CS 211 Introduction to Programming
Fall 2019

Programming Assignment 6

(Assigned: September 26, 2019; Due: October 3, 2019)

The goal of this assignment is to write a program that handles values of type **`string`** *and performs character I/O, by the member function* **`get`** *described in Section 6.3 of the textbook.*

Write a C++ program that prompts the user to enter a zip code, converts that zip code to a bar code representation described below, and displays the resulting bar code.

Zip Codes: The user may enter a zip code in one of the following three acceptable zip code formats:

1. **Short:** A 5-digit zip code, such as **`67218`**.

2. **Medium:** A hyphen and a 4-digit extension are appended to the short zip code to produce the medium version, such as **`67218-1234`**.

3. **Long:** A plus sign and the last two digits of the street address or box number are appended to the medium zip code to get the long version, such as **`67218-1234+12`**.

Checksum Digit: This is a unique digit (between 0 and 9), which when added to the sum of ***all*** the other digits in the zip code, will make the sum a multiple of 10. For example, if the zip code is **`67218`**, then 6 + 7 + 2 + 1 + 8 = 24, so the checksum digit must be 6. As another example, if the zip code is **`67218-1234+12`**, then 6 + 7 + 2 + 1 + 8 + 1 + 2 + 3 + 4 + 1 + 2 = 37, so checksum digit is 3. The usage of the checksum digit to obtain the bar code is explained below.

Bar Codes: The format for a bar code is: **`| Z E D C |`** where:

- **`|`** is a vertical bar character,

- **`Z`** is the 5-digit *encoded* short bar code (for all zip codes),

- **`E`** is the optional 4-digit *encoded* bar code extension (for medium and long zip codes only),

- **`D`** is the optional *encoded* 2-digit delivery point (for long zip codes only),

- **`C`** is the *encoded* checksum digit (for all zip codes).

Encodings of digits employ two bars: the short one '**`.`**' and a tall one '**`|`**', as follows:

```
1 = ...||      2 = ..|.|      3 = ..||.      4 = .|..|      5 = .|.|.
6 = .||..      7 = |...|      8 = |..|.      9 = |.|..      0 = ||...
```

Bar codes of the three zip code examples shown above are:

```
67218 =
     |.||..|...|..|.|...|||..|..||..|
67218-1234 =
     |.||..|...|..|.|...|||..|....||..|.|..||..|..|.||..|
67218-1234+12 =
     |.||..|...|..|.|...|||..|....||..|.|..||..|..|...||..|.|..||.|
```

Observe that a short bar code requires 32 bars, a medium bar code requires 52 bars, and a long bar code requires 62 bars.

Notes:

- The name of your source file ***must*** be: `pa6.cpp`

- A typical sample run of your program should look like (output **`blue`**, input **`red`**) :

```
Zip code: 67218
Bar code: |.||..|...|..|.|...|||..|..||..|
```

- Another run could look like (output **`blue`**, input **`red`**) :

```
Zip code: 67218-1234
Bar code: |.||..|...|..|.|...|||..|....||..|.|..||..|..|.||..|
```

- Employ **`cin.get`** to read the zip code. *You may assume that the input does not contain any errors, and the zip code entered by the user is in one of the three acceptable formats.* However, there is extra credit for performing simple error detection in the input, as explained in the grading rubric below.

- First decide on an overall strategy for constructing the bar code. It is possible to construct it on the fly as each input character is read. Alternatively, it is also possible to first read the entire input, and store its relevant information, perhaps in an array of characters, and then analyze that array to construct the bar code. Visualize in your mind the strategy you decide to adopt, write a pseudo-code that follows that strategy, and only then start writing the program code.

Assignment Submission Instructions:

Submit just your `pa6.cpp` file via Blackboard. Please do not submit any other file contained in your Visual Studio 2019 project for this assignment.

<u>Grading Rubric:</u>

Your program submission will be graded according to the following rubric:

The name of the submitted program source file is `pa6.cpp` (*not* `pa6.c` or `Pa6.cpp` or `PA6.cpp` etc.)	1
The first two lines (student's name, and student's WSU ID) of the required comments at the top of the program are included	1
The variable identifier names used are descriptive (single letter names for loop counters are OK)	1
A good indentation scheme is used throughout the program	1
The program compiles and links with no errors or warnings	5
The program works correctly for the zip code 67208	5
The program works correctly for the zip code 67260-0083	5
The program works correctly for the zip code 03902-7930+40	6
TOTAL	**25**

Extra credit of up to **5 points** will be awarded for detecting errors in the input, such as an illegal character, or inputs like `123-456+7`, `12345+6789`, etc. For such inputs, simply reporting that the input is erroneous and not displaying any bar code is acceptable program behavior, to get the extra credit.