Project 1 –V2

CAT FIGHT

CSC-5

Patrick De La Cruz

July 29, 2014

Introduction

This is a text base fighting game.

Basically you just type in one of the provided commands to attack.

Your enemy, which is the computer, will also randomly attack simultaneously.

Each attack command is assigned a damage and a percentage success rate.

The less the chance of the chosen attack of landing, the higher the damage and vice versa.

The player and the computer have the same health to start and first one to zero loses.

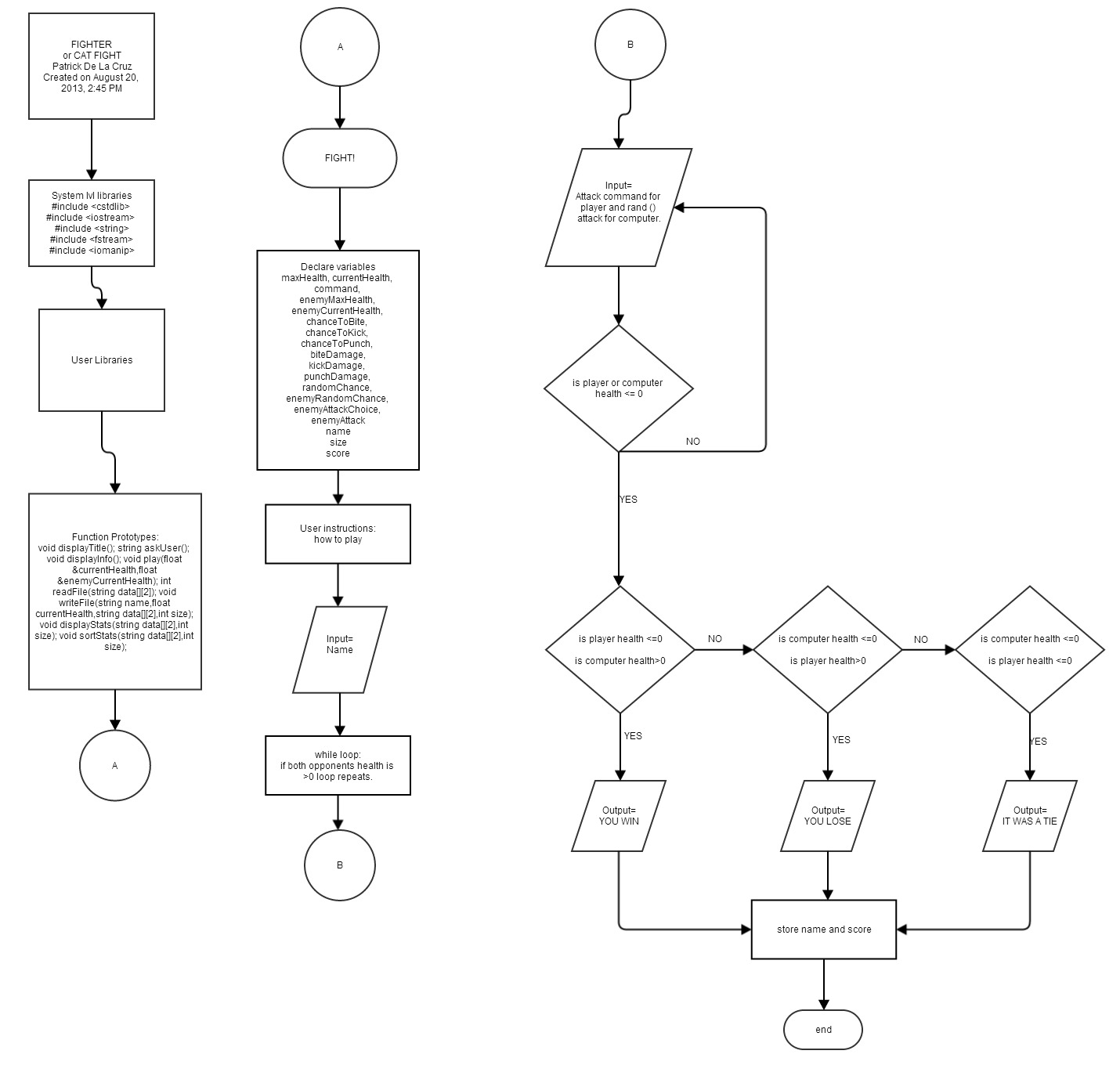
Summary

Lines of code: about 350

Number of variables: about 30

The first obstacle was figuring out how to implement the new concepts we have learned in the latest chapters. I was unsure about which sorting algorithm to use so I just used bubble since it was the one I understood the most. The program isn’t perfect, there are a few logical errors I could not fix. Sometimes it would show the list in alphabetical order and sometimes not so much. Also when typing in the game options command i.e. “info”, “score”, ”play”, or “exit”, it would not go through unless I typed it in a second time. One of the changes I wanted to make in the second version of the game was to implement a 2 player mode where you can challenge a friend instead of just playing against the computer but I could not get it to work. After a while I would just keep making syntax and logical errors to the point where I abandoned the 2 player idea. I was able to implement most of concepts learned in the later chapters as well as include a few that I had neglected with the first project. Over all it was a learning experience to say the least.

FLOWCHART



PROGRAM

#include <cstdlib>

#include <iostream>

#include <fstream>

#include <string>

#include <iomanip>

using namespace std;

void displayTitle();//this will display the Game Logo

string askUser();//this will ask user to play, exit, show info or show scores

void displayInfo();//this will display attack info

void play(float &currentHealth,float &enemyCurrentHealth);//this function will update the health of enemy and user according to the user inputs

int readFile(string data[][2]);//reads the text file and store the name and the score in the 2-d array

void writeFile(string name,float currentHealth,string data[][2],int size);//first write the 2-D array data[][] then write current name and score in the text file

void displayStats(string data[][2],int size);//display the contents of data[][]

void sortStats(string data[][2],int size);//this will sort the names in the data[][] in ascending order

/\*

\*

\*/

int main(int argc, char\*\* argv)

{

//player stats

float maxHealth=50;

float currentHealth=50;

//Enemy stats

float enemyMaxHealth=50;

float enemyCurrentHealth=50;

//user name

string name;

//stats of high scores

string data[20][2];

int i=0;

int size=0;

//exit game

string out;

//Title page

displayTitle();

size=readFile(data);//this function will return the number of names in the stats file

do

{

currentHealth=50;

enemyCurrentHealth=50;

//Ask User if he wnats scores ,info ,to play or to exit game

out=askUser();

if(out=="score")

{

size=readFile(data);//this function will return the number of names in the stats file

cout<<"CHRONOLOGICAL ORDER--------------->"<<endl;

cout<<"-----------------------------------"<<endl;

displayStats(data,size);

cout<<"\nALPHABETICLE ORDER--------------->"<<endl;

cout<<"-----------------------------------"<<endl;

sortStats(data,size);//this will sort the names in ascending order

displayStats(data,size); //display the names after sorting

}

else if(out=="info")

{

displayInfo();

out=askUser();;

}

else if(out=="exit")

{

cout<<"Thank you for playing";

return 0;

}

else if(out=="play")

{

cout<<"Enter your first name ";

cin>>name;

cout<<"FIGHT!"<<endl;

//While you and enemy are still alive

currentHealth=50;

enemyCurrentHealth=50;

while(enemyCurrentHealth > 0 && currentHealth > 0 )

{

play(currentHealth,enemyCurrentHealth);

}

//who wins or loses

if(currentHealth <= 0 && enemyCurrentHealth > 0)

{

cout<<"YOU LOST"<<endl;

cout<<"Your health: "<<currentHealth<<"/"<<maxHealth<<endl;

cout<<"Enemy health: "<<enemyCurrentHealth<<"/"<<enemyMaxHealth<<endl<<endl;

}

else if(enemyCurrentHealth <= 0 && currentHealth > 0)

{

cout<<"YOU WIN"<<endl;

cout<<"Your health: "<<currentHealth<<"/"<<maxHealth<<endl;

cout<<"Enemy health: "<<enemyCurrentHealth<<"/"<<enemyMaxHealth<<endl<<endl;

}

else

{

cout<<"IT WAS A TIE"<<endl;

cout<<"Your health: "<<currentHealth<<"/"<<maxHealth<<endl;

cout<<"Enemy health: "<<enemyCurrentHealth<<"/"<<enemyMaxHealth<<endl<<endl;

}

writeFile(name,currentHealth,data,size);

}

else if(out!="play"&&out!="info"&&out!="exit"&&out!="score")

{

cout<<"Invalid command"<<endl;

}

}while (1);

}

//Functions

void displayTitle()

{

cout<<" .@@@@@; 8@@@ i@@@@@@@@@ "<<endl;

cout<<" @@@. i@ ;@@@@. @@L "<<endl;

cout<<" @@@ @@ 1@@ @@L "<<endl;

cout<<" @@C @@t @@@ @@L "<<endl;

cout<<" @@@ t@@@@@@@@; @@L "<<endl;

cout<<" ,@@@i t@ @@. G@@ @@L "<<endl;

cout<<" ,@@@@0 8@G @@8 @@t "<<endl;

cout<<""<<endl;

cout<<" @@@@@ @@ @@@@8@ C@. @@ @@@@@@@@ "<<endl;

cout<<" @@ @@ @@ C@. @@ @@ "<<endl;

cout<<" @@@@@ @@ @@ 00@@ C@0000@@ @@ "<<endl;

cout<<" @@ @@ i@@ @@ C@. @@ @@ "<<endl;

cout<<" @@ @@ .@@@G C@ @@ GG "<<endl;

cout<<" C, : "<<endl;

cout<<" ;@8 ,8 "<<endl;

cout<<" t880 .08: "<<endl;

cout<<" .888C t88C "<<endl;

cout<<" 0888 ;8@@ ; "<<endl;

cout<<" ,888 @@@0 .0 "<<endl;

cout<<" ;888, 8888: ;8. "<<endl;

cout<<" t880 8888G 180 "<<endl;

cout<<" @8@; 88888 t@@L "<<endl;

cout<<" ;88t :@888 @88i i "<<endl;

cout<<" C88. .18888. ,@8@ @ "<<endl;

cout<<" C8@ ;@88@ @881 ,@0 "<<endl;

cout<<" . L8G L8881 @@8@ .@8f "<<endl;

cout<<" C@. .@88@ @888; 888: "<<endl;

cout<<" t8 18@@C @888 .@@@ "<<endl;

cout<<" .f 888@. :@888 .@@f "<<endl;

cout<<" , L8@f G888. ;@88 "<<endl;

cout<<" i8C ,888f L880 "<<endl;

cout<<" ,@ @88@ @881 "<<endl;

cout<<" 0 888: i88L "<<endl;

cout<<" i 080 G88 "<<endl;

cout<<" f@, 88@ "<<endl;

cout<<" :@ 08G "<<endl;

cout<<" 8 08. "<<endl;

cout<<" f 08 "<<endl;

cout<<" ;1 "<<endl;

cout<<" i "<<endl;

}

//reads the text file and store the name and the score in the 2-d array, so that we can sort it for later use

int readFile(string data[][2])

{

int i=0;

string empty;

//open the file in read mode

ifstream file;

file.open("stats.txt");

//store the names and score in a 2-d char array

while(file!=NULL)

{

file>>data[i][0];//word goes in i'th row and 1st column

file>>data[i][1];//number goes in i'th row and 2nd column

i++;//update the row number

}

file.close();//close the file after reading all the names and numbers

return i;

}

//first write the 2-D array data[][] then write current name and score in the text file

void writeFile(string name,float currentHealth,string data[][2],int size)

{

ofstream file;

file.open("stats.txt");

int i=0;

for(i=0;i<size;i++)

{

file<<data[i][0]<<"\n"<<data[i][1]<<"\n";

}

file<<name<<"\n"<<currentHealth;

}

//this will ask user to play or exit or for info or last scores

string askUser()

{

string out;

cout<<"Enter 'info' for damage statistics"<<endl;

cout<<"Enter 'score' for last 10 games statistics"<<endl;

cout<<"Enter 'play' to start game"<<endl;

cout<<"Enter 'exit' to end game"<<endl<<endl;

cin>>out;

return out;

}

//this will display attack info

void displayInfo(){

cout<<"Commands\tDamage\t\tsuccess rate\n";

cout<<"======================================\n";

cout<<"pull hair\t5\t\t75 percent\n";

cout<<"scratch\t\t10\t\t50 percent\n";

cout<<"bite\t\t20\t\t25 percent\n";

cout<<"punch\t\t45\t\t10 percent\n\n";

}

//display the contents of data[][] i.e. the NAMEs and SCOREs

void displayStats(string data[][2],int size)

{

int i=0;

if(size==0)

{

cout<<"No statistics !"<<endl;

}

else

{

cout<<"Player's Name Score"<<endl;

cout<<"============================="<<endl;

for(i=size-1;i>=0;i--)

{

cout<<data[i][0]<<"\t\t\t"<<data[i][1]<<endl;

}

}

cout<<"\n";

}

//this will sort the names in the data[][] in ascending order

void sortStats(string data[][2],int size)

{

int i,j;

string temp1;

//bubble sort algorithm used to sort the names

for(i=0;i<size;i++)

{

for(j=i;j<size;j++)

{

if(data[i][0]<data[j][0])

{

temp1=data[i][0];

data[i][0]=data[j][0];

data[j][0]=temp1;

temp1=data[i][1];

data[i][1]=data[j][1];

data[j][1]=temp1;

}

}

}

}

//this will update the health of enemy and user according to the user inputs/commands

void play(float &currentHealth,float &enemyCurrentHealth)

{

float maxHealth=50;

float enemyMaxHealth=50;

string command;

//hit chance

int chanceToBite=25;

int chanceToScratch=50;

int chanceToPullHair=75;

int chanceToPunch=10;

//damage from hit

float biteDamage=20;

float scratchDamage=10;

float pullDamage=5;

float punchDamage=44.50;

//chance is out of 100

int randomChance;

int enemyRandomChance;

int enemyAttackChoice;

string enemyAttack;

cout<<setprecision(2)<<fixed;

cout<<"Commands are 'punch', 'scratch', 'bite', 'pullhair'"<<endl;

cout<<"Your health: "<<currentHealth<<"/"<<maxHealth<<endl;

cout<<"Enemy health: "<<enemyCurrentHealth<<"/"<<enemyMaxHealth<<endl;

cin>>command;

cout<<endl;

//Input commands

if(command=="pullhair")

{

randomChance=rand()%100+1;

if(randomChance<=chanceToPullHair)

{

enemyCurrentHealth-=pullDamage;

}

}

else if(command=="scratch")

{

randomChance=rand()%100+1;

if(randomChance<=chanceToScratch)

{

enemyCurrentHealth-=scratchDamage;

}

}

else if(command=="bite")

{

randomChance=rand()%100+1;

if(randomChance<=chanceToBite)

{

enemyCurrentHealth-=biteDamage;

}

}

else if(command=="punch")

{

randomChance=rand()%100+1;

if(randomChance<=chanceToPunch)

{

enemyCurrentHealth-=punchDamage;

}

}

else

{

cout<<"Not a valid command"<<endl;

}

//enemy attack

enemyRandomChance=rand()%100+1;

enemyAttackChoice=rand()%4+1;

//attacks performed by enemy

switch(enemyAttackChoice)

{

case 1:{

enemyAttack="punch";

break;

case 2:

enemyAttack="pullhair";

break;

case 3:

enemyAttack="scratch";

break;

case 4:

enemyAttack="bite";

break;

}

}

//damage dealt by enemy

if(enemyAttack=="punch")

{

enemyRandomChance=rand()%100+1;

if(enemyRandomChance<=chanceToPunch)

{

currentHealth-=punchDamage;

cout<<"punched you in the face!"<<endl;

}

}

else if(enemyAttack=="scratch")

{

enemyRandomChance=rand()%100+1;

if(enemyRandomChance<=chanceToScratch)

{

currentHealth-=scratchDamage;

cout<<"you got scratched"<<endl;

}

}

else if(enemyAttack=="bite")

{

enemyRandomChance=rand()%100+1;

if(enemyRandomChance<=chanceToBite)

{

currentHealth-=biteDamage;

cout<<"you got bit"<<endl;

}

}

else if(enemyAttack=="pull hair")

{

enemyRandomChance=rand()%100+1;

if(enemyRandomChance<=chanceToPullHair)

{

currentHealth-=pullDamage;

cout<<"pulled your hair"<<endl;

}

}

}