# Project 1 v2

**<The Hangman>**

**CASS: CSC 5**

**NAME: Eduardo Rangel**

**DATE: 5/28/2015**

## INTRODUCTION:

Title: The Hangman

This is a word guessing game where you have to guess the word. You have 5 chances to guess the word by entering a letter each time you enter a letter that is not in the word you will lose a chance, if you miss waste the 5 chances you have to guess the word you lose the game and you can play it again if you lose the game that means you let the person to be hanged☹.

## Summary:

Number of lines: 301

Number of variables: 16

I tried to use all of the information that I have learned from chapter 1 through chapter 7 those include arrays, strings, functions, loops and so on. The game is just a basic game I still need to implement more things but I think that while I learned more I will be able to make more implementations to the game.

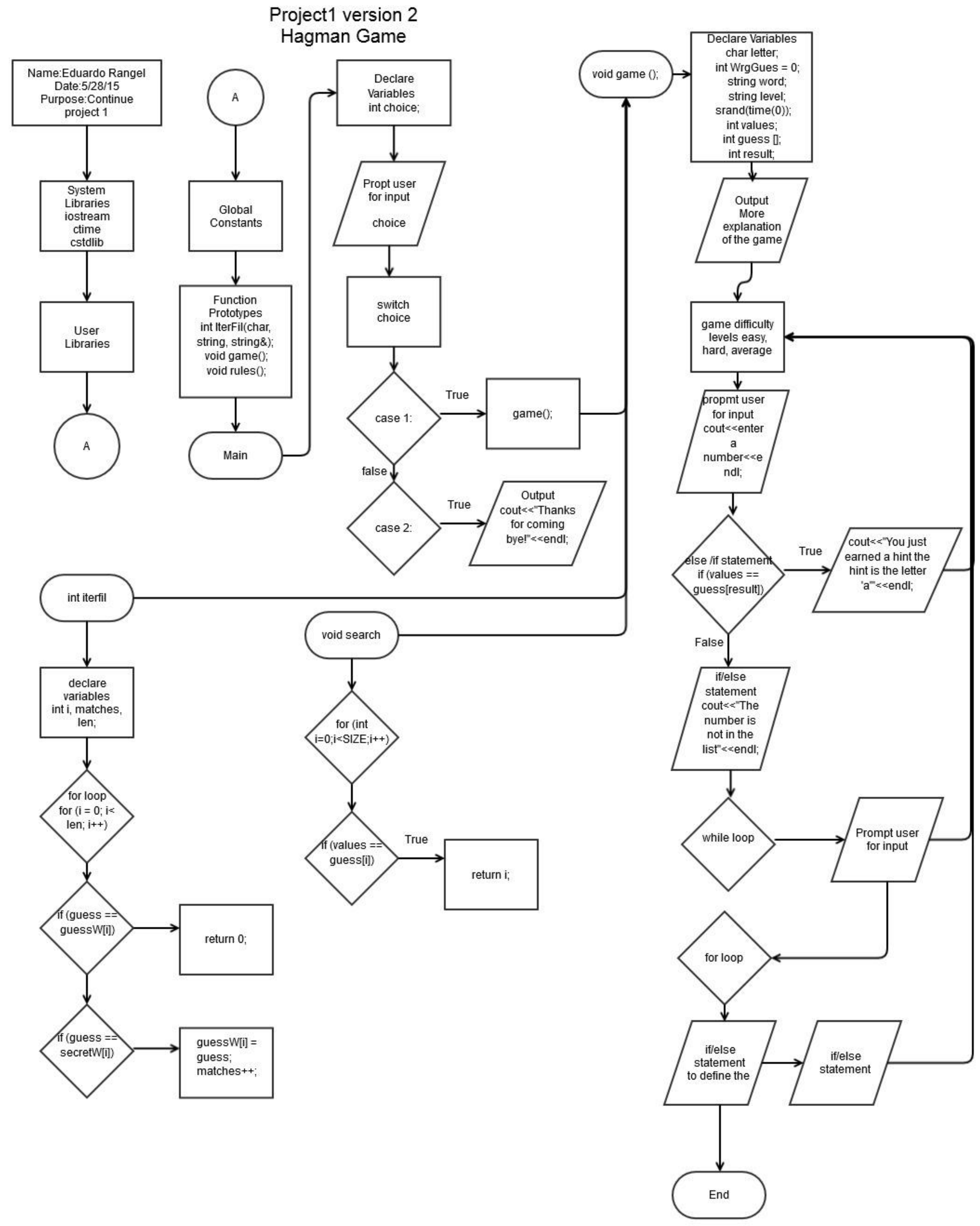
It took me around a week and a half to finish the game since I had to do a lot of thinking on what features I could implement into the game.

I went through the internet to look for some information about the game and some problems that I had to make some features in the game. But mostly I used the “string” library which we haven’t used that much in class.

## Description:

The main point of this game is to guess the word which is random I used an array to set difficulty levels easy, average, and hard. Depending on the difficulty level is the length of the word I also made the array to hold certain names of countries that will appear random on each difficulty level.

## Flowchart


Project1 version 2
Hagman Game
Name:Eduardo Rangel
Date:5/28/15
Purpose:Continue project 1
System Libraries iostream ctime cstdlib
User Libraries
A
A
Global Constants
Function Prototypes int IterFil(char, string, string&); void game(); void rules();
Main
Declare Variables int choice;
Propt user for input choice
switch choice
case 1:
True
game();
false
case 2:
True
Output cout<<"Thanks for coming bye!"<<endl;
void game ();
Declare Variables char letter; int WrgGues = 0; string word; string level; srand(time(0)); int values; int guess []; int result;
Output More explanation of the game
game difficulty levels easy, hard, average
propmt user for input cout<<enter a number<<endl;
else /if statement if (values == guess[result])
True
cout<<"You just earned a hint the hint is the letter 'a'"<<endl;
False
if/else statement cout<<"The number is not in the list"<<endl;
while loop
Prompt user for input
for loop
if/else statement to define the
if/else statement
End
int iterfil
declare variables int i, matches, len;
for loop for (i = 0; i< len; i++)
if (guess == guessW[i])
return 0;
if (guess == secretW[i])
guessW[i] = guess; matches++;
void search
for (int i=0;i<SIZE;i++)
if (values == guess[i])
True
return i;

## Pseudo Code:

```
System Libarries
User Libraries
Global Constants
Function Prototypes
Execution Begins Here
           Declare variables
           Number of maximum tries
          Number of wrong guesses
        Set Random numerator seed
        Ask user for Difficulty level

        Compare the level
   If the user chooses easy
Put all the string inside the array
Generate Random Words using the random numerator seed
Call the function here for guessing the word

Loop until all the guesses are used up user has up to 5 tries
         Fill secret word with letter if the guess is correct
        Increment the number of wrong guesses
       Tell user how many guesses has left.
```

```
        Check if user guessed the word.
        If the user uses all 5 chances without guessing the word output
        the result to the user.

Use the average level with more letters than the easy level.
        Put all the string inside the array here.
        Call the function here for guessing game.
        Initialize the secret word with the * character.
        Generate Random Words using the random numerator seed
Call the function here for guessing the word

Loop until all the guesses are used up user has up to 5 tries
        Fill secret word with letter if the guess is correct
        Increment the number of wrong guesses
        Tell user how many guesses has left.

The hard level is supposed to have words with more letters than the
easy and average levels
        Put all the string inside the array here.
        Call the function here for guessing game.
        Initialize the secret word with the * character.
        Generate Random Words using the random numerator seed
```

```
Call the function here for guessing the word

Loop until all the guesses are used up user has up to 5 tries
    Fill secret word with letter if the guess is correct
    Increment the number of wrong guesses
    Tell user how many guesses has left.

Function to calculate the letters we already used
    Identify if a word has already been used in another guess
    Is the guess in the secret word?
```

## Variables:

| **Type** | **Variable Name** | **Description** | **Location** |
|---|---|---|---|
| **const int** | maxTrys | Sets the maximum number of tries a player has. | int main (int argc, char** argv) |
| **string** | word | Is used as the reference to generate a random word when playing the game | int main (int argc, char** argv) |
| | Average[] | Array that contains the names of the countries for the average level | Void game(); |
| | hard[] | Array that contains the names of the countries in the hard difficulty level. | Void game(); |
| | unknown | Sets the word length and uses the “ * “ to hide word. | Void game(); |
| | easy[] | Array that contains the 4 names in the easy level of difficulty | Void game(); |
| | level | Is used to set the level of the game for example easy, average, hard. | int main (int argc, char** argv) |
| **char** | letter | Is used to hold the letter that player inputs to guess the word. | Void game (); |

| | | | |
|---|---|---|---|
| **int** | wrgGues = 0; | keeps track of the number of guesses the player has left to guess the word or how many guesses the player has used. | void game(); |
| | n | Is set to carry the value of the random generator into the array of each difficulty level to generate random words that are into the array. | Void game(); |
| | I | Goes inside of the for loop. | int lterFil(char guess, string secretW, string &guessW) |
| | Matches = 0; | It takes the number of matches the player has played and increments the number of guesses. | lterFil(char guess, string secretW, string &guessW) |
| | len | Sets the length of the secret word | lterFil(char guess, string secretW, string &guessW) |
| | guess | Array used to make the linear search it holds 10 numbers | Void game(); |
| | Values | It stores the number entered by the user and cmpares it to the linear search numbers | Void game(); |

| | | | |
|---|---|---|---|
| | result | This variable holds the linear search functions called in the void game function to compare the number entered by user with numbers in the array | Void game(); |

## Key concepts of each chapter

| **Chapter** | **Key Words** | **Library** |
|---|---|---|
| 2 | Input and output “cout”. Variales int, char, etc. Mathematical operators | <iostream> |
| 3 | Introduces the “cin” object to read data entered. Assign values to a variable. Type casting.<br>Cin.get, cin.ignore<br>Mathematical libraries | <iostream> |
| | Formatting output.<br>Setw( );<br>Cout<<Fixed<<setprecision()<<showpoint; | <iomanip> |
| | strings | <string> |
| | Random number generator<br>Srand; | <ctime> |
| 4 | Relational Operators<br>The if Statement if(statement) { }<br>The if/else Statement: if (statement){<br>}else<br>Nested if Statements: if (statement){<br>If(statement){ }<br>}<br>Flags | <iostream> |

| | | |
|---|---|---|
| | Logical Operators: AND &, OR \|\|, NOT !. | |
| 5 | Loops and files<br>LOOPS: while (statement){ }, for (int i=0;i< 3;i++), do-While: do{ statement }while (statement);<br>Nested loops: for (int i=0;i <=3;i++){<br>For(int j=0;j<=5;i++){ }<br>} | <iostream> |
| | Files ofstream:<br>ofstream.open("name.dat")<br>Out.close; | <fstream> |
| 6 | Functions: Function Prototypes: void: void function();<br>Calling a functionin the main: inr main(){function();}<br>Function & Return statement: void function(statement){ all the statements return something;<br>} | |
| 7 | Arrays: int array[ number of arrays ]; { arrays start from 0};<br>For loops with arrays: for (int i=0;I <5;i++){<br>Cout<<array[i]<<endl; | <iostream> |
| 8 | Searching and sorting arrays: int index = 0; 42 int position = −1; 43 bool found = false; 44<br>45 while (index < numElems && !found)<br>46 {<br>47 if (list[index] == value) 48 {<br>49 found = true; 50 position = index; 51 } | |

| | | |
|---|---|---|
| | 52 index++;<br>53 }<br>54 return position; | |
| | Vector: vector<int> lines<br>lines.push_back(12345);<br>lines.push_back(12345678);<br>lines.push_back(17845);<br>lines.push_back(110564); | |
| 9 | Pointers: int *pointer;<br>Int pointer(int *p){<br>Cout<<*p<<ednl;<br>} | |

**CODE:**

```
/*
 * File:   main.cpp
 * Author: EDUARDO
 *
 * Created on June 7, 2015, 2:46 PM
 */
//System libraries
#include <iostream>
#include <cstdlib>
#include <ctime>
#include <string>
using namespace std;
//User libaries

//Global Constants

//Function Prototypes
int lterFil(char, string, string&);
void game();
void rules();
int search(int guess[], int, int);
```

```
//Execution Begins here
int main(int argc, char** argv)
{
   //Declare variables
   int choice;
   //Call the rules function to display the rules before the player starts
playing the game
   rules();
   //Prompt user for input
   cout<<"Do you want to continue to play the game if yes enter 1 if
not enter number 2"<<endl;
   cin>>choice;
   //Use switch statement to let the player decide whether he wants
to continue to play the game or not
   switch(choice){
      case 1: game();
      break;
      case 2:
         cout<<"Thanks for coming bye!"<<endl;
         break;
      default:;
   }
   return 0;
   }


void rules()
{
   //Output the rules for the player
  cout<<"                    Hangman Game"<<endl;
  cout<<"Welcome to Hangman here are the rules to be successful in the game."<<endl;
  cout<<"                       Rules "<<endl;
  cout<<"--------------------------------------------------------------"<<endl;
  cout<<"1. You have to Guess the country that is behind the word."<<endl;
  cout<<"2. You have to enter a letter until you guess the whole word press enter after you enter a letter."<<endl;
  cout<<"3. You have 5 chances to guess the word, you will lose a chance each time you enter a letter that is not in the word."<<endl;
  cout<<"4. If you miss 5 times the game will be over."<<endl;
  cout<<"5. If you guess the word correctly before wasting your 5 chances you win the game."<<endl;
  cout<<"--------------------------------------------------------------"<<endl;
  cout<<endl;


}
```

```
int search(int guess [], int SIZE, int values)
{
   for (int i=0;i<SIZE;i++)
   {
      if (values == guess[i])
      {
         return i;
      }
   }
   return -1;


}
//Function to calculate the letters we already used
int lterFil(char guess, string secretW, string &guessW)
   {
   int i;
   int matches = 0;
   int len = secretW.length();
   for (i = 0; i< len; i++)
   {
         // Identify if a word has already been used in another guess
```

```
        if (guess == guessW[i])
            return 0;
        // Is the guess in the secret word?
        if (guess == secretW[i])
        {
            guessW[i] = guess;
            matches++;
        }
    }
    return matches;
    }

//Game function to make all the procedure of the game
void game()
{
    //Declare variables
     const int  maxTrys = 5;//number of maximum tries
    //string name;
    char letter;
    int WrgGues = 0;//Number of wrong guesses
    string word;
    string level;
```

```
srand(time(0)); //Set random numerator seed



int guess [] = {10, 1, 3, 67, 58, 5, 9, 28, 7, 47};


int values;

int result;


// Ask user for for Easy, Average, Hard

cout << "\nChoose a LEVEL(E - Easy, A - Average, H - Hard):" << endl;

cin >> level;

// compare level

if (level == "Easy" || level == "easy")

{

result = search(guess, 10, values);

	//put all the string inside the array here

	string easy[] = { "india", "japan", "nepal", "china" };

	string word;

	//Generate Random words into the array

	int n = rand() % 4;

	word = easy[n];
```

```
        //call the function here for guessing game
        // Initialize the secret word with the * character.
        string unknown(word.length(), '*');
        cout << "\nEach letter is represented by an asterisk.";
        cout << "\nYou have to type only one letter in one try.";
        cout << "\nYou have " << maxTrys << " tries to try and guess
the country.";
        cout <<
"\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"<<endl;


        cout<<"You have the opportunity to earn a hint of the word
you just need to guess the number. Enter a number. "<<endl;
        cin>>values;


        if (values == guess[result])
        {
          cout<<"The number is not in the list"<<endl;
        }
        else{
          cout<<"You just earned a hint the hint is the letter 'a'"<<endl;
        }
        // Loop until the guesses are used up
        while (WrgGues < maxTrys)
```

```
    {
        cout << "\n" << unknown;
        cout << "\nGuess a letter: ";
        cin >> letter;
        // Fill secret word with letter if the guess is correct,
        // increment the number of wrong guesses.
        if (lterFil(letter, word, unknown) == 0)
        {
            cout << endl << "Whoops! That letter isn't in there!" <<
endl;
            WrgGues++;
        }
        else
        {
            cout << endl << "You found a letter! Isn't that
exciting?" << endl;
        }
        // Tell user how many guesses has left.
        cout << "You have " << maxTrys - WrgGues;
        cout << " guesses left." << endl;
        // Check if user guessed the word.
        if (word == unknown)
        {
            cout << word << endl;

            cout << "Yeah! You got it!";

            break;

        }

    }

    //If the user uses all 5 chances without guessing the word
output the result to the user

    if (WrgGues == maxTrys)

    {

        cout << "\nSorry, you lose...you've been hanged." << endl;

        cout << "The word was : " << word << endl;

    }

}

//Use the average level with more letters than the easy level

else if (level == "Average" || level == "average")

{

    //put all the string inside the array here

    string average[] = { "madagascar", "azerbaijan", "kyrgyzstan" };


    int n = rand() % 3;

    word = average[n];
    //call the function here for guessing game
    // Initialize the secret word with the * character.
    string unknown(word.length(), '*');
    cout << "\nEach letter is represented by an asterisk.";
    cout << "\nYou have to type only one letter in one try.";
    cout << "\nYou have " << maxTrys << " tries to try and guess
the country.";
    cout <<
"\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"<<endl;


    cout<<"You have the opportunity to earn a hint of the word
you just need to guess the number. Enter a number. "<<endl;
    cin>>values;


    if (values == guess[result])
    {
      cout<<"The number is not in the list"<<endl;
    }
    else{
      cout<<"You just earned a hint the hint is the letter 'a'"<<endl;
    }
    // Loop until the guesses are used up
    while (WrgGues < maxTrys)
```

```
	{
		cout << "\n" << unknown;
		cout << "\nGuess a letter: ";
		cin >> letter;
		// Fill secret word with letter if the guess is correct,
		// otherwise increment the number of wrong guesses.
		if (lterFil(letter, word, unknown) == 0)
		{
			cout << endl << "Whoops! That letter isn't in there!" << 
endl;
			WrgGues++;
		}
		else
		{
			cout << endl << "You found a letter! Isn't that 
exciting?" << endl;
		}
		// Tell user how many guesses has left.
		cout << "You have " << maxTrys - WrgGues;
		cout << " guesses left." << endl;
		// Check if user guessed the word.
		if (word == unknown)
		{
```

```
                cout << word << endl;
                cout << "Yeah! You got it!";
                break;
            }
        }
        if (WrgGues == maxTrys)
        {
            cout << "\nSorry, you lose...you've been hanged." << endl;
            cout << "The word was : " << word << endl;
        }
    }
    //The hard level is supposed to have words with more letters than
the easy and average levels
    else if (level == "Hard" || level == "hard")
    {
        //put all the string inside the array here
        string hard[] = { "turkmenistan", "yugoslav", "uzbekistan" };

        int n = rand() % 3;
        word = hard[n];

        //call the function here for guessing game
        // Initialize the secret word with the * character.
        string unknown(word.length(), '*');
        cout << "\nEach letter is represented by an asterisk.";
        cout << "\nYou have to type only one letter in one try.";
        cout << "\nYou have " << maxTrys << " tries to try and guess
the country.";
        cout <<
"\n~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~"<<endl;


        cout<<"You have the opportunity to earn a hint of the word
you just need to guess the number. Enter a number. "<<endl;
        cin>>values;


        if (values == guess[result])
        {
          cout<<"The number is not in the list"<<endl;
        }
        else{
          cout<<"You just earned a hint the hint is the letter 'a'"<<endl;
        }
        // Loop until the guesses are used up
        while (WrgGues < maxTrys)
        {
```

```
		cout << "\n" << unknown;
		cout << "\nGuess a letter: ";
		cin >> letter;
		// Fill secret word with letter if the guess is correct,
		// otherwise increment the number of wrong guesses.
		if (lterFil(letter, word, unknown) == 0)
		{
			cout << endl << "Whoops! That letter isn't in there!" <<
endl;
			WrgGues++;
		}
		else
		{
			cout << endl << "You found a letter! Isn't that
exciting?" << endl;
		}
		// Tell user how many guesses has left.
		cout << "You have " << maxTrys - WrgGues;
		cout << " guesses left." << endl;
		// Check if user guessed the word.
		if (word == unknown)
		{
			cout << word << endl;
```

```
                cout << "Yeah! You got it!";
                break;
            }
        }
        if (WrgGues == maxTrys)
        {
            cout << "\nSorry, you lose...you've been hanged." << endl;
            cout << "The word was : " << word << endl;
        }
    }
    }
```