**Project 1**

**<Jumble>**

**CSC-5 42829**

**Name: Cervantes, Jose**

**Date: 5/04/2016**

**Introduction**

Game Title: JUMBLE

This is a puzzle game that requires the ability to unscramble words that have been arranged in an unusual order. The player can require assistance bye typing in “hint” if he/she requires it. The game has two levels, in which the game gets difficult as he/she progresses in the game. The game JUMBLE is also a two player game in which player one inputs a four to eight letter word followed with a hint, to assist player two. Then player two has to figure out what player one has typed in. If player two is unable two solve the word, then he/she can type in “hint”. The hint that player one typed in will show on the screen and the user can repeat this game as many times as they wish. This program also has a file name called score.txt which will record all the attempts the user has tried.

**Summary**

Project Size: about 850 lines

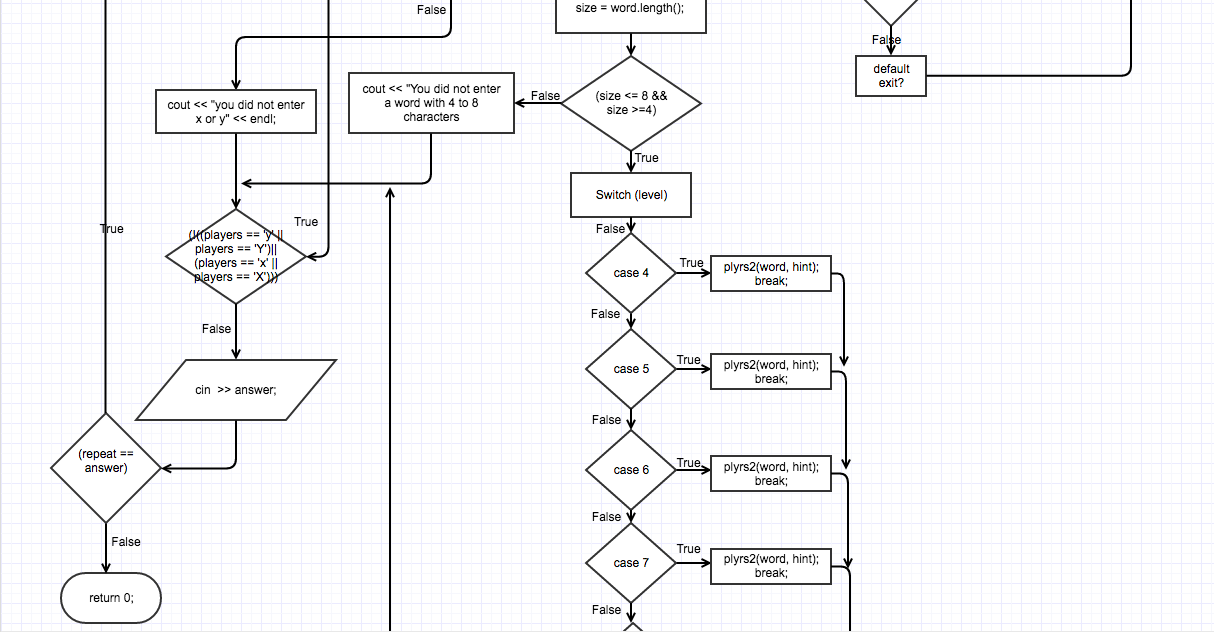
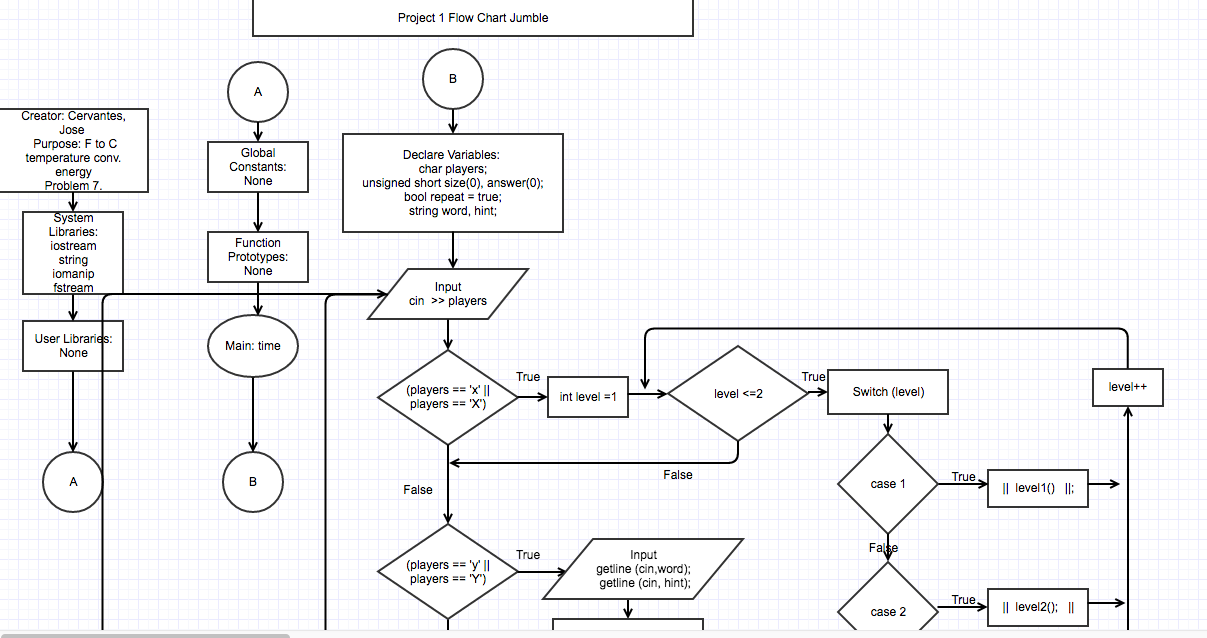
The number of variables: about 15

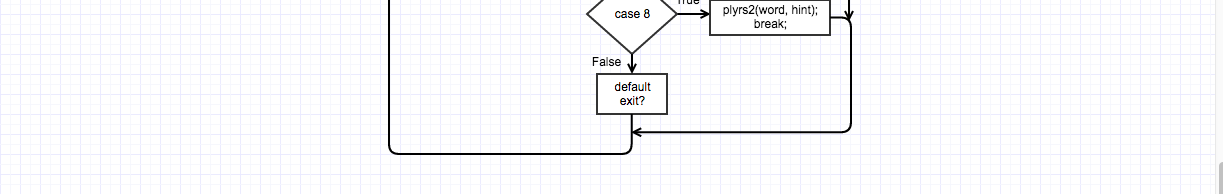
The number of comments: about 138

This project includes for loops, while loops, if and else, switch and many more concepts we learned. This project can be improved by using arrays to store more words in a word bank and generate random words from the array. If I were to do that with if and else statement the code will be incredibly long. And maybe create a time limit in how long the user has to solve the word. I spent about a week on this project. I had trouble unscrambling the word that a user would enter, but looked up a function called “ string.at(#) ” which will return a character at any position of the word.

**Flow Chart**

**I Only flow charted the main function which contained all the main concepts because the functions would take a good amount of pages to flow chart**

****

****

**Major Variables**

|  |  |  |  |
| --- | --- | --- | --- |
| **TYPE** | **Variable Name** | **Description** | **Location** |
| **char** | players | Holds the number of players in a game | char players;  Lline 62 |
| **unsigned short** | size(0) |  | unsigned short size(0), answer(0);  Line 64 |
|  | answer(0) | Holds the the value if  The user wants to repeat the game | unsigned short size(0), answer(0);  Line 64 |
|  | attempts(0) | Holds the number of  Attempts the user needed to solve the word | unsigned short attempts(0)  Line 214 |
| **ofstream** | out | Stores the score of previous games in a .txt file. | out.open("hello.txt",ios::app);  line 222 |
| **string** | Word | Take the user inputs for the word they want player 2 to solve. | string word, hint;  Line 74 |
|  | Hint | Holds the key word hint if the user needs help | string answer, word1, hint;  Line 74 |
|  | answer | takes in the user Guess of the scrambled word | string answer, word1, hint;  Line 208 |
|  | word1 | Holds the first word the level | string answer, word1, hint;  Line 208 |
|  | Word2 | Holds the second word the level | string word2;  Line 317 |

**C++ Constructs**

|  |  |  |
| --- | --- | --- |
| **Chapter**  **Gaddis** | **New Syntax and Keywords** | **Location** |
| **2** | cout << “”; | Everywhere; |
|  | Unsigned short | unsigned short attempts(0); |
|  | Int variable | for(int level =1; level <=2; level++) |
|  | #include <string> | string answer, word1, hint; |
|  | Equality operators and relational operators (==, !=, >, =, <=) | while(!((players == 'y' || players == 'Y') || (players == 'x' || players == 'X'))) |
|  | Arithmetic operators (+, -, \*, /) | attempt = attempt + 1; |
| **3** | User Interactivity | cin >> answer; |
|  | cin.ignore() | Line 97 |
|  | getline (cin, variable) | getline (cin, word); |
| **4** | Equality operators and relational operators (==, !=, >, =, <=) | if (answer == "hint") |
|  | Bool | bool repeat = true; |
|  | If/else function | if (players == 'x' || players == 'X') |
|  | #include <iomanip> | Iomanip library |
|  | Setw() | cout << setw(47)<<"\*" << endl; |
|  | Nested If Else Statements | Line 82 |
|  | And Or operators(|| &&) | else if(players == 'y' || players == 'Y') |
|  | Comparing Strings | if ((!(answer == word1))) |
|  | Switch Statement | switch(size) |
|  | Break operator | case 4 : plyrs2(word, hint); break; |
| **5** | Increment Operator | attempts++; |
|  | While loop | while(repeat == answer) |
|  | Do While loop | do { }while (!(answer == word1)); |
|  | For loop | for(int level =1; level <=2; level++) |
|  | File storage | ofstream out;  out.open("hello.txt",ios::app); |
| **6** | Functions | void level1();  void level2();  void plyrs2(string , string ); |
|  | #include <fstream> | Fstream library |
|  | #include <iostream> | Iostream library |

**Pseudo code**

Declare players as char variable

Declare size, answer as unsigned short

Declare repeat as a true bool

Declare word, hint as strings

Do

Do

X = 1 player

Y = 2 players

Ask user for amount of players

Input player;

If (players == 'x' || players == 'X')

| For int level =1 ; level<=2; level++

| Switch (level)

| Case 1 level1();

| Case 2 level2();

|

Else if (players == 'y' || players == 'Y')

| Ask user for a 4 to 8 letter word

| Input word

| Ask user for a hint of the 4 to 8 letter word

| Input hint

| Calculate size by getting the number of letters in the word

|

| | if (size <= 8 && size >=4)

| | switch (size)

| | case 4 plyrs2(word, hint);

| | case 5 plyrs2(word, hint)

| | case 6 plyrs2(word, hint)

| | case 7 plyrs2(word, hint)

| | case 8 plyrs2(word, hint)

| |

| |else You did not enter a word with 4 to 8

|

else the user did not enter x or y

While (!((players == 'y' || players == 'Y')|| (players == 'x' || players == 'X')))

Ask the user if the want to rerun the program again

Input answer

While repeat == answer