**Project 2  
<Pokemon>**

**CSC-5 42829**

**Name: Christian Sandoval**

**Date: 05/04/2016**

**Introduction:**

**Title:** Pokemon Battle

This is based on the pokemon battles from Gameboy advance where the player’s who’s Hp reaches zero first loses. The player is supposed to think strategically to beat the computer’s attacks. Both the computer and player can lose miss… if the player does an incorrect move the player’s forfeits their attacking turn… There is also a weakness system where the damage is multiplied by 2.

**Summary:**

Project size: 700+ lines

Number of variables: 14+ (repeated a few in for health)

Number of constructs: 6?

I tried to recreate a pokemon battling concept but was unable to create a weakness damage variable but I have recreated the standard battle with possibilities of missing an attack, damaging difference between attacks and accuracy differences… I have also included a Health system using inputs and outputs to .dat files the default health is set to 150…

I also used a pause / sleep function to get the times to differ differently.

**Description:**

The main point of this program is to have a battle till a player hp is 0 or less than 0…

The User is asked to input attacks and the computer is also launching attacks…

**Sample Play:**

**Pseudo Code:**

main ()

Show Menu text and pokemon type options.

output type selected

Computer chooses random type

output type selected

Go to attack choice menu

user inputs attack

does attack and calculates damage and if it’s hit or miss and calls computers random attack

check

if playershp or computer hp is at zero if it is the program is terminated

**Major Variables**

|  |  |  |  |
| --- | --- | --- | --- |
| Type | Variable Name | Description | Location |
| Integer | draCho | Integer determines what the pokemon’s type will be. | Main() |
|  | cmDrag | Integer determines random what the pokemon’s type will be. | Dragchoi() |
| Unsigned short | acc | Random number generator for accuracy | accnum(unsigned short acc) |
| Integer | Attk | Number for attack choice | Attks() |
| Bool | Damage | Used for a true and false statement to damage or not | pelattk()  pnchatt()  biteAtt()  kickAtt()  celattk()  cnchatt()  citeAtt()  cickAtt() |
| Integer | curChp  curPhp | Used for player and computer HP | eletoPc()  pnDtoPC()  biteDPc()  kickDPc()  eletopy()  cnDtoPC()  citeDPc()  cickDPc() |

**Program:**

/\*

\* File: main.cpp

\* Author: Christian Sandoval

\* Created on April 30, 2016, 10:05 PM

\* Purpose: Pokemon Battle

\* To-do at bottom...

\*/

//System Libraries

#include <iostream>//I/O

#include <cstdlib> //Rand and Set Random

#include <ctime> //Utilize time to set the seed

#include <fstream> //Writing to a file

#include <cmath>//math

#include <iomanip> //Formatting output

#include <unistd.h>//used to pause time so that random seed generator can differ between pc and player

#include <vector>

using namespace std;

//User Libraries

//Global Constants

//Function Prototypes

unsigned short accnum(unsigned short);

void dragchoi();

void menutxt();

void comDrag();

void accnum();

void hitsuc();

//Attacks Menus

void attks();

void comattk();

//Basic Player attacks

void pelattk();//element attk

void eletoPc();//element damage/health calculator/watcher

void pnchatt();//punch attakc

void pnDtoPC();//Punch damage/health calculator/watcher

void biteAtt();//punch attakc

void biteDPc();//Punch damage/health calculator/watcher

void kickAtt();//punch attakc

void kickDPc();//Punch damage/health calculator/watcher

//Basic PC attacks

void celattk();//element attk

void eletopy();//element damage/health calculator/watcher

void cnchatt();//punch attakc

void cnDtoPC();//Punch damage/health calculator/watcher

void citeAtt();//punch attakc

void citeDPc();//Punch damage/health calculator/watcher

void cickAtt();//punch attakc

void cickDPc();//Punch damage/health calculator/watcher

//end/win/lose

void win();

void lose();

void wMenu();

void restart();

//pass by reference,use value from another function

void getRand(int &);

//Execution Begins Here!

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

{

//Title

cout << " Welcome to the pokemon battle simulation game\n"

" Your objective is to choose a pokemon and beat the computer\n"

"The Game will end when either player's pokemon's hp goes below 0."

<< endl << endl;

int curChp, curPhp;

//outputs for reseting health

ofstream outChp;//Output / out used can be anything but its used for file

ofstream outPhp;//Output / out used can be anything but its used for file

outChp.open("compHp.dat");

curChp = 150;

outChp << curChp;

outPhp.open("plyrHp.dat");

curPhp = 150;

outPhp << curPhp;

outPhp.close();

outChp.close();

menutxt();

//ask user for choice/bring up menu

dragchoi();

return 0;

}

//displaymenu text

void menutxt()

{

//Title

cout << "Prepare to battle choose your 1 pokemon type wisely!\n"

"--------Pokemon Type Menu--------\n"

"Press 1 for Fire type Pokemon .\n"

"Press 2 for Water type Pokemon.\n"

"Press 3 for Grass type Pokemon.\n"

"Press 4 for Eletric type Pokemon ." << endl;

}

//User choose pokemon type

void dragchoi()

{

const int TYPES=5;

string pokemon[TYPES];

pokemon[0]="Fire";

pokemon[1]="Water";

pokemon[2]="Grass";

pokemon[3]="Eletric";

int draCho, i(0);

ofstream outtype;//Output / out used can be anything but its used for file

do

{

cin >> draCho;

switch(draCho)

{

case 1:

{

cout<<endl;

cout<<"Thank you for choosing a " << pokemon[i] << " type." <<endl;

//computer picks randomly

comDrag();

//Saves type chosen for weakness's

outtype.open("plyrPokeType.dat");

outtype << draCho;

outtype.close();

attks();

cout<<endl;

break;

}

case 2:

{

cout<<endl;

i++;

cout<<"Thank you for choosing a " << pokemon[i] << " type." <<endl;

//computer picks randomly

comDrag();

//Saves type chosen for weakness's

outtype.open("plyrPokeType.dat");

outtype << draCho;

outtype.close();

attks();

cout<<endl;

break;

}

case 3:

{

cout<<endl;

i++;

i++;

cout<<"Thank you for choosing a " << pokemon[i] << " type." <<endl;

//computer picks randomly

comDrag();

//Saves type chosen for weakness's

outtype.open("plyrPokeType.dat");

outtype << draCho;

outtype.close();

attks();

cout<<endl;

break;

}

case 4:

{

cout<<endl;

i++;

i++;

i++;

cout<<"Thank you for choosing a " << pokemon[i] << " type." <<endl;

//computer picks randomly

comDrag();

//Saves type chosen for weakness's

outtype.open("plyrPokeType.dat");

outtype << draCho;

outtype.close();

attks();

cout<<endl;

break;

}

default:

{

cout<<"Please input proper choice\n" <<endl;

menutxt();

dragchoi();

}

}

}

while(draCho < 5);

}

//Random computer Pokemon Selector

void comDrag()

{

int cmDrag;

ofstream outtype;//Output / out used can be anything but its used for file

getRand(cmDrag);

switch(cmDrag)

{

case 1:

{

cout<<endl;

cout<<"Computer chose a Fire type" <<endl;

//Saves type chosen for weakness's

outtype.open("pcPokeType.dat");

outtype << cmDrag;

outtype.close();

cout<<endl;

break;

}

case 2:

{

cout<<endl;

cout<<"Computer chose a Water type" <<endl;

//Saves type chosen for weakness's

outtype.open("pcPokeType.dat");

outtype << cmDrag;

outtype.close();

cout<<endl;

break;

}

case 3:

{

cout<<endl;

cout<<"Computer chose a Grass type" <<endl;

//Saves type chosen for weakness's

outtype.open("pcPokeType.dat");

outtype << cmDrag;

outtype.close();

cout<<endl;

break;

}

case 4:

{

cout<<endl;

cout<<"Computer chose a Eletric type" <<endl;

//Saves type chosen for weakness's

outtype.open("pcPokeType.dat");

outtype << cmDrag;

outtype.close();

cout<<endl;

break;

}

default:

{

comDrag();

}

}

}

//random number generator

void getRand(int &cmDrag)

{

const int MIN\_VALUE = 0, MAX\_VALUE = 4;//Sets constant min value and max to give student workable problems

//seed time need for random values within max and min values

unsigned seed = time(0);

srand(seed);

cmDrag = (rand() % (MAX\_VALUE - MIN\_VALUE + 1)) +MIN\_VALUE;

}

//Random number generator for accuracy

unsigned short accnum(unsigned short acc)

{

const int MIN\_VALUE = 0, MAX\_VALUE = 100;//Sets constant min value and max to give student workable problems

acc = (rand() % (MAX\_VALUE - MIN\_VALUE + 1)) + MIN\_VALUE;

//return value of acc

return acc;

}

//\*\*\*\*\*\*\*\*\*\*\*\*Attack menu's\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void attks()

{

int attk;

cout << "--------Attack Menu--------\n"

"Press 1 for Special Type Attack.(Damage: 18)\n"

"Press 2 for Punch.(Damage: 13)\n"

"Press 3 for Bite.(Damage: 10)\n"

"Press 4 for Kick.(Damage: 8)" << endl;

do

{

cin >> attk;

switch(attk)

{

case 1:

{

cout<<endl;

cout<<"Your Pokemon Used it type attack" <<endl;

pelattk();

cout<<"Please wait\nComputer is deciding on attack" <<endl;

sleep(1);

comattk();

cout<<endl;

break;

}

case 2:

{

cout<<endl;

cout<<"Your pokemon used Punch attack" <<endl;

pnchatt();

cout<<"Computer is deciding on attack" <<endl;

sleep(1);

comattk();

cout<<endl;

}

case 3:

{

cout<<endl;

cout<<"Thank you for choosing Bite attack" <<endl;

biteAtt();

cout<<"Computer is deciding on attack" <<endl;

sleep(1);

comattk();

cout<<endl;

break;

}

case 4:

{

cout<<endl;

cout<<"Your pokemon used Kick attack" <<endl;

kickAtt();

cout<<"Computer is deciding on attack" <<endl;

sleep(1);

comattk();

cout<<endl;

break;

}

default:

{

cout<<"Please input proper choice for attack"<<endl;

}

}

}

while(attk < 5);

}

void pelattk()// special attack

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy

acc = accnum (acc);

//cout << acc << endl; displays random number generated...

//accuracy

bool damage;

if (acc <= 59 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 60 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

eletoPc();

}

}

}

//damage calculator and health updater

void eletoPc()

{

ofstream outPhp, outChp;//Output / out used can be anything but its used for file

ifstream inPhp, inChp, inType;//Input /used for reading files

//Declare variables

int curChp, curPhp, draCho, comDrag;

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

inType.open("plyrPokeType.dat");

inType >> draCho;

inType.close();

inType.open("pcPokeType.dat");

inType >> comDrag;

inType.close();

if(draCho == 1 && comDrag == 3)

{

cout << "Computer's pokemon is weak against fire move" << endl;

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curChp = (curChp - 18) + pow(2, 2);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outChp.open("compHp.dat");

outChp << curChp;

outChp.close();

}

else if(draCho == 2 && comDrag == 1)

{

cout << "Computer's pokemon is weak against water move" << endl;

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curChp = (curChp - 18) + pow(2, 2);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outChp.open("compHp.dat");

outChp << curChp;

outChp.close();

}

else if(draCho == 4 && comDrag == 2)

{

cout << "Computer's pokemon is weak against eletric move" << endl;

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curChp = (curChp - 18) + pow(2, 2);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outChp.open("compHp.dat");

outChp << curChp;

outChp.close();

}

else

{

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curChp = (curChp - 18);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outChp.open("compHp.dat");

outChp << curChp;

outChp.close();

}

if(curChp <=0 )

{

win();

}

}//end special attack by user

void pnchatt()//punch attack by player

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy if number is greater than 55 than hit is a success..

//cout << acc << endl; displays random number generated...

//accuracy

bool damage;

if (acc <= 55 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 56 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

pnDtoPC();

}

}

}

void pnDtoPC()

{

//Declare variables

int curChp, curPhp;

ofstream outChp;//Output / out used can be anything but its used for file

ifstream inChp;//Input /used for reading files

ofstream outPhp;//Output / out used can be anything but its used for file

ifstream inPhp;//Input /used for reading files

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curChp = curChp - 13;

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outChp.open("compHp.dat");

outChp << curChp;

outChp.close();

if(curChp <=0 )

{

win();

}

}// end punch attack by user

void biteAtt()//Bite attack by player

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy if number is greater than 55 than hit is a success..

acc = accnum (acc);

//cout << acc << endl; displays random number generated...

//accuracy

bool damage;

if (acc <= 45 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 46 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

biteDPc();

}

}

}

void biteDPc()

{

//Declare variables

int curChp, curPhp;

ofstream outChp;//Output / out used can be anything but its used for file

ifstream inChp;//Input /used for reading files

ofstream outPhp;//Output / out used can be anything but its used for file

ifstream inPhp;//Input /used for reading files

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curChp = curChp - 10;

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outChp.open("compHp.dat");

outChp << curChp;

outChp.close();

if(curChp <=0 )

{

win();

}

}//end bite attack

void kickAtt()//Kick attack by player

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy if number is greater than 55 than hit is a success..

acc = accnum (acc);

//cout << acc << endl; displays random number generated...

//accuracy

bool damage;

if (acc <= 35 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 36 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

kickDPc();

}

}

}

void kickDPc()

{

//Declare variables

int curChp, curPhp;

ofstream outChp;//Output / out used can be anything but its used for file

ifstream inChp;//Input /used for reading files

ofstream outPhp;//Output / out used can be anything but its used for file

ifstream inPhp;//Input /used for reading files

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curChp = curChp - 8;

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outChp.open("compHp.dat");

outChp << curChp;

outChp.close();

if(curChp <=0 )

{

win();

}

}//end kick attack

//\*\*\*\*\*\*\*\*\*\*\*\*Computer Radnom Attack\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void comattk()

{

const int MIN\_VALUE = 0, MAX\_VALUE = 4;//Sets constant min value and max to give student workable problems

//seed time need for random values within max and min values

unsigned seed = time(0);

srand(seed);

for(int cmattk; cmattk < 4;)

{

cmattk = (rand() % (MAX\_VALUE - MIN\_VALUE + 1)) +MIN\_VALUE;

switch(cmattk)

{

case 1:

{

cout<<endl;

cout<<"Computer chose Special type attack" <<endl;

//test hit success rate for attack patterns later add 2 or more attacks for fun

celattk();

attks();

cout<<endl;

break;

}

case 2:

{

cout<<endl;

cout<<"Computer chose Punch attack" <<endl;

//test hit success rate for attack patterns later add 2 or more attacks for fun

cnchatt();

attks();

cout<<endl;

break;

}

case 3:

{

cout<<endl;

cout<<"Computer chose Bite attack" <<endl;

//test hit success rate for attack patterns later add 2 or more attacks for fun

citeAtt();

attks();

cout<<endl;

break;

}

case 4:

{

cout<<endl;

cout<<"Computer chose Kick attack" <<endl;

//test hit success rate for attack patterns later add 2 or more attacks for fun

cickAtt();

attks();

cout<<endl;

break;

}

}

}

}

void celattk()// special attack

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy if number is greater than 55 than hit is a success..

acc = accnum (acc);

// cout << acc << endl;

//accuracy

bool damage;

if (acc <= 59 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 60 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

eletopy();

}

}

}

void eletopy()

{

ofstream outPhp, outChp;//Output / out used can be anything but its used for file

ifstream inPhp, inChp, inType;//Input /used for reading files

//Declare variables

int curChp, curPhp, draCho, comDrag;

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

inType.open("plyrPokeType.dat");

inType >> draCho;

inType.close();

inType.open("pcPokeType.dat");

inType >> comDrag;

inType.close();

if(draCho == 1 && comDrag == 3)

{

cout << "Your pokemon is weak against fire move" << endl;

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curPhp = (curPhp - 18) + pow(2, 2);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outPhp.open("plyrHp.dat");

outPhp << curPhp;

outPhp.close();

}

else if(draCho == 2 && comDrag == 1)

{

cout << "Your pokemon is weak against water move" << endl;

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curPhp = (curPhp - 18) + pow(2, 2);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outPhp.open("plyrHp.dat");

outPhp << curPhp;

outPhp.close();

}

else if(draCho == 4 && comDrag == 2)

{

cout << "Your pokemon is weak against eletric move" << endl;

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curPhp = (curPhp - 18) + pow(2, 2);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outPhp.open("plyrHp.dat");

outPhp << curPhp;

outPhp.close();

}

else

{

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curPhp = (curPhp - 18);

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outPhp.open("plyrHp.dat");

outPhp << curPhp;

outPhp.close();

}

if(curPhp <=0 )

{

lose();

}

}//end special attack by user

void cnchatt()//punch attack by player

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy if number is greater than 55 than hit is a success..

acc = accnum (acc);

// cout << acc << endl;

//accuracy

bool damage;

if (acc <= 55 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 56 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

pnDtoPC();

}

}

}

void cnDtoPC()

{

//Declare variables

int curChp, curPhp;

ofstream outChp;//Output / out used can be anything but its used for file

ifstream inChp;//Input /used for reading files

ofstream outPhp;//Output / out used can be anything but its used for file

ifstream inPhp;//Input /used for reading files

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curPhp = curPhp - 13;

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outPhp.open("plyrHp.dat");

outPhp << curPhp;

outPhp.close();

if(curPhp <=0 )

{

lose();

}

}// end punch attack by user

void citeAtt()//Bite attack by player

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy if number is greater than 55 than hit is a success..

acc = accnum (acc);

// cout << acc << endl;

//accuracy

bool damage;

if (acc <= 45 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 46 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

citeDPc();

}

}

}

void citeDPc()

{

//Declare variables

int curChp, curPhp;

ofstream outChp;//Output / out used can be anything but its used for file

ifstream inChp;//Input /used for reading files

ofstream outPhp;//Output / out used can be anything but its used for file

ifstream inPhp;//Input /used for reading files

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curPhp = curPhp - 10;

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outPhp.open("plyrHp.dat");

outPhp << curPhp;

outPhp.close();

if(curPhp <=0 )

{

lose();

}

}//end bite attack

void cickAtt()//Kick attack by player

{

//Random number generator

unsigned short acc; //attack success is 55+ max value assigned has been 100

//random seed

srand(static\_cast<unsigned int>(time(0)));

//get value of accuracy if number is greater than 55 than hit is a success..

acc = accnum (acc);

// cout << acc << endl;

//accuracy

bool damage;

if (acc <= 35 )

{

damage = false;

if(damage == false)

{

cout << "Miss" << endl;

}

}

else if (acc >= 36 )

{

damage = true;

if(damage == true)

{

cout << "Hit" << endl;

cickDPc();

}

}

}

void cickDPc()

{

//Declare variables

int curChp, curPhp;

ofstream outChp;//Output / out used can be anything but its used for file

ifstream inChp;//Input /used for reading files

ofstream outPhp;//Output / out used can be anything but its used for file

ifstream inPhp;//Input /used for reading files

//read info from a file

inChp.open("compHp.dat");

inChp >> curChp;

inChp.close();

inPhp.open("plyrHp.dat");

inPhp >> curPhp;

inPhp.close();

cout << "---HP: BEFORE ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

curPhp = curPhp - 8;

cout << "---HP: AFTER ATTACK---" << endl;

cout << "Computer HP: " << curChp << setw(10) << " Your HP: " << curPhp << "\n"<< endl;

outPhp.open("plyrHp.dat");

outPhp << curPhp;

outPhp.close();

if(curPhp <=0 )

{

lose();

}

}//end kick attack

void win()

{

//declare health variables

int curChp, curPhp;

//outputs for reseting health

ofstream outChp;//Output / out used can be anything but its used for file

ofstream outPhp;//Output / out used can be anything but its used for file

cout<< "\n"

"▕▔╲ ╱▔▏ \n"

" ╲╱╲ ╱╲╱ \n"

" ╲ ╲▂▂▂╱ ╱ ╱╲\n"

" ╱ ╲ ╱┈┈╲\n"

" ▏ ▆ ▂ ▆ ▕ ╱┈┈┈╱\n"

" ▕╭╮ ┳┻┳ ╭╮▏╲┈┈╱\n"

" ╲ ╯ ╰━╯ ╰ ╱ ╱┈┈╲\n"

" ╱ ╲ ╲┈┈┈╲\n"

" ▕╲ ▕ ▏ ╱▏ ╱ ┈ ╱\n"

" ▕ ▔▔ ▔▔ ▏╱ ┈ ╱\n"

" ▕ ▕▔ ┈╱\n"

" ╲ ╱▔▔\n"

" ▕▂╱▔▔▔╲▂▏\nYou have Won!!!!!\n" << endl;

sleep(3);

outChp.open("compHp.dat");

curChp = 150;

outChp << curChp;

outPhp.open("plyrHp.dat");

curPhp = 150;

outPhp << curPhp;

outPhp.close();

outChp.close();

wMenu();

}

void lose()

{

//declare health variables

int curChp, curPhp;

//outputs for reseting health

ofstream outChp;//Output / out used can be anything but its used for file

ofstream outPhp;//Output / out used can be anything but its used for file

cout<< "\n"

"▕▔╲ ╱▔▏ \n"

" ╲╱╲ ╱╲╱ \n"

" ╲ ╲▂▂▂╱ ╱ ╱╲\n"

" ╱ ╲ ╱┈┈╲\n"

" ▏ ▆ ▂ ▆ ▕ ╱┈┈┈╱\n"

" ▕╭╮ ┳┻┳ ╭╮▏╲┈┈╱\n"

" ╲ ╯ ╰━╯ ╰ ╱ ╱┈┈╲\n"

" ╱ ╲ ╲┈┈┈╲\n"

" ▕╲ ▕ ▏ ╱▏ ╱ ┈ ╱\n"

" ▕ ▔▔ ▔▔ ▏╱ ┈ ╱\n"

" ▕ ▕▔ ┈╱\n"

" ╲ ╱▔▔\n"

" ▕▂╱▔▔▔╲▂▏\nYou have lost!!!\n" << endl;

sleep(3);

outChp.open("compHp.dat");

curChp = 150;

outChp << curChp;

outPhp.open("plyrHp.dat");

curPhp = 150;

outPhp << curPhp;

outPhp.close();

outChp.close();

wMenu();

}

void restart()

{

//declare health variables

int curChp, curPhp;

//outputs for reseting health

ofstream outChp;//Output / out used can be anything but its used for file

ofstream outPhp;//Output / out used can be anything but its used for file

sleep(3);

outChp.open("compHp.dat");

curChp = 150;

outChp << curChp;

outPhp.open("plyrHp.dat");

curPhp = 150;

outPhp << curPhp;

outPhp.close();

outChp.close();

menutxt();

dragchoi();

}

void wMenu()

{

int gamecho;

cout << "\nWhat would you like to do?\n\n"

"Press 1 to restart the game\n"

"Press 2 to end the program" << endl;

do

{

cin >> gamecho;

switch(gamecho)

{

case 1:

{

cout << "Game is restarting please wait..." << endl;

restart();

}

case 2:

{

cout << "Good bye program now closing" << endl;

sleep(2);

exit(0);

}

default:

{

cout<<"Please input proper choice" <<endl;

}

}

}

while(gamecho < 2);

}