# CSC134 Project

|  |  |
| --- | --- |
| Project Number | Project 1 |
| Project Name | Morse Code Project |
| Project Filename | Firstname\_Lastname\_Project1 (example: *Jessica\_Smith\_Project1*) |
| Chapter Review | Chapters 1 – 3Use only the concepts covered in Chapters 1 - 3 to complete this project. Failure to do so may result in a 0 for the assignment. Students are expected to complete the practice programs before starting the project. |
| Points | 50 |
| Assistance | Instructors and teaching assistants have been available to assist with all practice work. Students are now expected to complete this project without assistance from others (this includes receiving assistance from individuals inside or outside of CPCC). Students should consider projects as non-proctored exams. Please review the academic integrity policy on your syllabus.Please note: students can continue to receive assistance with practice work up to 5:00 p.m. on the due date (review the late period on the syllabus). |
| Project Description | In the 1850's, the hottest new technology was the telegraph. Messages or telegrams could be sent via electrical forces traveling through a wire. Write a program that will generate a bill for sending a telegram, accept payment and instruct cashier how to give change. The amount owed is based on the number of words sent. Customers are charged at a rate of $1.50 for blocks of 5 words and $0.50 for single words. Your program should calculate the smallest charge possible for the given amount of words sent. Named constants must be used in the calculation.  **Instructions:**   1. Prompt user for the customer’s name, street address, city, state, zip code and number of words sent. 2. Calculate and display a telegram bill with all information specified in the **Sample Output** on page 2. 3. After displaying the telegram bill, prompt user for the amount of money received from the customer. The user should supply the amount given in pennies, for example 275 instead of 2.75. 4. Calculate the amount of change owed and display the number of dollars, quarters, dimes, nickels, and pennies that the customer should receive. 5. Your program's output should be formatted in the same manner as the **Sample Output** on page 2. 6. Save your **.cpp** file using the Firstname\_Lastname\_Project1 naming format. Upload in Moodle. Verify that your file actually uploaded. 7. **Bonus Pts (5pts)**. After completing all instructions above, add code to randomly generate the customer’s payment. That number should be an integer. Use the number for the payment rather than the one received from the user (overwrite the value stored in your payment variable with the random number). Each time the program is run, a different random number should be generated for the payment. You must follow all these instructions to receive full credit. |

# Sample Output

|  |
| --- |
| Enter the name of the customer: Larry Smith  Enter street address: 122 Main Street  Enter city: Charlotte  Enter state: NC  Enter zip code: 23499  Enter the number of words sent: 157  Larry Smith  122 Main Street  Charlotte, NC 23499  Amount Owed: $47.50  Enter the amount received from customer: 5000  Denomination Number  -------------- ---------------  Dollars 2  Quarters 2  Dimes 0  Nickels 0  Pennies 0 |

# Grading Score Card: 50

|  |  |  |
| --- | --- | --- |
| **OBJECTIVE** | **DESCRIPTION** | **POINTS** |
| User Interface Well Designed | User prompts provide adequate direction and have a professional, easy to read format. | 5 |
| Named Constant Used | Named constant(s) used appropriately in calculation(s) AND naming convention for constants is followed. | 5 |
| Required Value(s) Correctly Calculated | Program displays correct output for all input values. | 20 |
| Algorithm is efficient. Code is clean, simple and easy to understand. | 5 |
| Output Correctly Formatted | Ouput is formatted in a professional, easy to read format that matches the **Sample Output** on page 2. | 10 |
| Documentation and Programming Style | * Multi-line comment present at the beginning of the program that contains: your **Name**, **Date** and **Purpose** which fully describes what your program does. * Adequate descriptive comments used throughout program. There must be at least four comments to receive credit for this step. * Proper indentation and spacing used throughout program. * Proper naming conventions used throughout program. | 5 |
| **TOTAL** |  | **50** |
| **Bonus Points** | A random number is used for the customer’s payment rather than input from the user by overwriting the user’s input with the random number. Each time the program is run, a different random number is generated for the payment. | **+5** |
| **Point Deduction** | Program does not compile without errors. | **-10** |