Project 2
Hangman Game

Jose Roman
CSC 5
06/08/15

Project 2 documentation and flowcharts. This PDF include the rules of the game, pseudo code of the project. And the code of the program.

Introduction

Hangman is a paper and pencil guessing game that many people play around the world. The games consists of two players or more. Player 1 thinks of a word, phrase or sentence and the player 2 tries to guess it by suggesting letters. Player 1 suggests a letter If the suggested letter is not part of the word then player 2 draws one element of a hanged man stick figure as a tally mark. If player 1 gives a correct letter then player 2 writes down the letter in the correct position that the letter is located in the word. The game is over when player 1 guesses the whole word correctly or when player 2 completes the hanged man stick figure.

The Hangman Game project consists of a player vs. computer game. The computer select a random word from "HangmanGame.txt" so that the player is able to guess the random word. The player then is going to guess the word by pressing the alphabet letters there is not number in the word chosen from the computer. The player is going to have six chances or strikes to guess the word. The player can asks for a hint but the computer is going to deduct five points. The player can only get a hint for each game. Each correct letter correspond from the random chosen word will earn the player five points per letter. But if the player input the incorrect letter then the computer deduct 1 point from each incorrect word. The player wins until he guess the word. The computer wins until the player is unable to resolve the game in seven strikes (chances). The final score is put in a file "game.txt" inputting if the player won or lost and their final score. The computer display the rules to the player so the player is able to play the game. For every wrong letter than the computer draw a part of the hangman figure until it completes a hangman figure. The player is able to end the game by pressing the '#'.

Summary

Project size: 500+ lines
Number of variables: Around 19
Number of methods: 19

I used concepts that were covered in the course. I implemented: if statements, if-else statements, switch statements, array, for-loops, while-loops, and enumeration types, and function prototypes using void. It also includes 2D arrays, structures, vectors, formatted output and defaulted parameters.

The computers reads a random word from the "HangmanGame.txt" file containing a list of many words. When the games end, there is a file created with the random word and the player's results: the file record if the player won or lost, the word, and their final score. This file are very helpful because the game needs the text document to function. And the file that records the result help the player to keep track of their score. The final project it took many days. I used cplusplus website, YouTube videos, and the textbook and some other websites to implement a Hangman Game. I added a loop where the computer asked the player if they wanted to play again the game because I found some examples in the Internet. A lab aid from the lab helped me to improve the code. The project is using iostream, string, fstream, vector, iomanip and cstdlib

Major Variables

Type	Variable Name	Description	Location
Char []	WORD	String that the computer chose at random	Main.cpp (Global Variable)
GamerGame	game	A structure containing commonly used variables for the game.	main.cpp
Gamer	gamer	a structure used like a database for the gamer.	Main.cpp
int	wLength	The length of the random word.	Main.cpp

C++ Constructs

Chapter	New syntax and keywords	Location
2	cout	rGame(GamerGame *,Gamer *);
	cin	rGame(GamerGame *,Gamer *);
	Output formatting	oFile(Gamer *, ofstream &);
	int	main(int argc, char** argv)
	char	main(int argc, char** argv)
	bool	rGame(GamerGame *,Gamer *);
	string	dHangman(GamerGame *, Gamer *);
	assignment operator (+=)	dHangman(GamerGame *, Gamer *);
	arithmetic operator (+,-,*,/)	fLetter(GamerGame *,Gamer *, char);
	increment operator (++)	fLetter(GamerGame *,Gamer *, char);
	decrement operator(--)	fLetter(GamerGame *,Gamer *, char);

Chapter	New syntax and keywords	Location
3	enumerator type	main.cpp(Global Variable)
	if-else statements	fLetter(GamerGame *,Gamer *, char);
	switch statements	Hint(Gamer *, int)
	break	Hint(Gamer *, int)
	while loops	fLetter(GamerGame *,Gamer *, char);
	do while loops	rGame(GamerGame *,Gamer *);
	for loops	search(char [],char,int)
4	function	main.cpp
	pass by value	display(char [],char,int)
	global variable	main.cpp
	overloaded functions	-fLetter(char [],char); -fLetter(GamerGame *,Gamer *, char);
5	void functions	main.cpp
	returning primitive data type	fLetter(char [], char);
	procedural abstraction	rGame(GamerGame *,Gamer *);
6	ifstream	oFile(Gamer *, ofstream &);
	ofstream	oFile(Gamer *, ofstream &);
	defaulted arguments	search(char [],char,int)
7	arrays	struct GamerGame.h
	passing array between functions	search(char [],char,int)

Chapter	New syntax and keywords	Location
8	multi-dimensional array	struct GamerGame.h
	searching an array	search(char [],char,int)
	sorting an array	pSort(Gamer *)
	string array []	struct GamerGame.h
	vectors	struct Gamer.h
10	structures	struct GamerGame.h, struct Gamer.h

Reference

1. http://www.cplusplus.com
2. YouTube
3. Textbook
4. Lab aid in the computer lab

Pseudo Code

Initialize hangman drawing

Read a file
choose a random word from file

Display the fill in the blanks

display the rules of the game

while the user has yet to completely fill in the blanks
    read: the user's guess
    if the player input is a ?

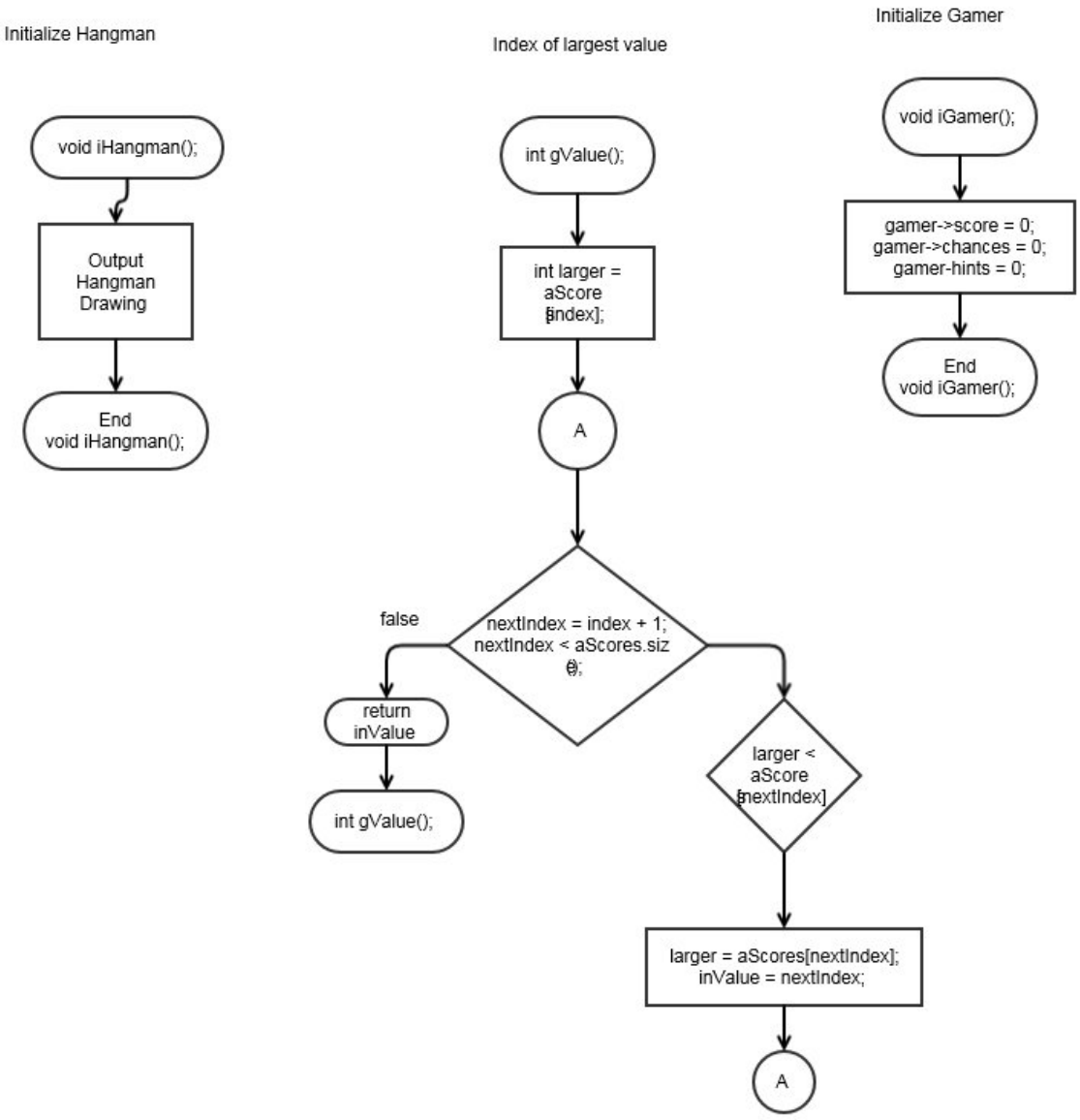
```
                if the number of hints used is equal to zero
                        output: the number of vowels there are in the letter
                        decrement their score by 5
    else
                        output: no more hints
                increment the number of hints used
      else
                if the letter was not found
                        output: incorrect! -1pt
                        decrement their score
                        increment their strike count
       else
                        if the letter was found
                                output: correct! +5 pts.
                                Increment score by 5
                        else
                                output: letter was already found

                        display the fill in the blanks
                        if the word filled out is equal to the word chosen by the computer
                                end the game user won
                        else
                                if strike count is equal to 6
                                        end the game user lost
display the game results
```

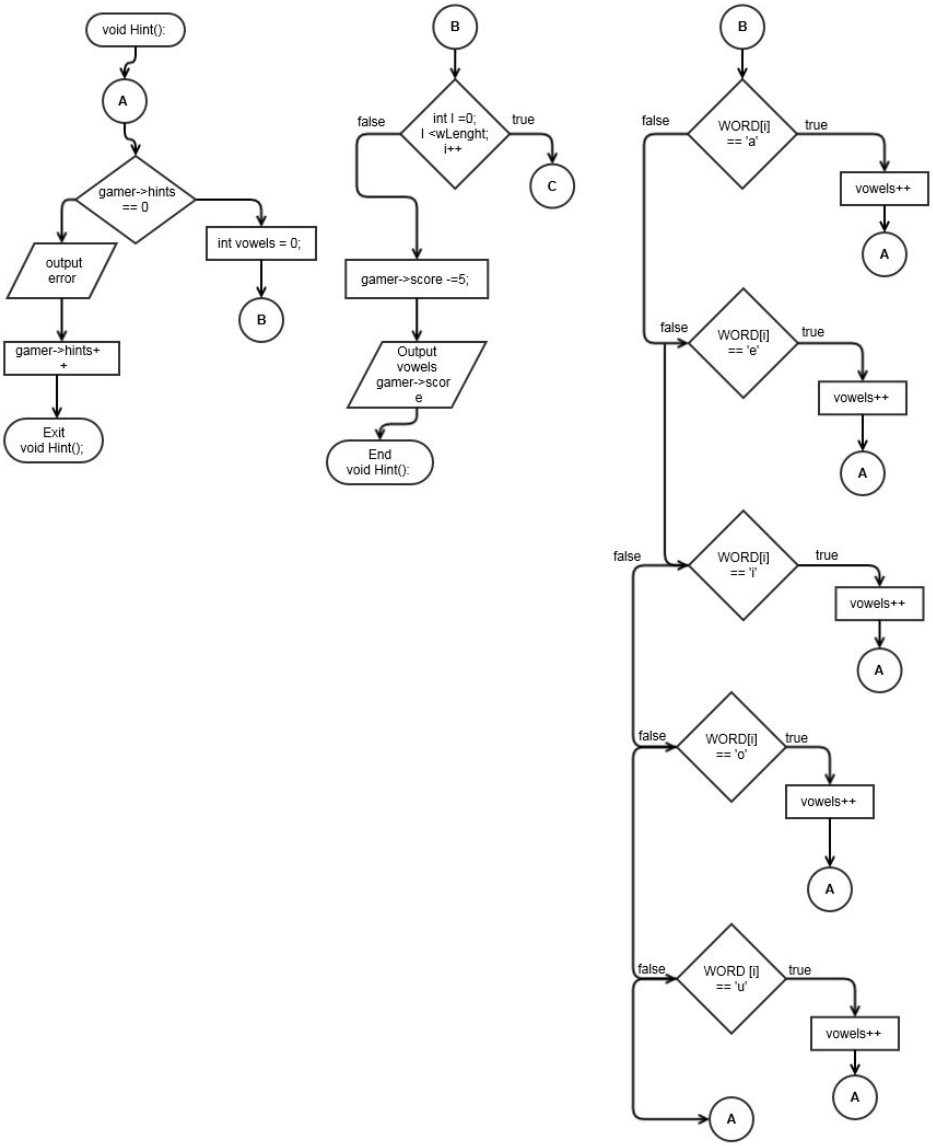
Continue the game until the player no longer wants to play.

Flowcharts

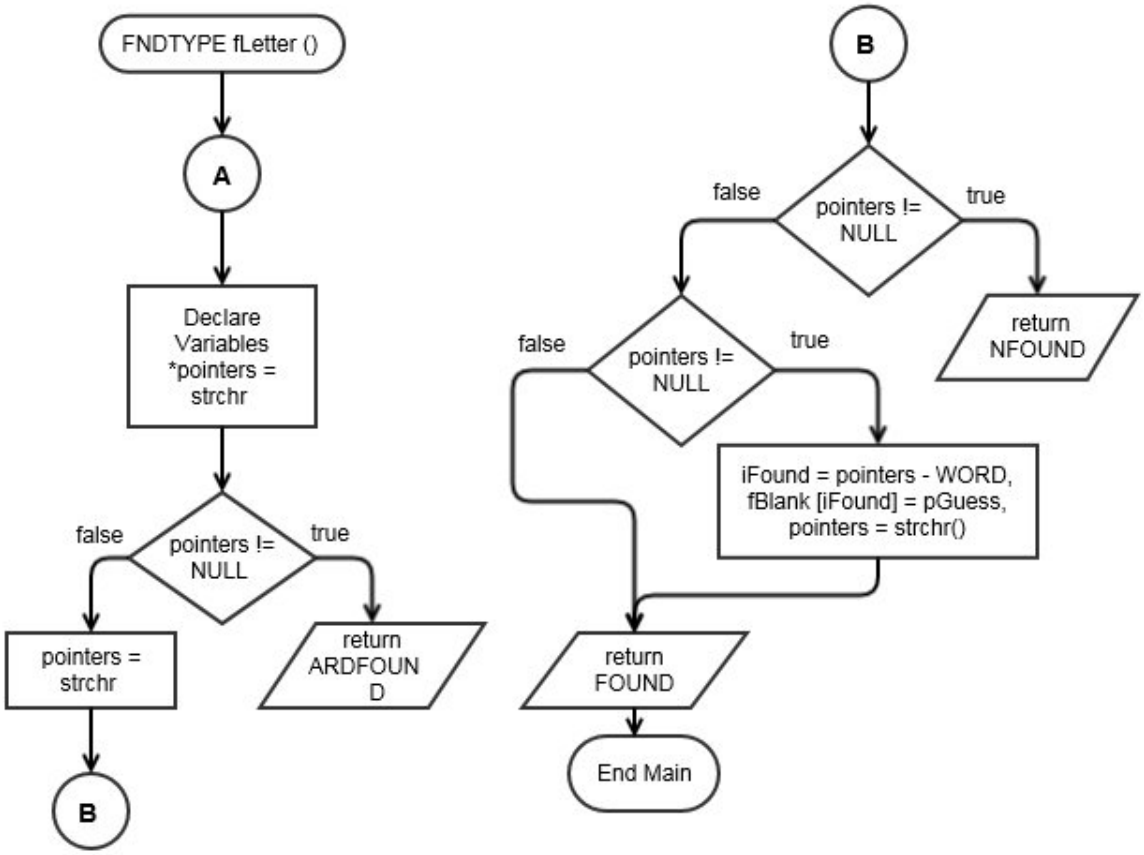
Hangman Game



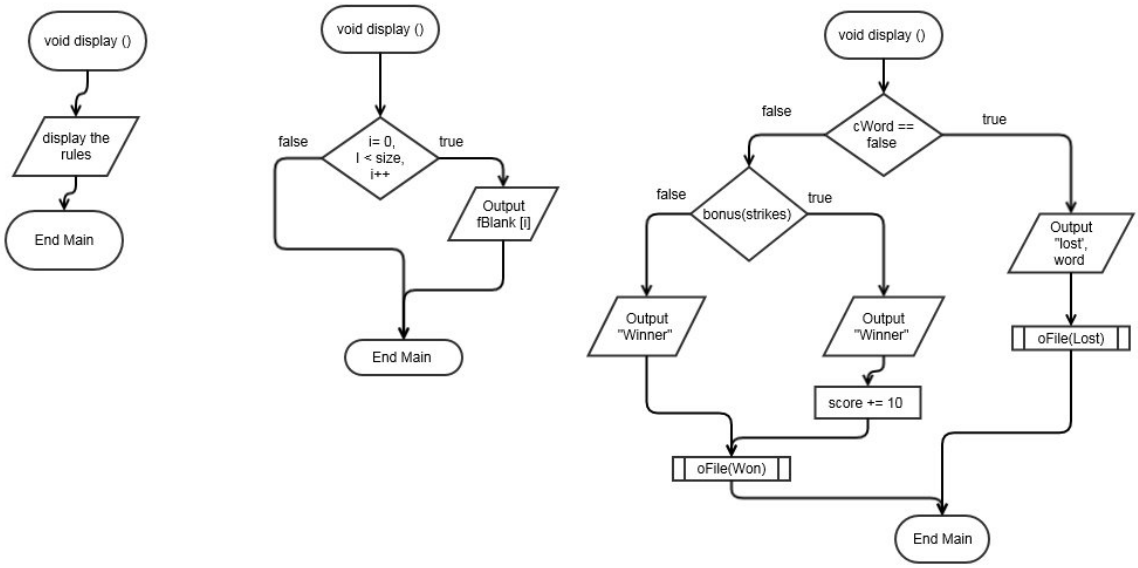
Hint


void Hint():
A
gamer->hints == 0
output error
gamer->hints++
Exit void Hint();
int vowels = 0;
B
B
int I =0; I <wLenght; i++
false
true
C
gamer->score -=5;
Output vowels gamer->score
End void Hint():
B
WORD[i] == 'a'
false
true
vowels++
A
WORD[i] == 'e'
false
true
vowels++
A
WORD[i] == 'i'
false
true
vowels++
A
WORD[i] == 'o'
false
true
vowels++
A
WORD [i] == 'u'
false
true
vowels++
A
A

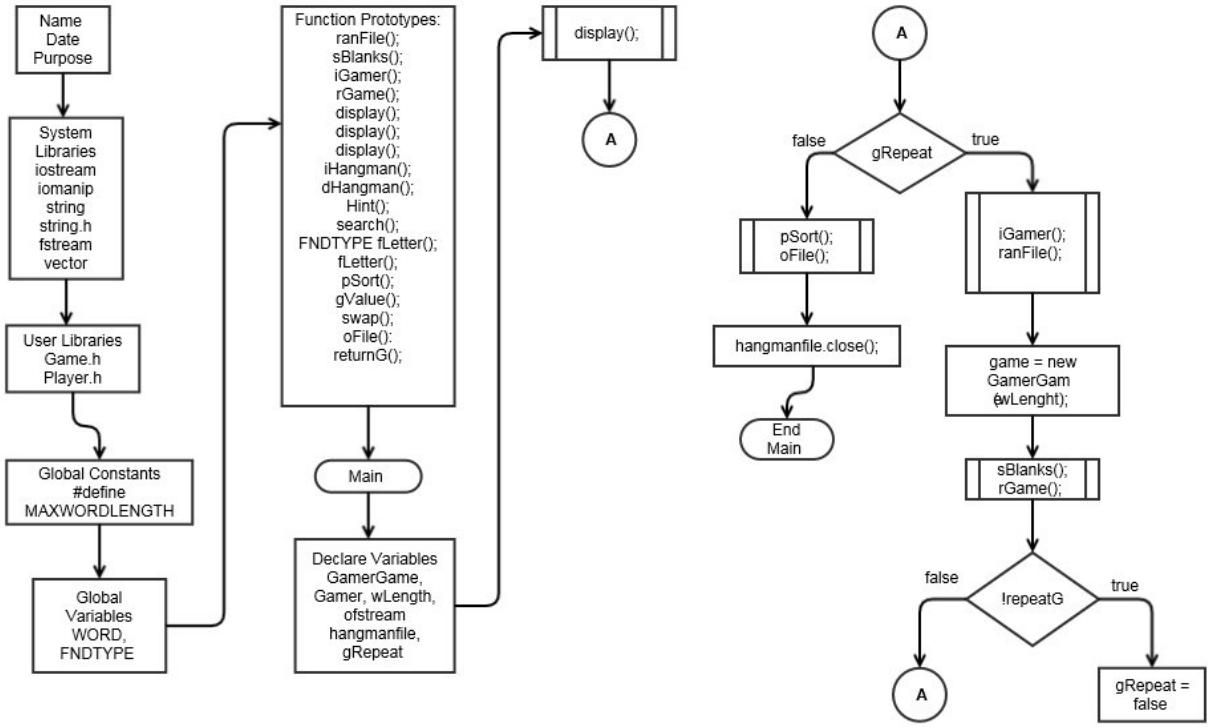
Hangman Game
Letter



Display the Game

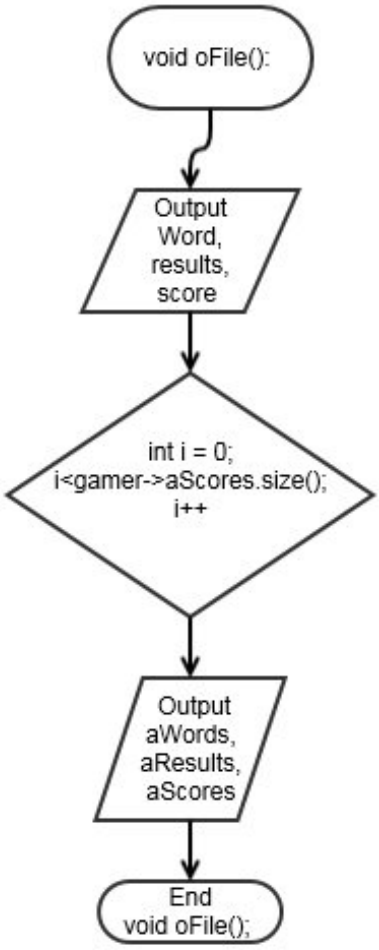


Project 1
Hangman Game

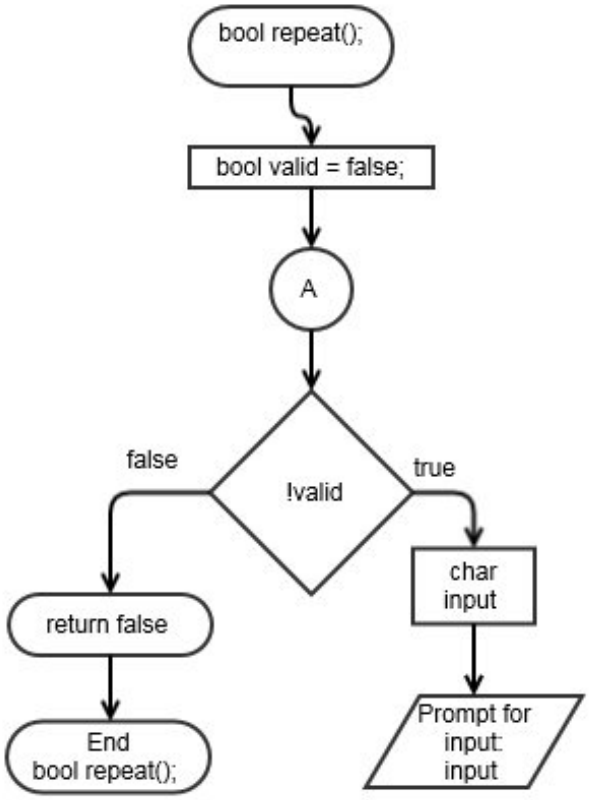


Hangman Game

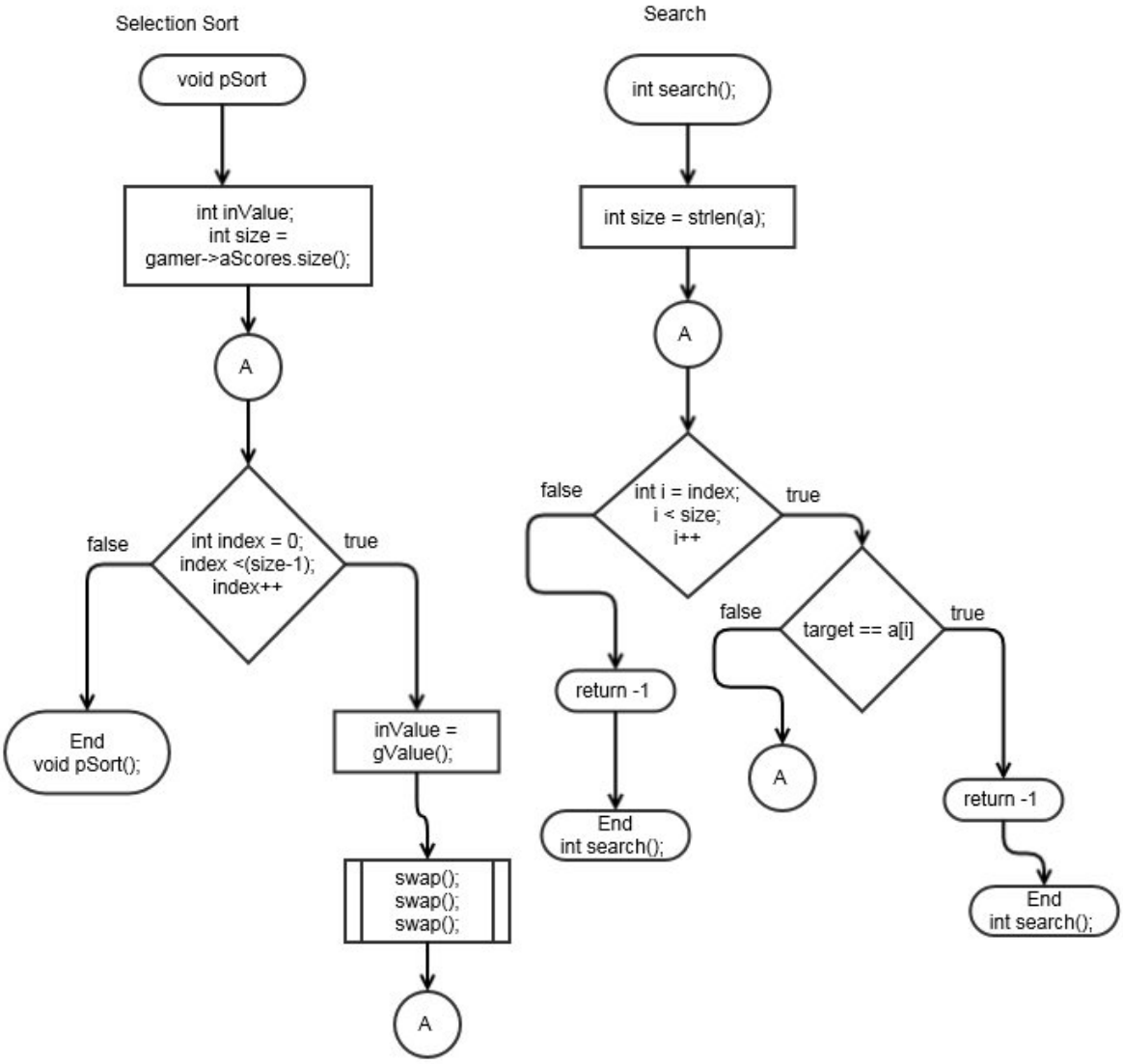
Output to file



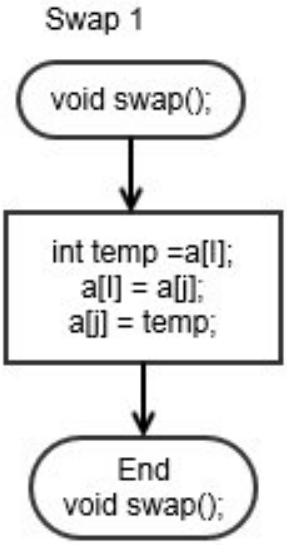
Repeat Game

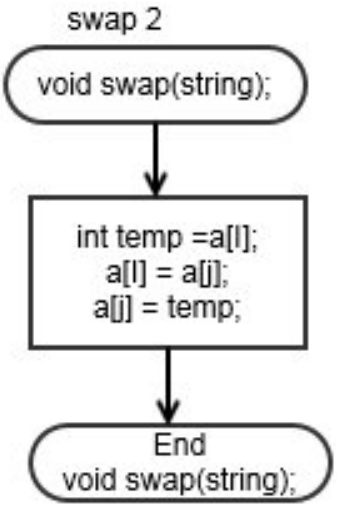


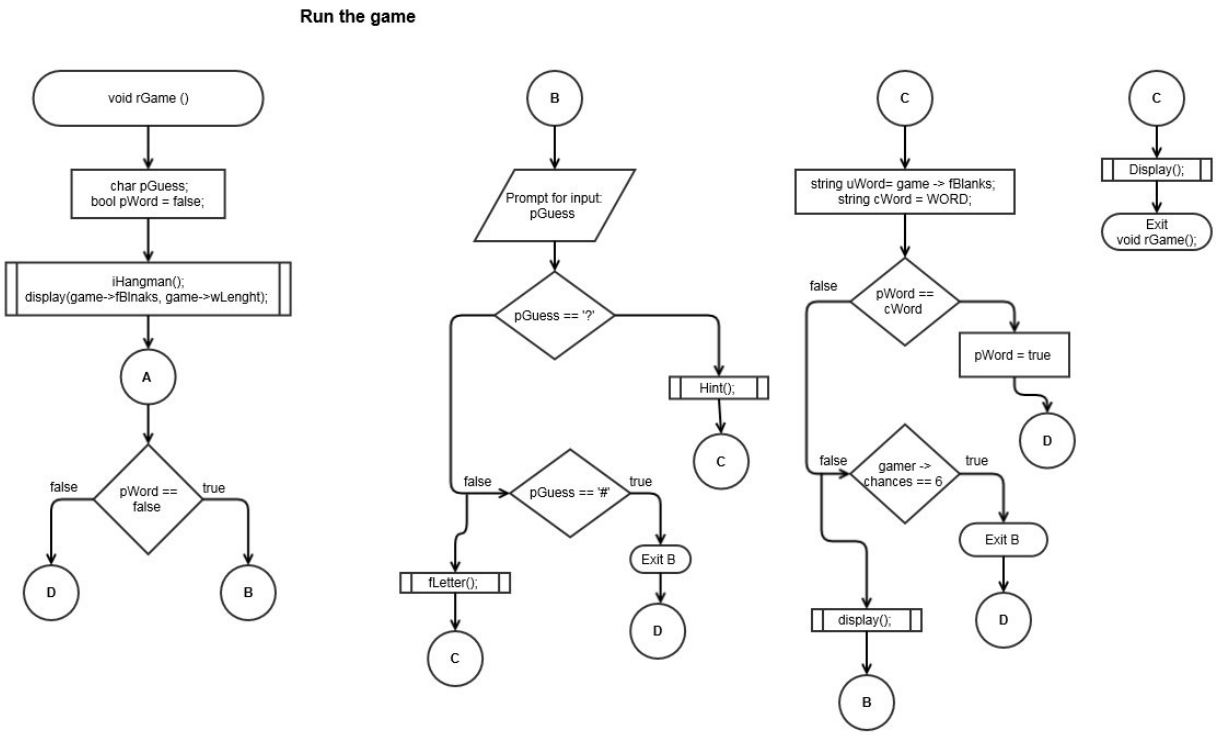
Hangman Game



Hangman Game







Program

```
/*
* File:   main.cpp
* Author: Jose Roman
* Created on May 4, 2015, 10:33 AM
*      Purpose: Project 1: Hangman Game
*/

//System Libraries
#include <iostream>
#include <string.h>
#include <string>
#include <fstream>
#include <cstdlib>
using namespace std;
//User Libraries
```

```
//Global Constants
const char *WORD;
enum FNDTYPE {NFOUND, FOUND, ARDFOUND};// Compare Results
//Function Prototypes
void display();//rules of the games
void rGame(char fBlank[],int,char,int,int,int);
void display(char fBlank[], int);// Fill in the blanks
FNDTYPE fLetter(char,char fBlank[]);//Blanks to be filled
bool bonus(int);//Finish the word in seven chances
void display(bool,int,int);
void oFile(string,int,int);// Output the result in a file
//Execution Begins Here!
int main (int argc, char** argv){
   //Initialize the random seed
   srand (time(NULL));

   //Declare Variables
   char pGuess;//Player Guess
   int score   =0;//Player Score
   int strikes =0;//Guessed Wrong
   int hints   =0;//Hints at the player

   const int MLNIF = 300; //Max lines in file
   string wArray[MLNIF];
   int wCount = 0;//Word Count
   ifstream fin("HangmanWords.txt");//File name for the hangman words
   if (fin.is_open()){

      while(!fin.eof() && wCount < MLNIF){
         getline(fin, wArray[wCount]);
         wCount++;
      }
   }
   else
      cout<<"File was not opened"<<endl;//Input this if file is not found
   //Random Word from file
   int index = rand() % wCount;
   WORD = wArray[index].c_str();
   int wLen = strlen(WORD);

   //Input player guesses in a string
   string space;
   for (int i = 0;i < wLen; i++)
```

```
        space += "_";
    const char *blanks = space.c_str();//Empty string
    char fBlank[wLen];
    strcpy(fBlank,blanks);
    rGame(fBlank,wLen,pGuess,hints,strikes,score);

    return 0;
}

//Rules of the game
void display(){
    cout<<"Welcome to the Hangman Game..."<<endl;
    cout<<"To win the game you need to guess a random word."<<endl;
    cout<<"Rules:"<<endl;
    cout<<"Rule 1. You will only have seven chances to guess the random word"<<endl;
    cout<<"Rule 2. If you guess the correct word before the seven chances you"<<endl;
    cout<<"         you will earn 10 points"<<endl;
    cout<<"Rule 3. You are allowed for only one hint but I will deduct five points."<<endl;
    cout<<"Rule 4. For each correct letter from the random word you earn five points."<<endl;
    cout<<"Rule 5. For each incorrect letter from the random word I will deduct you 1
point."<<endl;
    cout<<"Rule 6. Good Luck!! May the odd be in your favor.."<<endl;
    cout<<endl;

}

void rGame(char fBlank[],int size,char pGuess, int hints,int strikes,int score){
    //Input the rules of the games
    display();
    //Display how many letters in a word
    display(fBlank, size);

    bool cWord = false;//
    while (cWord == false){

        cout<<"Your Guess? ";
        cin>>pGuess;




        if(pGuess == '?'){
            if(hints == 0){
```

```
        int vowels=0;
        for(int i=0;i<size;i++){
            switch(WORD[i]){
                case 'a':{vowels++;break;}// vowels++ means vowels=vowels+1;
                case 'i':{vowels++;break;}
                case 'u':{vowels++;break;}
                case 'e':{vowels++;break;}
                case 'o':{vowels++;break;}
                default:break;
            };
        }
        cout<< vowels <<" vowel(s) is this word."<<endl;
        cout<< "-5 points"<<endl;
        score -=5;
    }
    else
        cout<<"No more hints are available."<<endl;
    hints++;
}
else{
    FNDTYPE result= fLetter(pGuess, fBlank);
    if(result == NFOUND){
        cout<<"Incorrect! -1 point."<<endl;
        score--;
        strikes++;
    }
    else{
        if(result == FOUND){
            cout<<"Correct! +5 points.";
            score += 5;
        }
        else
            cout<<"Letter was already found.";
    }
    cout<<endl<<endl;


//Where the word was filled
display(fBlank, size);

string uWord = fBlank;
string coWord = WORD;
if(uWord == coWord)
```

```
            cWord = true;
        else//
        {
            if(strikes == 7)
                break;
        }
    }

}
//Display results to user
display(cWord, strikes, score);
}

//
void display(char fBlank[],int size){
    //Display the blanks
    for(int i=0;i<size;i++)
        cout<< " " <<fBlank[i];
    cout<<endl;
}
//Letter given by the user
FNDTYPE fLetter(char pGuess,char fBlank[]){
    char *cPter = strchr(fBlank, pGuess);// Pointers
    if(cPter != NULL)
        return ARDFOUND;

    cPter=strchr(WORD,pGuess);
    if(cPter == NULL)
        return NFOUND;

    while(cPter != NULL){
        int iFound= cPter - WORD;
        fBlank[iFound]= pGuess;
        cPter = strchr( cPter + 1,pGuess);
    }
    return FOUND;

    }
//If the player completed the word in seven chances
bool bonus(int strikes){
    if(strikes < 7)
        return true;
    else
```

```
        return false;
}
//Display results and Output Results
void display(bool cWord,int strikes, int score){
    cout<<endl;
    if(cWord == false){
        cout<< "You Lose!";
        cout<< " The word was "<<WORD;
        oFile("Lost", strikes, score);
    }
    else
    {
        if(bonus(strikes)){
            cout<< "Congratulation! You completed the word before 7 strikes! "<<endl;
            cout<< "+10 points!";
            score+=10;

    }
    else
    cout<< "You have completed the game! Congratulations!";
oFile("Won",strikes,score);
}
}
void oFile(string results,int strikes,int score){
    ofstream myfile;
    myfile.open("game.txt");
    myfile<<"You "<<results<<" the game!"<<endl;
    myfile<<"The word was " <<WORD      <<endl;
    myfile << "You used up "   << strikes       << " strikes" << endl;
    myfile << "Your score = "  << score         << endl;
    myfile.close();
    // Tell User their result was outputted to a file
    cout << endl;
    cout << "Your score was printed to a file";
    cout << "...Go check out your score!" << endl;

}
```

Gamer.H

```
 /*
 * File:   Gamer.h
 * Author: JR
```

```
 *
 * Created on June 1, 2015, 11:37 PM
 */
#ifndef GAMER_H
#define GAMER_H
#include <vector>
#include <string>
#include <string.h>
struct Gamer
{
   // Initializes variables
   Gamer(): score(0), chances(0), hints(0), cResult("") {}
   // Current Game Variables
   int              score;   // User's score
   int              chances;       // User's strikes
   int              hints;         // User's number of hints used
   std::string        cResult;  // Current Results
   // All Game Variables
   std::vector<int>    aScores; //User all score
   std::vector<string> aWords;  //User all words
   std::vector<string> aResults; //User all results
};
#endif /* Gamer_H */
```

GamerGame.h

```
/*
 * File:   GamerGame.h
 * Author: JR
 *
 * Created on June 1, 2015, 11:48 PM
 */
#ifndef GAMERGAME_H
#define GAMERGAME_H
#define ROW 6
#define COL 2
struct GamerGame{
   int wLenght;//Word Lengths
   char *fBlanks;//Fill the blanks
   std::string HangmanGame[ROW][COL];

   //Initializes Variables
   GamerGame(int w){
      wLenght =w;
```

```
        fBlanks = new char[w];
        for (int i=0;i<ROW;i++)//Initialize the hangman string to empty
            for (int j=0;j<COL;j++)
                HangmanGame[i][j] = "";
    }


};

#endif /* GAMERGAME_H */
```