Hangman  
Project 1

CSC-5 Intro C++

Bradd Sean Carey

25 October 2013

**Table of Contents**

**Table of Contents** ...................................................................................................................... 2

**1- Introduction** ............................................................................................................................ 3

Gameplay and Rules ...........................................................................................................

Quick After Thoughts ...........................................................................................................

**2- Development** .......................................................................................................................... 4

Approached Strategy ..........................................................................................................

Research ..............................................................................................................................

Adjusted Strategy ............................................................................................................. 5

Dropped Functionalities and Future Implementations ................................................. 6

**3- Flowchart** ................................................................................................................................ 8

**4- Code**....................................................................................................................................... 11

**1 - Introduction**

**Rules and Gameplay -**

This program implements as basically as possible the common game of Hangman. While the program assumes basic familiarity with the game there is a basic introduction included. The game will present the user with an ascii rendition of a gallows and a number of underscores that each represent a letter that needs to be guessed. The user enters one letter at a time and if the letter exists in the word the underscore is replaced with the chosen letter. If the user chooses a letter that isn’t in the word then the gallows get an additional part of a man to hang. If six wrong guesses are made before the word is solved then the player loses. The majority of the code was written on the same day I put my Christmas tree up so all the words choosable are Christmas based words.

**Quick Thoughts After Program -**

The game was implemented very simply and assumes a common sense knowledge about the player. This prototype game was designed this way intentionally so that the basics of the game could be realized and implemented effectively without microfocusing beyond the scope of the project. In addition to avoiding my nature to become and insatiable perfectionist with details the simple nature of the program allows room to grow as I continue to learn more C++.

All this being said is because I forewent a feature I would have been a great quality of life improvement, input validation. While it’s more than safe to assume my players know not to enter integers, floats, multiple letters or the same letter several times, validation for these things would be simple enough to input. They weren’t due to the aforementioned reasons but also because going into this project I wasn’t quite sure if I even could implement some of the features with my current working knowledge of C++. In fact a good deal of the program relied on strategies that needed quite a bit of research to make happen. There will be more on that in the relative section.

**2 - Development**

**Approached Strategy -**

Since the game is well known and easy to implement my strategy was to just play the game a few times and try and break down each step in my head. A word has to be chosen, then the word needs to be broken down not only in substance but into a size of the word. The size is then output to the player. Based on the size the player then guesses letters relevant to the size. The letters need to replace the blank spaces if they are guessed correctly and chances need to be taken away if incorrect guesses are made. Breaking down the game into what’s needed by the computer and the user helped immensely with the structuring of sequential programming.

**Research -**

Quite a bit of research was needed to create the game’s basic functionality:

* Arrays
  + I’m not sure if this program would be even possible without arrays and even if it was possible coding it would have been unnecessarily tedious and the game would definitely be less appealing. Arrays allowed me to not have hundreds of variables and were used to store an in program database of strings for potential words. Arrays were also used to store characters for guessed letters and compared to another array that had the actual word. Arrays made dreams come true here.
* Converting a String to a Char Array
  + This one was a puzzler at first, but ended up not being very complicated at all. In fact based on my research strings may just be char arrays in an of themselves. They are stored in memory (as far as I can tell) exactly the same. A simple for loop converted the string into a character array officially (42-44).
* Ascii to make primitive graphics
  + Although the art used for the gallows was unsigned I’d like to mention that the art itself was not mine. Including it wasn’t even my idea but the resources online are out there. There are entire databases of ascii art for every occasion and even though mine is very basic for every game and situation someone else has made a beautiful piece of art out of symbols that can be displayed from command prompt. I also learned that in order to use “ \ “ in a string you must put two for each backslash you want.
* string.length()
  + this statement saved quite a few if then or switch statements. The string object has a .length method in it so I could automagically detected string size and use the method itself to create array sizes. I used the string.length nine times in less than 200 lines of code. Quite useful.

**Adjusted Strategy -**

My original strategy was to make main an immaculate function. That is to say that main only called other functions. The reasoning behind this may have been more about being able to do it rather than for ease of programming but I ran into a rather big snag. Being that I self taught myself arrays I didn’t have the knowledge of pointers needed to make methods that return elements of arrays effectively. When I decided to use arrays I quickly realised that I was diving into a pool too deep to touch the bottom. So I abandoned making everything into a function until I understand arrays better.

In addition to the array snag I couldn’t create function prototypes with return type string. I’m still not entirely sure why this wasn’t working but in the timeframe of this project (especially with the detail desired for this report) I didn’t have time to learn what was going on there. Savitch details strings in greater detail later on anyway.

\*\*As I was typing this I just realized that I put my function prototypes before my namespace which is why Netbeans couldn’t identify string. Easily fixable in a future version.

**Dropped Functionalities and Future Implementations -**

* Input Validation
  + Sure you’re probably thinking that it’s pretty lame I didn’t include input validation and yes it would be an easy implementation:

bool chosen[27];

do{

cout<<”Input a letter: “;

cin>>letter;

}while(!chosen[c]||letter!=’a’||letter!=’b’... );

for(char test = ‘a’, int i = 1; i<=26 ; test++, i++){

if ( test == letter){

chosen[ i ] = 1;

}

}

* + There was the original implementation, that check to see if the input was not only a letter, but a letter that hasn’t been used before. The bool chosen array essentially notches off already picked letters by turning an element true if the letter has been used before. I thought it was pretty brilliant incrementing a character, but then another thought occurred to me.
  + If a player is dumb enough to pick a letter they already chose, they **deserve** **to be hung**. Maybe in the future I’ll implement this again but instead of stopping the input I’ll let it pass and just output a message reminding the player that he/she is an idiot. We’ll see, it’s more of an artistic choice if anything.
* Tracker to output already chosen letters
  + I wanted to put this in either to display all the letters and then erase the ones already chosen, or simply show the letters that have already been picked but I couldn’t find a place I like to output it. The gallows and underscores gave the game a nice simple feel and I didn’t want to intrude anymore. Maybe if we learn some GUI in the future I can make it pretty and maintain the feel of the game. Or maybe I could implement a menu someway so the user can choose to look up their letters if they so desire.
* Tracker to keep score of wins/losses
  + Actually I could this in just a few lines of code. Quick custom function that returns a score.

short wins = addWin(wins);

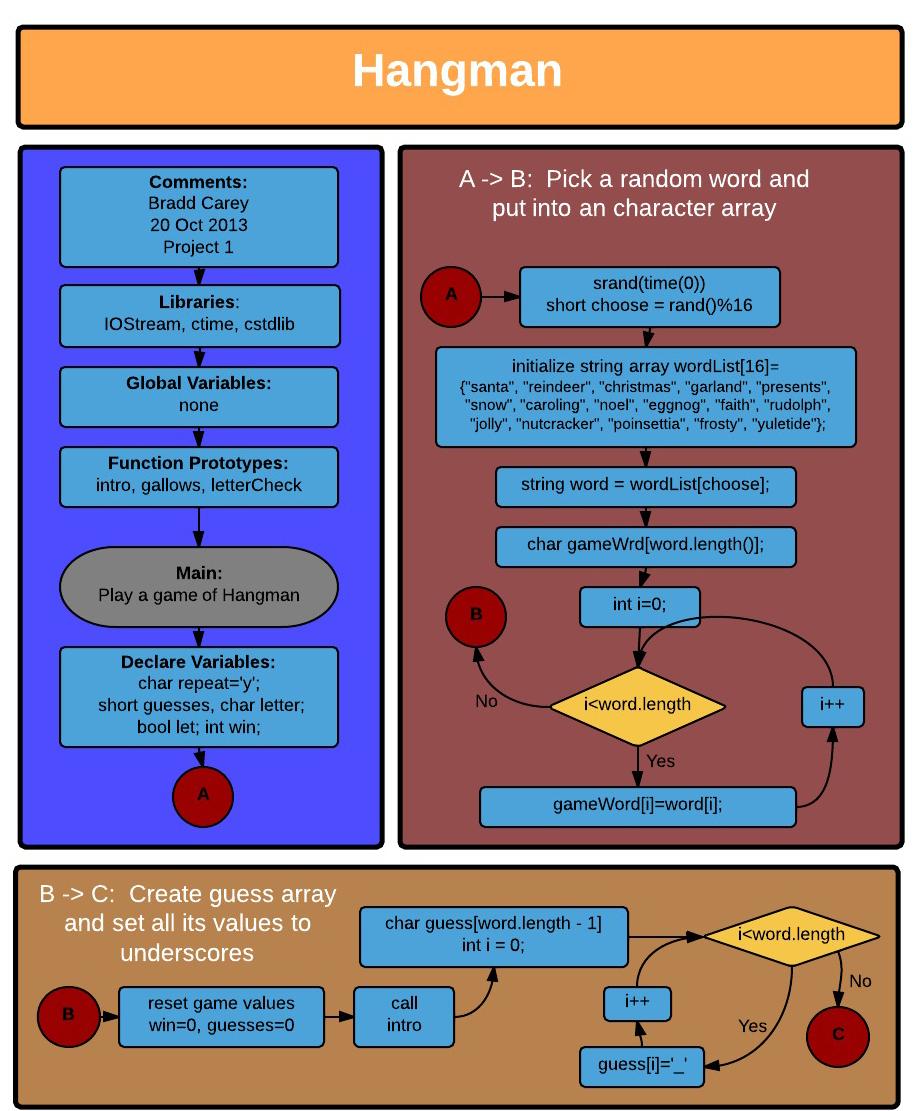
short loss = addLoss(loss);

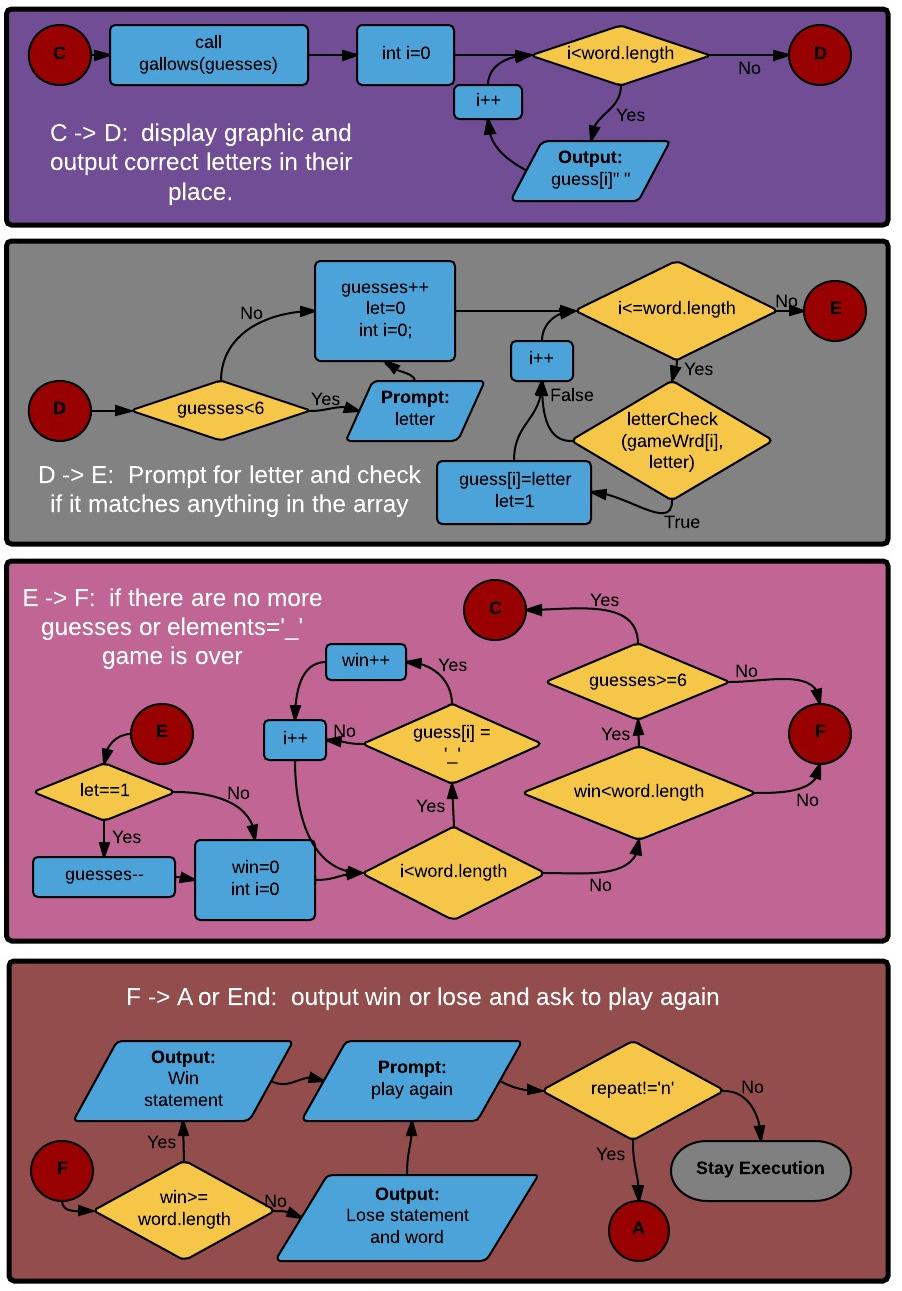
short addWin (wins){return wins++; }

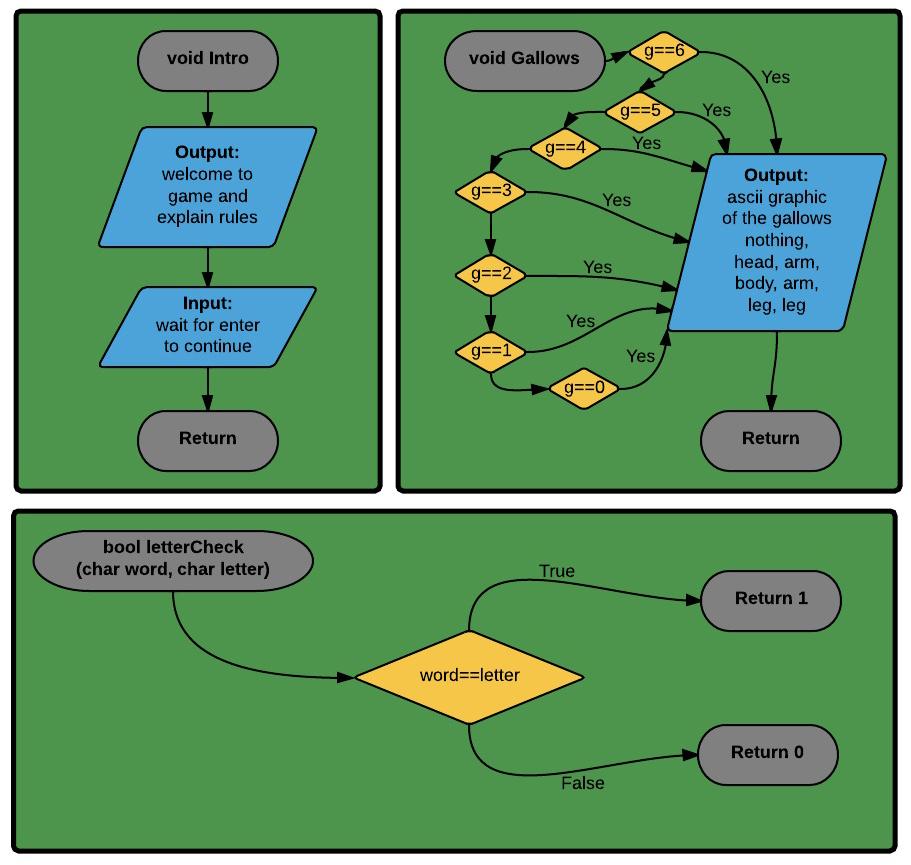
short addloss (loss){return loss++; }

* + I’d just have to throw this in with a cout before the ask to replay loop.

**3 - Flowchart**







4 - Code

1 /\* Carey, Bradd

2 \* 25 October 2013  
 3 \* C++ Project 1  
 4 \*/  
 5   
 6 //Libraries  
 7 #include <iostream>  
 8 #include <cmath>  
 9 #include <ctime>  
 10 #include <string>  
 11 #include <cstdlib>  
 12   
 13 //No Global Variables  
 14   
 15 //Function Prototypes  
 16 void **intro**();  
 17 void **gallows**(short);  
 18 bool **letterCheck**(char, char);  
 19   
 20 using namespace std;  
 21   
 22 int **main**(int argc, char\*\* argv) {  
 23   
 24 //Declare Variables  
 25 char repeat='y';  
 26 short guesses;  
 27 char letter;  
 28 bool let;  
 29 int win;  
 30   
 31 //Repeating Loop  
 32 do{  
 33 //Choose a string and convert it to a character array  
 34 srand(time(0));  
 35 short choose = rand()%17;  
 36 string wordList[17]={**"**santa**"**,**"**reindeer**"**,**"**christmas**"**,**"**garland**"**,**"**presents**"**,  
 37 **"**snow**"**,**"**caroling**"**,**"**noel**"**,**"**eggnog**"**,**"**faith**"**,**"**rudolph**"**,  
 38 **"**jolly**"**,**"**nutcracker**"**,**"**poinsettia**"**,**"**frosty**"**,**"**yuletide**"**  
 39 **"**elf**"**};   
 40 string word = wordList[choose];  
 41 char gameWrd[word.length()];  
 42 for(int i=0; i < word.length(); i++){  
 43 gameWrd[i] = word[i];   
 44 }  
 45   
 46 //Reset game values for replay  
 47 win=0;  
 48 guesses=0;  
 49 intro();  
 50 char guess[word.length()-1];  
 51 for(int i=0; i <= word.length(); i++){  
 52 guess[i] = '\_';   
 53 }  
 54   
 55   
 56 //Main Game Loop  
 57 do{   
 58 //Output the hangman graphic  
 59 gallows(guesses);  
 60 cout<<endl;  
 61   
 62 //Output the solved/unsolved letters  
 63 for(int i=0;i<word.length();i++){  
 64 cout<<guess[i]<<**"** **"**;  
 65 }  
 66   
 67 //Input user choice for letter  
 68 if(guesses<6){  
 69 cout<<**"** <--- Enter your letter choice: **"**;  
 70 cin>>letter;  
 71 }  
 72 //Assume their guess was wrong   
 73 guesses++;   
 74 let = 0;  
 75   
 76 //Check for letters and replace then in the guess array if correct  
 77 for(int i=0;i<=word.length();i++){  
 78 if(letterCheck(gameWrd[i],letter)){  
 79 guess[i]=letter;  
 80 let=1;  
 81 }  
 82 }  
 83   
 84 //If any of the letters were solved give them an extra guess  
 85 if(let){  
 86 guesses--;  
 87 }  
 88   
 89 //If none of the letters in guess are '\_' then the puzzle has been won  
 90 win=0;  
 91 for(int i=0; i < word.length(); i++){  
 92 if(guess[i]!='\_'){  
 93 win++;  
 94 }   
 95 }  
 96 }while(guesses<=6&&win<word.length());//End Main Game Loop  
 97   
 98 //Check for win/lose  
 99 if(win>=word.length()){  
100 cout<<word<<**"\n**YOU WIN!!**"**;  
101 }   
102 else{  
103 cout<<**"\n**Better luck next time friend, we were looking for: **"**  
104 <<word;  
105 }  
106   
107 cout<<**"\n**Would you like to play again [y/n]? **"**;  
108 cin>>repeat;  
109   
110 }while(repeat!='n');  
111   
112   
113 return 0;  
114 }  
115 void **gallows**(short g){  
116   
117 if(g==6){  
118 cout<<**"**\_\_\_\_\_\_\_\_\_\_\_\_ **\n"**  
119 <<**"**| 0 **\n"**  
120 <<**"**| /|**\\** **\n"**  
121 <<**"**|\_\_\_\_\_\_\_ / **\\** **\n"**  
122 <<**"**|\_\_\_\_\_\_| **\n"**   
123 <<**"**YOU HAVE BEEN HANGED!!! **\n"**;  
124 }  
125 else if(g==5){  
126 cout<<**"**\_\_\_\_\_\_\_\_\_\_\_\_ **\n"**  
127 <<**"**| 0 **\n"**  
128 <<**"**| /|**\\** **\n"**  
129 <<**"**|\_\_\_\_\_\_\_ / **\n"**  
130 <<**"**|\_\_\_\_\_\_| **\n"**;   
131 }  
132 else if(g==4){  
133 cout<<**"**\_\_\_\_\_\_\_\_\_\_\_\_ **\n"**  
134 <<**"**| 0 **\n"**  
135 <<**"**| /|**\\** **\n"**  
136 <<**"**|\_\_\_\_\_\_\_ **\n"**  
137 <<**"**|\_\_\_\_\_\_| **\n"**;   
138 }  
139 else if(g==3){  
140 cout<<**"**\_\_\_\_\_\_\_\_\_\_\_\_ **\n"**  
141 <<**"**| 0 **\n"**  
142 <<**"**| /| **\n"**  
143 <<**"**|\_\_\_\_\_\_\_ **\n"**  
144 <<**"**|\_\_\_\_\_\_| **\n"**;   
145 }  
146 else if(g==2){  
147 cout<<**"**\_\_\_\_\_\_\_\_\_\_\_\_ **\n"**  
148 <<**"**| 0 **\n"**  
149 <<**"**| / **\n"**  
150 <<**"**|\_\_\_\_\_\_\_ **\n"**  
151 <<**"**|\_\_\_\_\_\_| **\n"**;   
152 }  
153 else if(g==1){  
154 cout<<**"**\_\_\_\_\_\_\_\_\_\_\_\_ **\n"**  
155 <<**"**| 0 **\n"**  
156 <<**"**| **\n"**  
157 <<**"**|\_\_\_\_\_\_\_ **\n"**  
158 <<**"**|\_\_\_\_\_\_| **\n"**;   
159 }  
160 else if(g==0){  
161 cout<<**"**\_\_\_\_\_\_\_\_\_\_\_\_ **\n"**  
162 <<**"**| **\n"**  
163 <<**"**| **\n"**  
164 <<**"**|\_\_\_\_\_\_\_ **\n"**  
165 <<**"**|\_\_\_\_\_\_| **\n"**;   
166 }  
167 }  
168 bool **letterCheck**(char word, char letter){  
169 if(word==letter){  
170 return 1;  
171 }  
172 else{  
173 return 0;  
174 }  
175   
176 }  
177 void **intro**(){  
178 cout<<**"\n**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***\n"**  
179 <<**"**\*\*\*\*\*\* Let's Play Hangman, Christmas Style \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***\n"**  
180 <<**"**\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\***\n"**  
181 <<**"**I'm going to **\"**think**\"** of a Christmas word and tell you only the**\n"**  
182 <<**"**number of letters in that word. You will then guess letters that**\n"**  
183 <<**"**will be revealed if they are in the word. If you guess six letters**\n"**  
184 <<**"**that aren't in the word then you will be hanged.**\n\n"**  
185 <<**"**Well I've got my word ready just press Enter when you're ready to play.**"**;  
186 cin.ignore();  
187 cin.get();  
188 }  
189   
190