Logic

Mark Analyser

Workshop 3 (worth 3% of your final grade)
URL: https://github.com/Seneca-144100/IPC-WS3

In this workshop, you are to write a program that accepts several marks for an assessment in a class and analyses the marks and prints the results.

LEARNING OUTCOMES

Upon successful completion of this workshop, you will have demonstrated the abilities to:

- Create a simple interactive program.
- Use an if statement for decision making
- Use loops for repetition
- Use nesting

SUBMISSION POLICY

Your workshops are divided into two sections; in_lab and at_home.

The "in_lab" section is to be completed **during your assigned lab section**. It is to be completed and submitted by the end of the workshop. If you do not attend the workshop, you can submit the "in_lab" section along with your "at_home" section (a 20% late deduction will be assessed). The "at_home" portion of the lab is **due the day before your next scheduled workshop**

All your work (all the files you create or modify) must contain your name, Seneca email and student number.

You are responsible for regularly backing up your work.

IN-LAB: ITEM CLASS (70%)

Download or clone workshop 3 from https://github.com/Seneca-144100/IPC-WS3

Code a program that does the following:

1- Print the title of the application.

- Print:
  >`--=== IPC mark Analyser ===---`< and (go to newline)
- The first character of the title must be on column 8.

2- Prompt the user to enter the number of marks that are to be entered and make sure this value is between 3 and 40. (Assuming that in this college there are no classes with less than 3 or more than 40 students)
   - Print the following message:
     >`Please enter the number of marks(between 3 and 40):` <
   - In a loop get the number of marks
   - If the number is less than 3 or greater than 40 print:
     >`Invalid number, enter a number between 3 and 40 inclusive:` <
   - Exit the loop only if the value is between 3 and 40 inclusive otherwise repeat the loop.
3- Start getting the marks one by one and add them to a total value that you have created for this purpose.
   - Create a loop that repeats to the number of marks entered
   - Prompt the user in each repetition by a row number and ">"
   - In a loop get the marks up to the amount entered by the user.
   - Add the value to the total. (Make sure the total is initialized to zero).
4- Divide the total value by the number of marks to find the average of all marks.
5- Print the result of the division.
   - Print >`The average of all marks in this group is XX.X.`<
   - Replace "XX.X" with the calculated average.
   - There should be one digit after the decimal point.
6- Prompt the end of the program and end the program.
   - Print >`Program Ended.`<

Output Example:

```
       --=== IPC mark Analyser ===---
Please enter the number of marks(between 3 and 40): 2
Invalid number, enter a number between 3 and 40 inclusive: 41
Invalid number, enter a number between 3 and 40 inclusive: 5
1> 45
2> 90
3> 57
4> 89
5> 32
The average of all marks in this group is 62.6.
Program Ended.
```

For submission instructions, see the SUBMISSION section below.

IN_LAB SUBMISSION:

To test and demonstrate execution of your program use the same data as the output example above or any information needed….

If not on matrix already, upload your **`marks.c`** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account: (replace profname.proflastname with your professors Seneca userid)

```
~profname.proflastname/submit ipc_w3_in_lab <ENTER>
```

and follow the instructions.

AT_HOME: TITLE (20%)

After completing the in_lab section, upgrade marks.c to make sure that the marks entered are valid (between 0 and 100 inclusive) and also provide the following additional stats on the marks entered:

1- Total number of students who passed (greater than or equal to 50) and their average
2- Total number of students who failed (less than 50) and their average
3- Highest mark
4- Lowest mark

Making sure the marks entered are valid:

Use the same logic you used for getting the number of marks and apply it to where you are getting the mark. If the mark entered is not valid print this message and scan the mark again:
>`Error: Enter values between 0 and 100 inclusive.\n`<
until a valid mark is entered.

Additional stats:
First define variables for all the statistic values that need to be printed at the end:

1- Number of passes
2- Number of fails
3- Sum of passed marks
4- Sum of failed marks
5- Highest mark (initialized to 0; lowest mark possible)
6- Lowest mark. (initialized to 100, highest mark possible)

In the loop in which marks are being entered, after each entry examine the value of the mark entered:

- If it is a pass, add one to the number of passes and add the value of the mark to the sum of passed marks.
- If it is a fail, add one to the number of fails and add the value of the mark to the sum of failed marks.
- If the value is higher than the highest mark, set highest mark to the value read.
- If the value is lower than the lowest mark, set the lowest mark to the value read.

After all the marks are entered and examined, divide the sums by the corresponding number of marks to get the average and print the results:

```
Total of X students passed with an average of XX.X.
Total of X students failed with an average of XX.X.
Highest mark in group: XX
Lowest mark in group: XX
```

Output Example:

```
       ---=== IPC mark Analyser ===---
Please enter the number of marks(between 3 and 40): 5
1> 45
2> -1
Error: Enter values between 0 and 100 inclusive.
2> 90
3> 57
4> 110
Error: Enter values between 0 and 100 inclusive.
4> 89
5> 32
Total of 3 students passed with an average of 78.7.
Total of 2 students failed with an average of 38.5.
Highest mark in group: 90
Lowest mark in group: 32
The average of all marks in this group is 62.6.
Program Ended.
```

AT-HOME REFLECTION (10%)

Please provide brief answers to the following questions in a text file named **reflect.txt.**

1) Name all the iteration constructs?
2) Explain when is better to use “do while” rather than “while” loop?
3) What is a conditional expression?

AT_HOME SUBMISSION:

To test and demonstrate execution of your program use the same data as the output example above.

If not on matrix already, upload your **marks.c and reflect.txt** to your matrix account. Compile and run your code and make sure everything works properly.

Then run the following script from your account: (replace profname.proflastname with your professors Seneca userid)

```
~profname.proflastname/submit ipc_w3_at_home <ENTER>
```

and follow the instructions.