Employees

Program name : *Employees*

Remember: Name Comment Block, Line Comments

Name Zip as: Mod1Test_UserID

Idea:

Complete the following programming problem to the best of your ability without using the text book or searching the internet for code. At this point you should be able to complete 80% of this assignment in the time allotted.

Problem:

Create an application that allows the user to choose from the following:

1. Enter employee's name.
2. Enter employee's phone number, (create you own).
3. Enter employee's age.
4. Display employee's information. (include name, number, and age).
5. Display average age of employees.
6.Exit.

As the user enters the employee's name, it should be saved in an array that has 5 elements. The employee's number should also be saved in an array. The age of the employee should be stored in a List for future access.

All of the user's choices should work.

Hint:

Remember that all input from the user is a string datatype and you can not do any math with strings. Keep in mind that arrays do not grow or shrink.

Useful information:

Loops, Array, List, Decision Structures

You should never jump right to coding when you are presented with a programming problem. The first thing you should do is take time and break the problem down. If you can break it down, then you will be able to code it faster.

To break this problem down we will use I.P.O. This stand for Input, Process, and Output. You should read through the programming problem at less once for each of these. Let's look at what we may get when we read through it for input.

The first thing we can tell is this will be menu driven which means a loop of some sort. With a loop we will need a variable that we can change to end the loop.

Bool exit

We see that the user will be giving us all the information that we need. This means that we will need to store the users input in a variable. We can go ahead and give this variable a name "input". We also see that we are required to have 2 arrays and one List. As we write this down under our Input column, we can give them datatypes and names.

String[] employeeNames
String{} employeePhone
List<int> employeeAge

We are also told that the arrays should have 5 elements. The size could change, and we do not want to hunt down every array and change it so a constant variable would be in order.

Const int SIZE

As of this moment out Input will look like this:

Input

string input
bool exit
const in SIZE
String[] employeeNames
String{} employeePhone
List<int> employeeAge

This can and may change as we move on.

Next we will read through the problem for Output. We do it in this order so that we can see what we need to do in Processing to get from Input to Output.

The first thing we see this time is the menu, so we can list the menu as one of our Outputs. The next thing that will be an Output is the prompts to the user. There will be a total of three of these prompts and we will be using the Console.Write to display these. Next we will need to display the employee's information and then the average age. We now have our Output.

Output:

Menu
(3) Console.Write(user prompt)
Display employee information
    Name
    Phone
    Age
Display employees average age

Finally, we read through again for our process. By reading through the problem a third time and looking at what is put in then what is to come out, we can determine what processes we need. The first thing is to resolve the fact we noticed with the Output. This will be menu driven so it means we will be using a loop. Out of all the loops which would be a better fit here? Well, we want the menu to show at less one time so a do/while loop will do nicely here.

Next we will be getting the users choice from the menu so we will also need a Console.ReadLine(). Once we have the user's input the program needs to decide what to do with this. You can use a structure you want but for this we will use a switch statement. We can also tell from the menu that we will need 6 cases and one default in the switch.

For the first case we will prompt the user then add the input into the employeeNames array. This will be the first place that we notice we need another variable under our Input column. This variable will be used to keep track of the next element in the employeeName array. We will call it nameIndex and we will internalize it at the beginning of the program.

Case two is the same as case one except the variable we need we should call phone Index.

For case three we do not need to keep track of the index since List grow and shrink. Since input comes in as a string from the user, we will need to convert the input into an int. However, we do not want our program to crash if the user enters a letter. We will use a TryParse to get this

done. This also means that we need another variable to hold what comes out of the TryParse. We will name it:

Int number

This variable will be declared and initialized in the case since that is the only place it is needed. If the conversion fails, then we will inform the user. Now our Output will change for a Console.WriteLine.

In case four we will use a loop to display the employee's information.
In case five we display the average age.
In case six we change a variable to exit the loop
The default will display an error message

Now our complete I.P.O. should look like this:

Input

string input
bool exit
int nameIndex
int phoneIndex
const in SIZE
String[] employeeNames
String{} employeePhone
List<int> employeeAge

Process:

Do/While (menu)
Get and store user input
Decision Structure
    Switch
        Case 1
            Get and store employee's name
        Case 2
            Get and store employee's Phone
        Case 3
            Get, convert, and store employee's age
        Case 4
            Display employee's Information
        Case 5
            Display average age
        Case 6
            Exit program
        Default
            Display error

Output:

Menu
(3) Console.Write(user prompt)
Display employee information
    Name
    Phone
    Age
Display employees average age
2 Console.WriteLine(Error)

Now we can turn our attention to making this happen in code.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace ConsoleUI
{
   class Program
   {
      static void Main(string[] args)
      {
         //create input var for user input and sentury for loop
         string input;
         bool exit = false;
         //create constant var
         const int SIZE = 5;

         int nameIndex = 0, phoneIndex = 0;


         string[] employeeName = new string[SIZE];
         string[] employeePhone = new string[SIZE];
         List<int> employeeAge = new List<int>();

         do
         {
            Console.WriteLine("1. Enter employee's name");
            Console.WriteLine("2. Enter employee's phone number");
            Console.WriteLine("3.Enter employee's age");
            Console.WriteLine("4. Display employee information ");
            Console.WriteLine("5. Display average age of employees");
            Console.WriteLine("6. Exit");
            Console.Write("-->");
            input = Console.ReadLine();

            //Switch  to direct to propper process
            switch (input)
            {
               case "1":
                  Console.Write("Enter employee's name -->");
                  input = Console.ReadLine();
                  employeeName[nameIndex] = input;
                  nameIndex++;
```

```
                Console.WriteLine();
                break;
            case "2":
                Console.Write("Enter employee's phone number -->");
                input = Console.ReadLine();
                employeeName[phoneIndex] = input;
                nameIndex++;
                Console.WriteLine();
                break;
            case "3":
                int number = 0
                Console.Write("Enter employee's age -->");
                input = Console.ReadLine();
                if (int.TryParse(input, out number))
                {
                    employeeAge.Add(number);
                }
                else
                {
                    Console.WriteLine("Not a valid number!");
                }
                Console.WriteLine();
                break;
            case "4":
                                                    // We use the Count methode of the list since will
may not have as
                                                    // may element as the arrays
                for (int index = 0; index < employeeAge.Count; index++)
                {
                    Console.WriteLine($"Employee Name- {employeeName[index]}");
                    Console.WriteLine($"Employee Phone- {employeePhone[index]}");
                    Console.WriteLine($"Employee Age- {employeeAge[index]}");


                }
                Console.WriteLine();
                break;
            case "5":
                Console.WriteLine(employeeAge.Average());
                Console.WriteLine();
                break;
            case "6":
                exit = true;
                break;
```

```
                default:
                    Console.WriteLine("Not a valid choice!");
                    Console.WriteLine();
                    break;

            }

        } while (exit == false);
    }
  }
}
```