CIS 236 – Programming in C

Program:	**Week 06**
Points:	**20**
Chapters:	**5, 6**

Description

Write a program that prints a list of levels in a game, along with a count of how many players are at each level. The level will be determined by the amount of health points, using this guide:

Level	Points
Level 1	0 – 9
Level 2	10 – 19
Level 3	20 – 29
Level 4	30 – 39
Level 5	40 – 49
Level 6	Anything 50 or over

Player points are entered by the user. Your program must use a sentinel-controlled loop to process the data typed by the user, until the user types -1.

After all the data has been entered, display the final counts in a nicely formatted chart.

Learning Objectives

In this assignment, you will practice:

- Implementing a selection control structure
- Implementing a repetition control structure
- Using an array to store related data
- Implementing functions with input arguments and return value
- Displaying formatted output to stdout

Requirements

1. The number of players at each of the six levels in this game is stored in an **array**. The first element in the array stores the count for Level 1, the second element in the array stores the count for Level 2, etc. The array must be declared within main.
2. Use a **symbolic constant** to represent the size of the array wherever needed.
3. Your code must use **these two functions** (in addition to main):
    a. update_level
    b. display_levels

Requirements for the update_level Function

Purpose:	This function uses the points passed in as an input parameter, determines which level should be updated, and updates the count for that level in the array.
Parameters:	1. points for a player (as an integer) 2. the array
Algorithm:	Use a selection structure to determine which element of the array should be incremented.
Return value:	None

Requirements for the display_levels Function

Purpose:	This function displays the contents of the array after all the data has been entered.
Parameters:	1. the array
Algorithm:	Use a repetition structure statement to display the contents of the array neatly aligned as in the sample output. Use the symbolic constant!
Return value:	None

Requirements for main

1. Variable declarations in main should only include the variables needed within main. Do not declare all the variables needed in the program here – only the ones needed within main.
2. Be sure to initialize the array to 0's.
3. The main function must use a **sentinel-controlled loop** to process the data typed by the user, until the user types -1. The loop body will call the **update_level** function with the appropriate arguments.
4. After the loop is complete, call the **display_levels** function with the appropriate arguments.

Sample Run 1

```
Player points (-1 to quit): 5
Player points (-1 to quit): 15
Player points (-1 to quit): 25
Player points (-1 to quit): 35
Player points (-1 to quit): 45
Player points (-1 to quit): 55
Player points (-1 to quit): 37
Player points (-1 to quit): 24
Player points (-1 to quit): -1


T O T A L S

Level 1   1
Level 2   1
Level 3   2
Level 4   2
Level 5   1
Level 6   1
```

Sample Run 2

```
Player points (-1 to quit): -1


T O T A L S

Level 1   0
Level 2   0
Level 3   0
Level 4   0
Level 5   0
Level 6   0
```

Requirements for Full Credit on This Project

SUBMIT YOUR OWN WORK – Plagiarism is not tolerated in this course. Please review the section on the Academic Honor Code in the syllabus. I will not hesitate to drop you from this class if you submit a program that is plagiarized.

COMPLETE AND ACCURATE – Your program must compile, execute, and give accurate output.

FOLLOW ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS – Follow the instructions as written for completing this project, even if you [think you] know a “better” way to do something.

COMMENTS – Include comments in your code. There must be a comment at the top of your program that includes your name, the program number, and a description of the program. There must be comments at each important step in your program that describes that step. Every variable must include a comment describing its purpose.

BEST PRACTICES – Follow best practices in C programming as discussed in class and in the textbook, including, but not limited to, appropriate use of white space, indenting, alignment, meaningful identifier names, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.

SUBMIT ONLY .c SOURCE CODE – Pay attention to the file extension of the source code file you submit. I will deduct points for not following this requirement.

SUBMIT ALL FILES BEFORE THE DUE DATE – Submit your .c source code file to the dropbox for this assignment on Canvas before the due date. Do not submit executable files. Do not submit project files from an IDE. I will not accept links to online storage.