**ESE 224 2013 Fall Project Description**

This project aims to give the students a great opportunity to get better in C++ basic concepts taught in class including object oriented programming, template, overloading, stream I/O, basic data structures, algorithms, etc. You can form groups of 1-4 students to work on the project.

Some parts of the design have restrictions in terms of the implementation, e.g., what functions should be defined in class, how many levels of classes and how they are nested in terms of inheritance. This is to help you to better understand and grasp practical object oriented programming skills. Other than these requirements you have the freedom to realize the functionality by using appropriate data structures. You are expected to explain in details about your design considerations in your final report.

Grading is based on the functionality of your program (60%), your format and style of coding

(20%) and your final report (20%). The design is partitioned into basic task and bonus task from a functionality point of view. In format and style of coding we seek for **succinct yet smart coding and sufficient comments.** Every group is required to submit a soft copy of your source code (.cpp and .h), executable file (.exe and make sure it can work on another computer other than your own), and a project report. In your project report, please include: (1) architecture of your design, including the detailed explanation about the functionality of each class, and each function you have created. (2) Descriptions of the algorithm and design considerations of your functions; (3) the contribution of each group member.

**Project Description**

* The project is to realize a famous board game most of you definitely know named ‘MONOPOLY’(<http://en.wikipedia.org/wiki/Monopoly_(game)>. In the game, there are 4 players. They roll two dices, move their markers, and then buy the cities on which the players stand. Finally, the player who have the most money win. Your final goal of the project is just like this game. In other words, you are going to play/built/program ‘Computer-Version MONOPOLY.’ Your game will be displayed on Console window as usual. For this project, some basic knowledge will be given to you in the following context regarding the rules of this game so that you do not need to refer to any other materials



***Basic rules and Features***

* The number of players must be 4. You are Number 1 and the others are Computer. So, only player you can control is Number 1. The others are controlled by the computer randomly. Also, each player is going to have $10000 and has each name
* Four players play this game; 1 player is played by the user, other 3 are played by computer. Each player starts with $10,000 and takes a turn to roll a die to move around the board. When the player arrives at a city that is not owned by anyone, the player gets to choose whether or not to buy the city. For the first player, the user will make the decision. Other players buy the city (if they can) for 50% chance (use srand() for this). Once a city is bought by a player, any other players who stay in the city for 1 turn have to pay the owner of city 1/5 of amount the city is worth. At the end of X turns, assigned by the user at the beginning of the game, the winner is to be determined by the amount of money the player owns in his pocket. (Do not consider cities for this!)
* Each city has its own ID\_Number. If the ID\_Number is 2, 6, 10, 14, …, we define the corresponding city as a good city. If the ID\_Number is 3, 7, 11, 15, …, we define the corresponding city as a bad city. When a player arrives at a good city, the player gets to play a mini card game. The card game is really simple. The player randomly draws 2 cards from a deck, and if they have the different colors, the player wins the amount corresponding to [$100\*(card1 number + card2 number)]. For this calculation, assume J = 11, Q = 12, K = 13, A = 14. When a player arrives at a bad city, the player loses $500.

**Design Task**

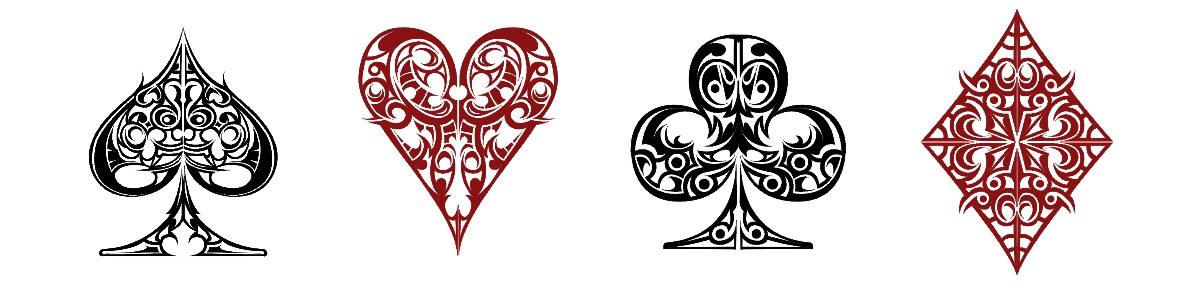
1. Condition
2. Player

* The number of players must be 4. You are Number 1 and the others are Computer. So, only player you can control is Number 1. The others are controlled by the computer randomly. Also, each player is going to have $10000 and has each name

1. Map

* The map should be a 5 by 5 array as below; you will only use the outer-most elements. Players are going around the map with the number of two dices. Each cell is a city such as New York or Seoul. Each city should be represented by ‘\*’ and each player should be represented on the map by ‘r’ for the player controlled by the user, and ‘1,’ ‘2,’ and ‘3’ for other players controlled by the computer. If there are more than 1 player in one city, you need to show only one of the players’ ID\_Number. Refer to the expected output in the last page for better understandings.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Start  City1 | City2  (good city) | City3  (bad city) | City4 | City5 |
| City16 |  | | | City6  (good city) |
| City15  (bad city) | City7  (bad city) |
| City14  (good city) | City8 |
| City13 | City12 | City11  (bad city) | City10  (good city) | City9 |

1. Special Point :Card Game Sample

* The card deck has 52 cards in the deck and you need to shuffle the deck every time you play the game. The player can draw 2 cards from the card deck and you need to print out the card that player drew. If the color of the 2 cards are not the same, the player will get money from the game. The amount of money is decided by the points on the card. For example you draw a heart 2 and spade K, and you can get
* 100\*(2+13) dollars from this game. If the color of the two cards are the same, the player will not get money from the game.

1. Input and Output
2. What the input are

* The input that you are going to put is the number of TURNS so that the game is going to finish after the turns are over. For example, once you put the number of turns, 3, the game is over after 3 times you roll the dices. Also, every time you roll the dice, you are going to stand a city or some special place such as card game. If you stand on a city, you have to be asked if you want to buy the city or not.

1. What the output are

* The output contains the information of the players such as the money the player has, the cities the player has, and the place where the player stand. Also, after the game is over, the result of the game should be on as well. The results is who the winner is.

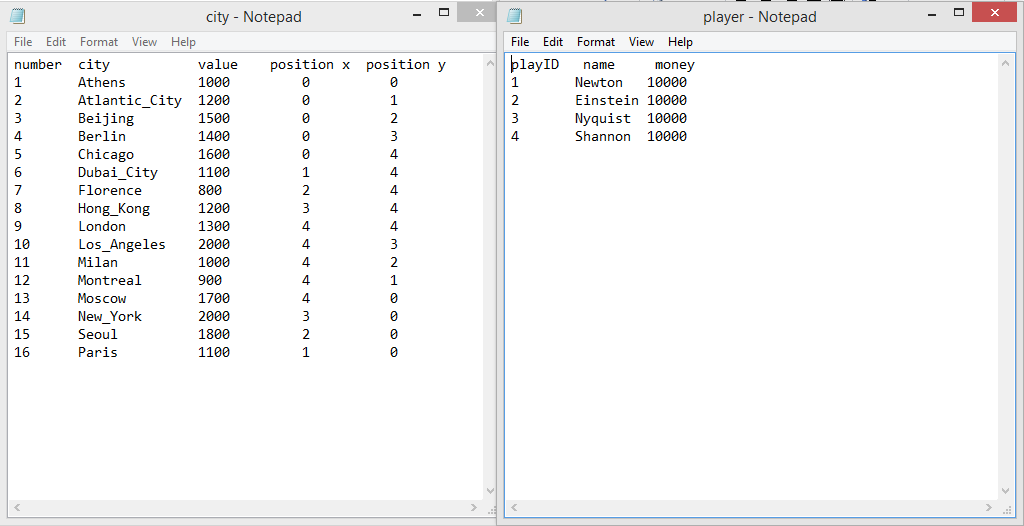
**Implementation**

**(This part is a hint for your project, you can write your own code using these examples.**

**It is only a part of you project code.)**

You will need 2 txt files: 1 for city and 1 for player.

Example text files:



Header files:

1. Card

Parameters: char suit, int rank

Constructor: default

Card(char suit, int rank);

Accessors: getRank() const;

getSuit() const;

method: void displayCard(ostream& outS) const;

Make a card class with accessors and a method that displays the card.

1. CardDeck

Parameters: vector<Card> theDeck

Constructor: default

Method: void shuffleDeck();

Card draw();

vector<Card>theDeck should contain 13 cards in each suit (52 in total).

draw() method draws one card from theDeck and returns the card.

shuffleDeck() method shuffles the deck.

1. City

Parameters: int id, int ownerId, double value, double charge, int xcoord, int ycoord,

String cityname, String owner, city\* nextCity;

Constructor: default

City(int idN, string ncityName, double nValue, int x, int y);

Accessors (getters)

Mutators (setters)

Method:

printCity();

Each city’s own id number is stored in int id. Owner id is in ownerId.

The price of the city is stored in int value, and int charge is 1/5 of the value.

The xcoord and ycoord are used to give the city’s x and y coordinates in the 5X5 array.

printCity() prints out all the parameters.

1. Player

Parameters: String name, int playerId, double money, vector<int> cityOwned,

int cityX, int cityY, int cityID

Constructor: default

player(string name, double money, int nplayerid);

Accessors (getters)

Mutators (setters)

Method: double loseMoney(double m);

double gainMoney(double m);

addCity(int cityId);

city\* move(int step, vector<city> cityset);

printPlayer();

String name is for the player name and playerId is for the player’s id number.

The double money means how much money is in the player’s pocket.

vector<int> cityOwned stores the corresponding ID numbers of the cities owned by the player.

For the player’s current position in the 5X5 array, cityX and cityY will be used.

All the methods should be straightforward based on the information given above.

1. Function

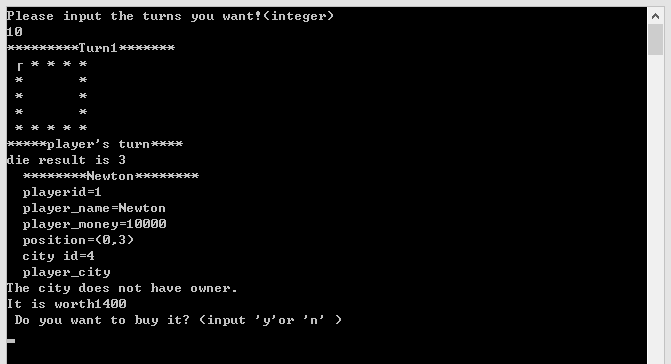
How to design this class is entirely up to you.

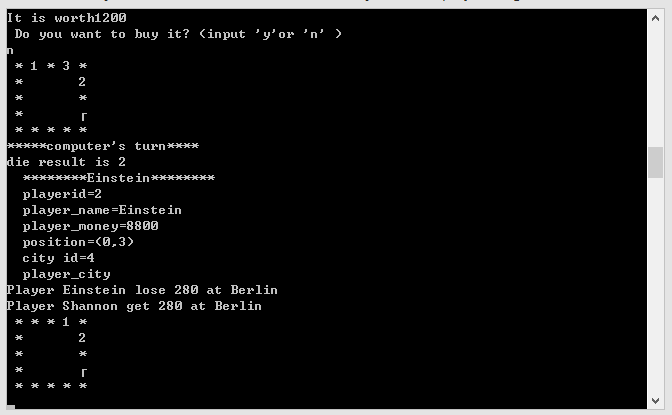
You should write any functions that are neccessary to implement the game described in ***Basic rules and Features.***

Hint: You might want to write functions for making/displaying the map, checking for good/bad cities, checking for ownership of the cities the player’s currently located, etc

The main cpp source file should #include these <headerfiles> and utilize methods in the header files to implement the game.

Expected Output:





**Bonus Points (15%)**

1. New Map (5%)(Use the extra.txt given in the project)

* The map is extended now, and you need to build the map again using the existing class you just built. Define a new class and use vector to create that.
* The data can be read from extra.txt.
* The only difference is that you need to add the judgment process on red square and it is convenient for you to use inheritance method.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| O |  |  |  |  |  |  |  |
|  |  |  |  |  |  | |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

( Hint: using inheritance method to build the class and add a pointer to the private attribute. )

1. Print Out the Map (5%)

* After establishing the new map, you need to print out the new map for this game. You can use “☺”(ASC =1) to represent the player and “\*” to represent the city or place you have .

1. Show the Path (5%)

* After you print out the winner of the game, print out the place’s position that the player has passed and the sequence should be from the end point to the start point.

(Hint: Use the stack structure in STL library)

FAQ:

1. If the player played the card game at a city and could the player buy this city?

The player can also buy the city after or before the card game.

1. If a player’s money goes to negative, is the player able to buy a new city?

The player is not able to buy a city if he have negative money in his pocket.

1. Why I met the error class type redefinition?

#ifndef CITY\_H

#define CITY\_H

......

your class city

......

#endif

1. If you already own a 'bad city' and land on it, do you still lose $500?

Yes, you still need to pay for 500 even you owned it.

1. If another player already owns a 'good city' and you land on it, do you still get to play the mini game?

Yes, the mini game is not influenced by the owner of the city.