ii] == arr[j] && ii != j)

//repeated

break;

}

if (j == 11)

{

Console.WriteLine("non repeating element is: "+ arr[ii]);

}

}

}



4) Take number from user until the user inserts a negative number.

Sort and print all the values

Find the median and mode(If no repeation then no mode)

public void FindMedianAndMode()

{

bool isPositive = true;

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

int no = 0;

while (isPositive)

{

Console.WriteLine("Please enter a neg no.");

no = Convert.ToInt32(Console.ReadLine());

if (no > 0)

list.Add(no);

else

isPositive = false;

}

list.Sort();

//median

int size = list.Count;

if (size % 2 != 0)

Console.WriteLine("Median: " + (double)list[size / 2]);

else

{

double xx = ((double)list[size / 2 - 1] + (double)list[size / 2]) / 2;

Console.WriteLine("Median: " + xx);

}

//mode

int[] count = new int[size + 1];

for (int i = 0; i < size + 1; i++)

count[i] = 0;

for (int i = 0; i < list.Count; i++)

count[list[i]]++;

// mode is the index with maximum count

int mode = 0;

int k = count[0];

for (int i = 1; i < size + 1; i++)

{

if (count[i] > k)

{

k = count[i];

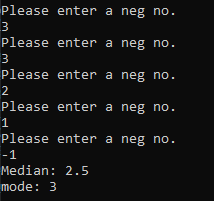
mode = i;

}

}

Console.WriteLine("mode: "+mode);

}



5)https://leetcode.com/explore/featured/card/fun-with-arrays/521/introduction/3237/

6) https://leetcode.com/explore/featured/card/fun-with-arrays/526/deleting-items-from-an-array/3248/

7) Update your project to add, edit, print appointments

public void BookAppointment(User u, User[] users)

{

//for doctor and patient

Appointment appointment = new Appointment();

Console.WriteLine("You have chosen book appointment.");

if (u.Type == "Doctor")

{

appointment.DoctorId = u.Id;

appointment.PatientId = FindPatientIDByName(users);

}

else

{

appointment.PatientId = u.Id;

//patient cannot choose doctor, to be set later

}

//any time slot for now

Console.WriteLine("Please enter a date and time in format dd-mm-yyyy hh:mm tt (eg. 20-08-22 12:30 AM)");

AppDate = Convert.ToDateTime(Console.ReadLine());

Console.WriteLine("Please enter a reason for the appointment.");

Category = Console.ReadLine();

DateCreated = DateTime.Now;

Status = "New";

}

public void ViewAppointmentDetails(User u, User[] users, Appointment[] appointments)

{

//for doctor and patient

Console.WriteLine("You have chosen to view appointment.");

//get those details n priint out line by line

Appointment app = FindAppoinmentByAppDateAndPatientId(appointments, users);

Console.WriteLine("Your selected appointment details are:");

Console.WriteLine("Doctor name: "+users[app.DoctorId-1].Name);

Console.WriteLine("Patient name: " + users[app.PatientId - 1].Name);

Console.WriteLine("Details: "+ app.Details);

Console.WriteLine("Remarks: "+ app.Remarks);

Console.WriteLine("Status: "+app.Status);

Console.WriteLine("Appointment Date: "+ app.AppDate);

Console.WriteLine("Category: "+ app.Category);

Console.WriteLine("Payment amount: "+ app.PaymentAmount);

Console.WriteLine("Payment Status: "+ app.PaymentStatus);

}

public void UpdateAppointment(User u, Appointment[] appointments, User[] users)

{

//for doctor(mostly actually) and patient

Console.WriteLine("You have chosen to update appointment.");

//find appointment

Appointment app = FindAppoinmentByAppDateAndPatientId(appointments, users);

//ask which field to edit

Console.WriteLine("Please enter which field do you want to edit.");

Console.WriteLine("(1) Details");

Console.WriteLine("(2) Remarks" );

Console.WriteLine("(3) Status");

Console.WriteLine("(4) Appointment Date in format dd-mm-yyyy hh:mm tt (eg. 20-08-22 12:30 AM)");

Console.WriteLine("(5) Category");

Console.WriteLine("(6) Payment amount");

Console.WriteLine("(7) Payment Status");

int choice = Convert.ToInt32(Console.ReadLine());

//edit field into

Console.WriteLine("Please enter what values you want to change to:");

string editTo = Console.ReadLine();

if(app == null)

{

Console.WriteLine("Invalid Id, cannot edit");

return;

}

switch (choice)

{

case 1:

app.Details = editTo ;

break;

case 2:

app.Remarks = editTo;

break;

case 3:

app.Status = editTo;

break;

case 4:

app.AppDate = Convert.ToDateTime(editTo);

break;

case 5:

app.Category = editTo;

break;

case 6:

app.PaymentAmount = Convert.ToDouble(editTo);

break;

case 7:

app.PaymentStatus = editTo;

break;

default:

Console.WriteLine("Invalid Entry.");

break;

}

Console.WriteLine("The appointment has been changed.");

ViewAppointmentDetails(u, users, appointments);

}

public Appointment FindAppoinmentByAppDateAndPatientId(Appointment[] apps, User[] users)

{

DateTime appDate;

int patientId = 0;

//int arrNoApp = 0;

Appointment app = null;

Boolean appFound = false;

while (appFound == false)

{

patientId = users[FindPatientIDByName(users)].Id;

Console.WriteLine("Please enter your appointment date in format dd-mm-yyyy hh:mm tt (eg. 20-08-22 12:30 AM)");

appDate = Convert.ToDateTime(Console.ReadLine());

//

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

{

if(apps[i].PatientId== patientId && apps[i].AppDate == appDate)

{

app = apps[i];

Console.WriteLine("Appointment found!");

appFound = true;

break;

}

}

}

return app;

}