**3460:209 PROGRAMMING PROJECT 2: GETTYSBURG DICE**

From July 1 to July 3, 1865, the Civil War and the attention of two nations focused on the area near Gettysburg, Pennsylvania. The Confederate Army of Northern Virginia, under the command of General Robert E. Lee, faced off against the Union Army of the Potomac controlled by Major-General George G. Meade. The actual battle, won by Meade, saw 7000 casualties, 32000 wounded soldiers and 10000 taken prisoners or unaccounted for.

The purpose of this assignment is to create a (*very*) simple simulation which re-enacts the Battle of Gettysburg using coin flips. That said, this is still complicated enough that we should do some high-level planning. Such a road map, known as *pseudocode*, for this program appears below, as well as an indication of what background you need to finish each individual task. Because we will be constructing this program out-of-order as we build up individual concepts, this outline will help you determine where the individual piece that you’re writing at a given time fits into the big picture. In fact, you may want to insert the lines of pseudocode as comments into your code to help document what you’re doing as you go along.

**PSEUDOCODE OUTLINE**

* Declare variables Chapter 2
* Print beginning instructions Chapter 1
* Loop through the 9 battles Chapter 5
* Print the battle’s name Chapter 5
* Prompt user for guess Chapter 2
* Echo guess Chapter 2
* …following correction Chapter 4
* Get and print coin flip Chapter 3
* …as either heads or tails Chapter 4
* Determine and print battle winner and increment

proper score Chapter 4

* Print running score Chapter 3
* Print final score and winner Chapter 4
* Print final historical data Chapter 1

**PART 1 (after Chapter 1)**

* Log into the network and create the folder *CSI* in your disk space. Here you will store all files related to any programming assignments for this class. I *strongly* advise you to further organize your work by creating additional subdirectories of *CSI*, one for each assignment. All work is to be placed in the Projects/Project2 folder.
* Start an IDE. Below appears the standard header.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34 | /\*  ACADEMIC INTEGRITY PLEDGE  - I have not used source code obtained from another  student nor any other unauthorized source, either  modified or unmodified.  - All source code and documentation used in my program  is either my original work or was derived by me from  the source code published in the textbook for this  course or presented in class.  - I have not discussed coding details about this project  with anyone other than my instructor. I understand  that I may discuss the concepts of this program with  other students and that another student may help me  debug my program so long as neither of us writes  anything during the discussion or modifies any  computer file during the discussion.  - I have violated neither the spirit nor letter of these  restrictions.  Signed:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date:\_\_\_\_\_\_\_\_\_\_\_\_\_  COPYRIGHT (C) 2015 *(your name)*. All Rights Reserved.  *(Description of program)*  author *(your name)*  version 1.01 *(date)*  \*/ |

(Line numbers are purely for reference and are not to be typed in by you.) Type it in and save to your folder under the title gettysburg.cpp. Update the name of the public class and the header information accordingly.

* Following the header, write a main method that displays the game’s beginning and ending screens. To begin the program print out

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26 | LET THE    FLIP OF THE COIN REWRITE HISTORY  /----------------------\\  | RETURN TO GETTYSBURG |  [\\----------------------/](file://\\----------------------\)  It is July 2, 1863, the second day  of the Battle of Gettysburg...  In this re-enactment of the second day  of battle, you are George G Meade in  charge of the Unionists  We will flip a coin to determine the  outcome of the small battles for  territories within the Gettysburg area  You have to predict whether the coin  will come up heads or tails... |

Your code will be a sequence of cout statements. The double ‘\’ is not a typo; this is what’s required to trick the computer into printing out a single backslash, for reasons that we will explain later. For now just instruct the computer to display this text exactly as it appears.

Next, in the same file, just as before, write code to print the end title:

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | In the actual Battle at Gettysburg,  Lee acknowledged defeat on July 4...  The Confederates lost about 4000 men,  18,000 were wounded, and 5,150 were  captured or missing...  The Unionists lost 3,155 men,  14,529 were wounded, and 5,365 were  captured or missing... |

* For this and all other programming assignments note the following:
* This is not a group project. If you are having difficulty, consult with the professor, the teaching assistant, or the lab instructor only.
* Follow the applicable elements of the documentation style and guidelines given in the text with regard to spacing, indentation, identifier names, comments, etc.

**PART 2 (after Chapter 2)**

* Make a copy of your code from Part 1 and place it in a separate subfolder. Since you are going to be making changes to your code, it is better to keep your previously working copy as-is and edit only this new copy.
* Before displaying the opening title declare integer variables for your score, the computer’s score and the coin. Initialize all of these variables to zero. Also declare a string variable for the user’s guess.
* Following the opening instructions, prompt the user for input (heads or tails) and echo back the choice. For example

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | Now General Meade  Do you predict the coin will come out  heads (H) or tails (T)?  Choose H or T: **H**  You have chosen H  Stand by to fight, General Meade... |

(Here the user input is shown in **boldface** type.)

* Finally, before printing the end title, print a summary of the action to this point as follows:

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | So far: Meade 0 victories, Lee 0 |

Remember that, instead of just printing zeros, you are printing the values of the score variables declared earlier.

**PART 3 (after Chapter 3)**

* As before make a copy of your code from Part 2 in a new folder and make changes only to this new copy.
* Simulate a coin flip by generating a random integer less than 2 (i.e., either 0 or 1). Store it in your coin variable (declared earlier) and display the result. For example, if the computer kicks out a zero as the random number, you should see

|  |  |
| --- | --- |
| 1  2  3  4  5 | The result of the coin flip was 0 |

Remember the import statements for these libraries, positioned after the Academic Integrity Pledge and before the code for the program.

* Submit this revised program as you did for Part 2. Furthermore, when completing this assignment, observe all stylistic guidelines discussed previously.

**PART 4 (after Chapter 4)**

* As before make a copy of your code from Part 3 in a new folder and make changes only to this new copy.
* Revise your program from Part 3 by doing the following:
* Account for a user messing up. If the user enters something other than “H” or “T” for their guess, set the guess to “H”. Remember both now and in what follows to use a case-insensitive test on the user’s input.
* Echo back the user’s choice by saying “heads” or “tails” instead of “H” and “T”. In other words, if the user enters an ‘H’, instead of printing “You have chosen H”, print “You have chosen heads”.
* Translate the coin flip from numbers to words. In other words, if the computer generated a random zero, print “The result of the coin flip was heads” instead of “The result of the coin flip was 0”. Similarly match a random one with “tails”.
* Remember from now on: a value of zero for the coin variable means “heads”, a one means “tails”. With this in mind determine the battle’s winner.
* If the user guess and the random result are the same print the following

|  |  |
| --- | --- |
| 1  2  3 | Well done, General Meade...  You have defeated Lee |

and add one to your (the human’s) score.

* Otherwise print

|  |  |
| --- | --- |
| 1  2  3 | I am sorry, General Meade...  but the battle goes to Lee |

and add one to the computer’s score.

* Immediately before printing the end titles, print a summary declaring a victor. In other words, after all battles have been fought, if the human’s score is greater than the computer’s, print the following

|  |  |
| --- | --- |
| 1  2  3  4  5 | The results show that  George G Meade  won the majority of the battles  and is thus declared the victor  on July 2 1863. |

If the computer’s score is larger, print “Robert E Lee” instead of “George G Meade” in this statement.

**PART 5 (after Chapter 5)**

* As before make a copy of your code from Part 4 in a new folder and make changes only to this new copy.
* Enclose everything from the user guess for the coin flip to the printing of the running score inside a for loop running from 1 to 9. Remember to adjust all indentations accordingly.
* Inside the loop, immediately before getting the user guess, print the name of the individual battle based on the value of the loop index variable.

|  |  |
| --- | --- |
| **If the index variable is…** | **the battle name is…** |
| 1 | Little Round Top |
| 2 | Cemetry Ridge |
| 3 | Devils Den |
| 4 | The Wheatfield |
| 5 | The Peach Orchard |
| 6 | Culps Hill |
| 7 | Zieglers Grove |
| 8 | Seminary Ridge |
| 9 | Gettysburg |

Your print statement should display the word “Battle ”, the battle number, a “ – ”, and then the battle’s name.

* Check and make sure this all works right. The beginning and ending instructions, as well as the declaration of a final winner, should print only once. The game itself, with the guess at the coin flip, the actual coin flip, the declaration of an individual battle winner and the display of the running score should happen 9 times in all.
* And you did it! Submit this final version of the program as you did for Part 4. Furthermore, when completing this assignment, observe all stylistic guidelines discussed previously.

Submission Instructions – for projects

On the student Subversion repository, navigate to **Projects/Project2.** Create programs formatted under the name of the project.

**We will not grade a commit made after 11:59 p.m. on Monday, March 2, 2015.**

**Reference**: *Tim Hartnell’s Executive Games for the IBM PC & XT*, 1984.

*Last updated 8.23.2012; previous revision 2.19.2007.*