Project 1

**Title:** Monopoly Game

**Course:** CSC-17A-48983

**Due Date:** 10/25/15

**Author:** Hector Espinoza

Table of Contents

1. Introduction

2. Table of Contents

3. How to play

4. Project summary

4. Description

7. Flow chart

8. Pseudo code

9. Major Variables

10. Code

**How to play:**

Title: Monopoly

**Objective:** The objective of the game is to become the wealthiest player through, buying, renting, and selling property. Each player starts with $1500 to begin their endeavor of owning the entire board. I made this game because I thought it would be a challenging game to make at my level of knowledge.

**The play:** After you have completed your roll. Two or more players may rest on the same place at the same time. Depending on the space you may be entitled to buy property, or be obligated to pay rent, pay taxes, draw a Chance or Community Chest card, Go to Jail, or etc...

**Go:** Each time a player passes go the player receives $200.

**Buying Property:** Whenever you land on an unowned property you may buy that property from the Bank at its printed price.

**Paying Rent:** When you land on a property that is owned by another player, the owner collects rent from you in accordance with the rent cost.

**Chance & Community Chest:** When you land on either of these spaces, take the top card from the deck indicated, follow the instructions.

**Income Tax:** If you land here you must pay $200 to the bank.

**Jail:** You land in jail when you move to the space marked "Go to Jail", if you land on jail and are not in jail then you are "just visiting".

**Free Parking:** A player landing on this place does not receive any money, property or reward of any kind. This is just a "free" resting-place.

**Bankruptcy:** When you are declared bankrupt, if you owe more than you can pay either to another player or to the bank. You must retire from the game.

**Project Summary:**

Project Size: 900+ lines

I used an object oriented game for the property’s (name, buy cost, and rent cost). Used arrays of all sorts to make the game simple for me to code and to keep computer and player separate to avoid problems. The game itself is nearly complete, I have moving along the board one space at a time, community chest and chance, buying property, going to jail, and paying taxes and paying for rent. It still needs other functions to be added for example: get out of jail free card, houses and hotels, and selling property.

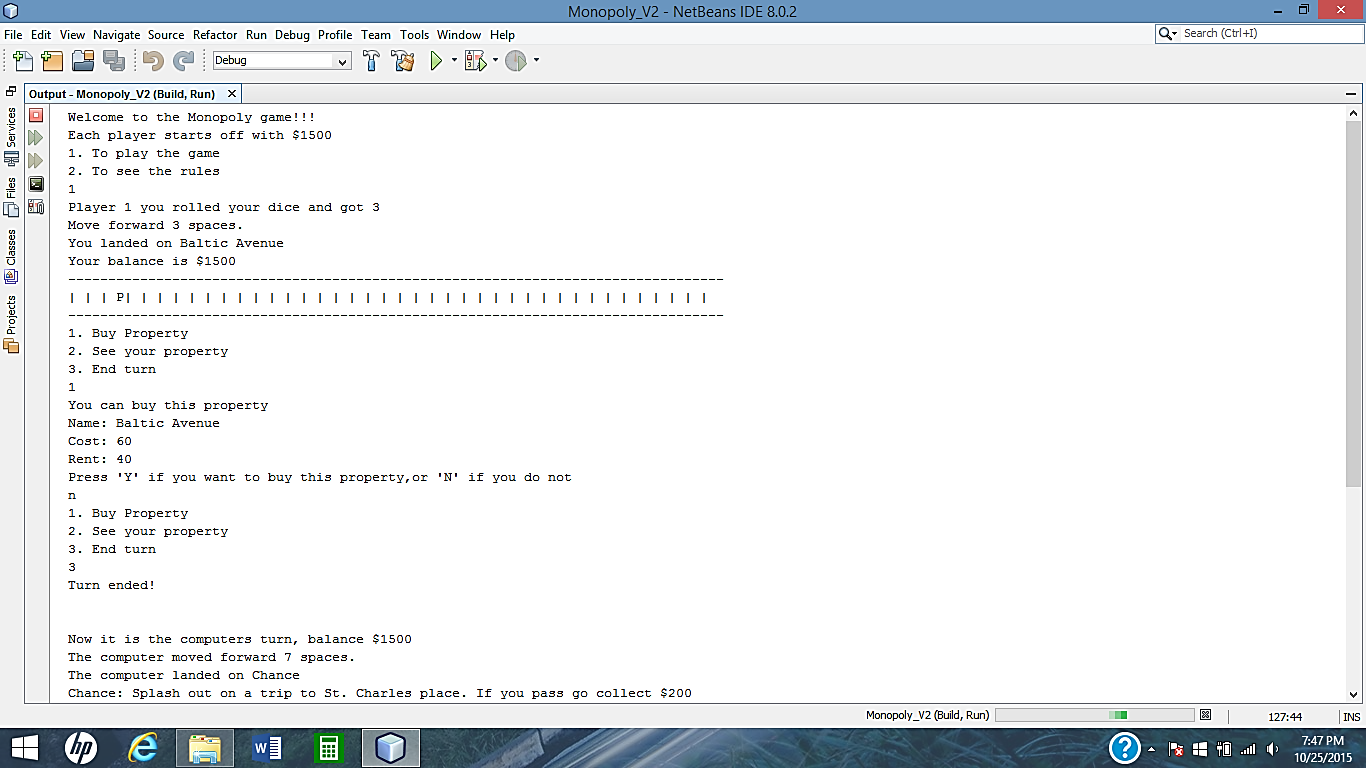
My game meets the criteria because it has objects and classes, a wide variety of arrays, and its concepts to flow together were a struggle that took me days in and day out of work together properly. This game was a challenge for me because there were so many variables to the game that makes them constantly change. In order for me to get over those challenges I decided to have separate functions for the computer, and the player to keep the program from being a mess or disrupting the other players.

**Description:**

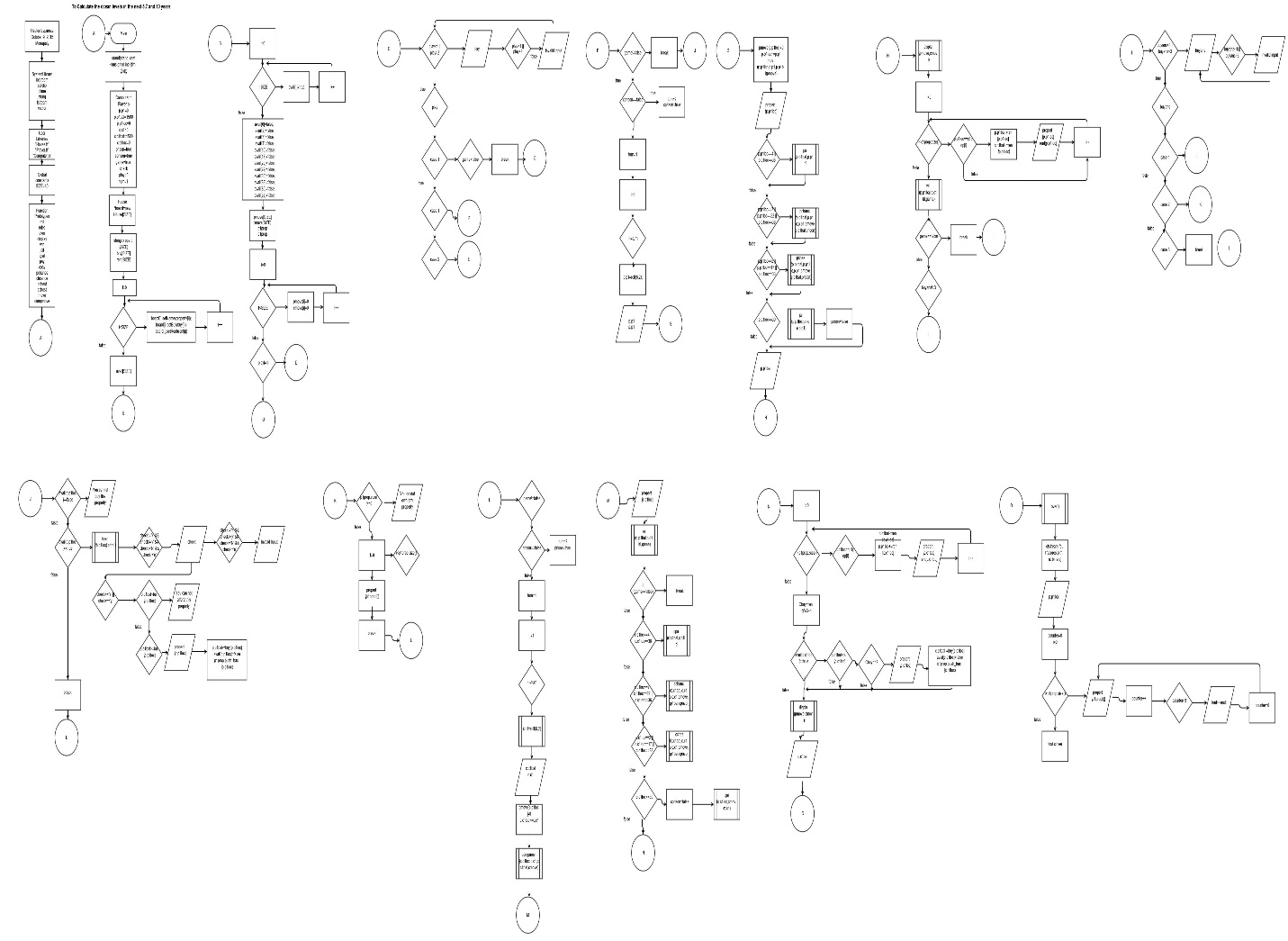
One problem I kept having, was that when community chest was called it would call both player and computer. So to fix that I used separate functions for each player and that was easier for me to keep to bugs to a minimum. Now I have a better idea on how to program the community chest and chance functions so that the program is more compact. Another issue I came across was that to display the board like a regular monopoly board, it would have taken me just as much time as it would have been to program the game. Instead I decided to just go with a horizontal board to make it simple to follow.

**Inputs and outputs:**

Initially the game will ask you to either press 1 to play the game or press 2 to view the rules. For this demonstration we are going to press 1, and after doing so the game will roll the two dice for you. Moving you forward along the board, display the property you landed on, if landed on chance, community chest, or taxes it will automatically do what the card says or pay taxes. Next the game will display the board, and ask the user to choose to buy the property, see the property you own or end turn. If the player land on a property he/she can choose to buy it, press 1 to buy. If the user is allowed to buy the property it will display the name, buying cost, and rent. The user can press y to buy or n to not buy. After the user decides, he can then end his turn. Afterwards it’s the comptuers turn to roll and play the game. Once the computers turn is done then the game continues until there is a winner.

**Visual of input and output:**

**FlowChart:**



**Pseudo Code:**

Call the random number genertaor.

Declare the variables and set the arrays equal to 0 or true;

After decalreing the arrays to true or 0 set the exceptions equal to false.

Prompt the user to either play the game or display the rules.

If the game is not over then palyer 1 can take his turn.

If computer is in jail then the palyer gets to roll twice, if the computer is not then it is set back to 1 roll.

Player 1 rolls the dice and moves forward.

After check to see if player landed on chance, community chest, taxes, or jail.

Display the board. Check if player lands on a computer owned location.

Check to see if there is a winner, if not continue playing.

If the location player landed on is available then propmpt the user to either buy the location or to pass.

If the player says yes to buying the location then sets the location equal to false and subtracts the buy amount from player 1’s bank account.

The player can then choose to see his/her property or end the turn.

If the player chooses to view property then it will display all the loactions player 1 owns.

Otherwise the player can end the game and it will become the computers turn.

The computer will roll if the game is not over. If the game is over then the computer will not be able to roll.

Then it will check to see if player is in prison, if so then the computer gets to roll twice.

The computer will now roll and move forward the dice amount spaces.

It will determine if the computer or player won, if not then the game continues.

Checks to see if player landed on taxes, community chest, chance, or jail.

Then checks to see if the computer landed on a player owned property.

If the computer can buy the property it will generate a random number eitehr 1,2, or 3.

If the answer is 2 then the computer will buy the property and the property will no longer be able to be bought.

Once the game is over, where either player becomes bankrupt then it will display the text ”Game over”

Then input players balance and the property’s he/she owned that game.

**Major variables:**

|  |  |  |
| --- | --- | --- |
| Variables | Location | Name |
| Functions | Above main | Win(); |
| Vectors | Line 104 | P1prop |
| Bool | Declaring variables | Prison=true; |
| Int | Declaring Variables | Play=0; |
| Structures | In the player.h file | Player p; |
| Objects and classes | In house.h file | House |
| Dynamic arrays | Declaring variables | Board |
| For loops | Line 133 | I<=turn |
| If-else | Line 160 | p.p1loc==c1prop[i] |
| Do-while | Line 288 | While game=true; |
| Switch | Line 179 | (buyans) |
| Random number | Roll function | Roll(); |
| Binary files | Line 299 | Ofstream fout |
| Reading a file | rules function | Rules(); |
| String | Declaring variables | Property[]; |

**Code:**

/\*

\* File: main.cpp

\* Author: Hector Espinoza

\* Created on October 9, 2015, 11:40 PM

\* Purpose: Monopoly

\*/

//System Libraries

#include <iostream>

#include <cstdlib>

#include <ctime>

#include <string>

#include <fstream>

#include <vector>

using namespace std;

//User Libraries

#include "House.h"

#include "Player.h"

#include "Computer.h"

//Global constants

const int SIZE=40;//size of the board

//Function Prototypes

unsigned char roll(unsigned char,unsigned char);//roll dice

void rules();//couts the rules of the game

void over();//couts game over pic

void display(int[],int[]);//displays the board

int win(int,int,bool&);//determine the winner

int jail(int&,int[],int&);

int cjail(int&,int[],int&);

int pay(int &,int);//player income tax

int cpay(int &,int);//computer income tax

int pchance(int &,int &,int &,int[] ,int &,bool&);//Player chance

int cchance(int &,int &,int &,int [],int &,bool&);//Computer chance

int pchest(int &,int &,int &,int [],int &,bool&);//Player community chest

int cchest(int &,int &,int &,int [],int &,bool&);//computer community chest

int move(int&,int&,int&,int[]);//player moves thru the board

int compmove(int &,int &,int &,int[]);//computer moves thru the board

//Execution Begins here

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

//Set the random number seed and size the array

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

//Declare Variables

Player p;

Computer c;

p.p1=0;//player dice

p.p1bal=1500;//player bank account

p.p1loc=0;//player location on the board

c.c1=0; //computer dice

c.c1bal=1500;//computer bank account

c.c1loc=0;//computer location on the board

bool prison=true;//misses a turn

bool cprison=true;//computer misses a turn

bool game=true;//to continue the game until there is a winner

char check;//is the player would like to buy

int play=0;//play the game or see rules

int buyans;//If the user would like to buy the property

int turn=1; //Turns if the player or computer land on jail

House \*board=new House[SIZE];//dynamic array to hold the name, buy and rent of the property

string property[SIZE]={"Go","Mediterranean Avenue","Community Chest","Baltic Avenue"," Income Tax","Reading Railroad",

"Oriental Avenue","Chance","Vermont Avenue","Connecticut Avenue","In Jail or Just visiting","St. Charles Place",

"Electric Company","States Avenue"," Virginia Avenue","Pennsylvania RailRoad","St. James Place",

"Community Chest","Tennessee Avenue"," New York Avenue","Free Parking","Kentucky Avenue","Chance",

"Indiana Avenue","Illinois Avenue"," B. & O. Railroad","Atlantic Avenue","Ventnor Avenue",

"Water Works","Marvin Gardens","Go to Jail","Pacific Avenue","North Carolina Avenue",

"Community Chest","Pennsylvania Avenue","Short Line","Chance","Park Place","Luxury Tax","BoardWalk"};

int buy[SIZE]={0,60,0,60,0,200,100,0,100,120,0,140,150,140,160,200,180,0,180,

200,0,220,0,220,240,200,260,260,150,280,0,300,300,0,320,200,0,350,0,400};

int rent[SIZE]={0,20,0,40,0,25,60,0,60,80,0,100,75,100,120,250,140,0,140,160,

0,180,0,180,200,250,220,220,75,240,0,260,260,0,280,250,0,350,0,500};

for(int i=0;i<SIZE;i++){

board[i].setName(property[i]);

board[i].setBuy(buy[i]);

board[i].setRent(rent[i]);

}

bool avail[SIZE];//if the house is owned or available

//sets the whole board to true available to buy

for(int i=0;i<SIZE;i++){

avail[i]=true;

}

//Sets the chance,go,income tax, etc.. to not able to buy

avail[0]=false;

avail[2]=false;

avail[4]=false;

avail[7]=false;

avail[10]=false;

avail[17]=false;

avail[20]=false;

avail[22]=false;

avail[30]=false;

avail[33]=false;

avail[36]=false;

avail[38]=false;

int pmove[SIZE];//is the player's board

int cmove[SIZE];//the computers board

vector <int> p1prop;

vector <int> c1prop;

//sets the whole array to 0

for(int i=0;i<SIZE;i++){

pmove[i]=0;

cmove[i]=0;

}

do{

do{

cout<<"Welcome to the Monopoly game!!!\n";

cout<<"Each player starts off with $1500\n";

cout<<"1. To play the game\n2. To see the rules\n";

cin>>play;

if(play<1 || play>2){

cout<<"Invalid Input\n";

}

}while(play<1 || play>2);

switch(play){

case 1:{

do{

if(game!=false){

if(cprison==false){

turn=2;

cprison=true;

}else{

turn=1;

}

for(int i=1;i<=turn;i++){

p.p1=roll(6,2);

cout<<"Player 1 you rolled your dice and got "<<p.p1<<endl;

cout<<"Move forward "<<p.p1<<" spaces."<<endl;

pmove[p.p1loc]=0;

p.p1loc+=p.p1;

move(p.p1loc,p.p1,p.p1bal,pmove);

cout<<"You landed on "<<property[p.p1loc]<<endl;

if(p.p1loc==4 || p.p1loc==38){

pay(p.p1bal,p.p1loc);

}

if(p.p1loc==7 || p.p1loc==22 || p.p1loc==36){

pchance(p.p1bal,p.p1loc,p.p1,pmove,c.c1bal,prison);

}

if(p.p1loc==2 || p.p1loc==17 || p.p1loc==33){

pchest(p.p1bal,p.p1loc,p.p1,pmove,c.c1bal,prison);

}

if(p.p1loc==30){

prison=false;

jail(p.p1loc,pmove,p.p1);

}

cout<<"Your balance is $"<<p.p1bal<<endl;

//Output the board

display(pmove,cmove);

//Used to check if player lands on a computer owned location

for(int i=0;i<c1prop.size();i++){

if(p.p1loc==c1prop[i]){

p.p1bal-=rent[p.p1loc];

c.c1bal+=rent[p.p1loc];

cout<<"Computer owns "<<property[p.p1loc]<<" you owe $"<<rent[p.p1loc]<<" for rent\n";

}

}

//To determine the winner

win(p.p1bal,c.c1bal,game);

if(game==false){

break;

}

do{

do{

cout<<"1. Buy Property\n2. See your property\n3. End turn\n";

cin>>buyans;

if(buyans<1 || buyans>3){

cout<<"Invalid Input\n";

}

}while(buyans<1 || buyans>3);

switch(buyans){

case 1:{

if(avail[p.p1loc]==false){

cout<<"You cannot buy this property\n";

}else if(avail[p.p1loc]==true){

cout<<"You can buy this property\n";

board[p.p1loc].print();

do{

cout<<"Press 'Y' if you want to buy this property,or 'N' if you do not\n";

cin>>check;

if(check!='Y' && check!='y' && check!='N' && check!='n'){

cout<<"Invalid Input\n";

}

}while(check!='Y' && check!='y' && check!='N' && check!='n');

if(check=='Y' || check=='y'){

if(p.p1bal<buy[p.p1loc]){

cout<<"You can not afford this property\n";

}else if(p.p1bal>=buy[p.p1loc]){

cout<<"You purchased "<<property[p.p1loc]<<endl;

p.p1bal-=buy[p.p1loc];//Subtracts the cost to buy from the players balance

avail[p.p1loc]=false;//Is no longer available

p1prop.push\_back(p.p1loc);//stores the location of where the player owns

}

}

}

break;

}

case 2:{

if(p1prop.size()==0){

cout<<"You do not own any property\n";

}else{

//Cout the property the player owns

for(int i=0;i<p1prop.size();i++){

cout<<"You own "<<property[p1prop[i]]<<endl;

}

}

break;

}

case 3:{

cout<<"Turn ended!\n\n\n";

break;

}

default:cout<<"Error: In the switch statment\n";

}

}while(buyans!=3);

}

}

if(game!=false){

if(prison==false){

turn=2;

prison=true;

}else{

turn=1;

}

for(int i=1;i<=turn;i++){

c.c1=roll(6,2);

cout<<"Now it is the computers turn, balance $"<<c.c1bal<<"\n";

cout<<"The computer moved forward "<<c.c1<<" spaces."<<endl;

cmove[c.c1loc]=0;

c.c1loc+=c.c1;

compmove(c.c1loc,c.c1,c.c1bal,cmove);

cout<<"The computer landed on "<<property[c.c1loc]<<endl;

win(p.p1bal,c.c1bal,game);//determines the winner

if(game==false)break;//doesn't let the computer turn happen if player wins

//calls the income tax function

if(c.c1loc==4 || c.c1loc==38){

cpay(c.c1bal,c.c1loc);

}

//calls the chance function

if(c.c1loc==7 || c.c1loc==22 || c.c1loc==36){

cchance(c.c1bal,c.c1loc,c.c1,cmove,p.p1bal,cprison);

}

//calls the chest function

if(c.c1loc==2 || c.c1loc==17 || c.c1loc==33){

cchest(c.c1bal,c.c1loc,c.c1,cmove,p.p1bal,cprison);

}

//Calls the jail function

if(c.c1loc==30){

cprison=false;

cjail(c.c1loc,cmove,c.c1);

}

//Used to check if player lands on a computer owned location

for(int i=0;i<p1prop.size();i++){

if(c.c1loc==p1prop[i]){

c.c1bal-=rent[c.c1loc];

p.p1bal+=rent[c.c1loc];

cout<<"You own "<<property[c.c1loc]<<" the computer owes you $"<<rent[c.c1loc]<<" for rent\n";

}

}

int Cbuy=rand()%3+1;

if(avail[c.c1loc]==true){

if(c.c1bal>buy[c.c1loc]){

if(Cbuy==2){

cout<<"The computer purchased "<<property[c.c1loc]<<endl;

c.c1bal-=buy[c.c1loc];//Subrtracts the cost to buy from the players balance

avail[c.c1loc]=false;//Is no longer available

c1prop.push\_back(c.c1loc);//stores the location of where the player owns

}

}

}

display(pmove,cmove);

cout<<"Computer's turn ended, balance $"<<c.c1bal<<"\n";

cout<<"\n\n";

}

}

}while(game);

break;

}

case 2:{

rules();

break;

}

}

}while(play!=1);

over();

//Writes to a file::binary the p1 balance and property's

ofstream fout("scores.txt",ios::binary);

cout<<"Writing down your score to a file..\n";

fout<<"Player 1 balance: $"<<p.p1bal<<endl;

//Cout the property the player owns

int counter=0;

for(int i=0;i<p1prop.size();i++){

fout<<"You own "<<property[p1prop[i]]<<", ";

counter++;

if(counter>3){

fout<<endl;

counter=0;

}

}

fout.close();

cout<<"Thank you for playing!!"<<endl;

return 0;