RACE OF THE ZODIACS

1 Introduction

Once upon a time, the animal kingdom wanted to choose 12 animals to be the zodiacs. Therefore, a race game to heaven was held. All the animals inside the race zone will be eligible to join the race. After that first race, many race games were held for all the animals. When the race game starts, all the animals will have to try their best to reach the goal that they only know when the game starts. The tracking of the game will end when one of the animals reaches the goal. Your job is to implement a program to track one of the races.

The animals that often appear in the race game are the 12 zodiacs:

Rats, the champion of the first race game ever, oxen, tigers, cats, dragons, snakes, horses, goats, monkeys, roosters, dogs, and boars. Each species can have more than one runner, and each of these species has a different moving strategy.

Default movement: the zodiac will move horizontally or vertically in the direction of the goal. They will prioritize the direction where the distance to the goal is farther. i.e.: an animal in location (2,2) will move horizontally to the goal (5,4) because $|2-5| > |2-4|$; when 2 directions have the same distance to the goal, move horizontally. Unless stated otherwise, the zodiac will be stopped by "Stone" or "Water" and will move to another direction if stand next to the obstacle. And will stand still if both directions are blocked. All animals can only move 1 cell when starting in a "Tornado". The animal will only try to move closer to the goal, if those directions are blocked by obstacles, they are "stuck".

- Rat, Slow movement but can tag along another animal:
  Move 1 cell. When a rat starts moving in the same cell as other animals, stick to the fastest animal and move the same as they do (whoever moves closest to the goal at that turn). When rats are at the same rank as other zodiacs, rats are ranking higher.
- Ox, Strong and even Faster in "Water":
  Move 2 cells at most. Can move into "Water" cell and can move 3 cells when start in a "Water" cell. Destroy "Stone" cell if move to or over it.
- Tiger, King of the forest:
  Move 2 cells at most. When another runner starts in the same cell as tiger, they will be pushed back 2 diagonal cells away from the goal before moving (can only stopped by the edge of the map), rat can't stick to tiger, tiger is not affected by other tiger.
- Cat, Live by themselves:
  Move 2 cells at most. Ignore the same cell reaction from other animals (including the Rat, only the Dog get "Motivated" when Dog and Cat are in the same cell).
- Dragon, Almighty flyer:
  Move 1 cell. Can move diagonally and can only be stopped by "Tornado" (treat "Tornado" as block obstacle).
- Snake, Poisonous slithering:
  Move 2 cells at most. When another runner starts in the same cell as snake, they cannot move for the a turn (including the tiger, snake also pushed back by the tiger).

- Horse, Fastest runner:
  Move 3 cells at most. Can move to or pass a “Stone” cell, however, will be “injured” when move to or over a “Stone” and can only move 1 cell at most after that. Can move pass a “Water” if standing next to it.
- Goat, Just move:
  Move 2 cells at most.
- Monkey, “Tree” jumper:
  Move 2 cells at most. When starting in a “Tree” cell, can move 4 cells that cannot be stopped by anything.
- Rooster, Trying to fly:
  Move 2 cells at most. Can move to a “Stone” (if stand next to the “Stone” and move to “Stone” cell direction can only move 1 cell onto the “Stone”). When starting on “Stone” or “Tree” Move 3 cells at most and can move pass “Stone” or “Water” but not into “Water”.
- Dog, Team player:
  Move 2 cells at most. When another runner starts in the same cell as dogs, +1 Move cell (if there are more than 1 dog in a cell, the dogs also got +1 Move cell).
- Boar, Lawn mower:
  Move 1 cell. Cannot be stopped by obstacle, destroy obstacles (“Water”, “Stone”) when moved on it.

At most 12 zodiacs (of any kind) can join a race. The program needs to announce the movement every turn and ranking the participants at the end of the game (using Manhattan distance, those who has the same distance will belong to the same rank)

2 Requirements

You are required to implement the following classes and functions:

*Note: Student can implement more sub-functions/sub-classes than the requirements for any class or for the program if needed.

2.1 Map Creating

To create the map for the race, student is required to implement class `gameMap` as the following description:

gameMap
string** mapMat Int row Int col

printMap()
~gameMap()//clear the string**
string*& operator[](int i);//overloading the [] operator

The constructor

```
gameMap(string **inputMat,int r,int c)
```

will receive the input:

```
string tempMap[50][50]=
     {   {"","","W","","S"},
         {"","W","","","S"},
         {"W","S","","","S"},
         {"W","S","","","S"},
         {"W","S","","",""}};
int r=5;
int c=5;
string **Map=new string*[r];
for(int i=0;i<r;i++){
    Map[i]=new string[c];
    for(int j=0;j<c;j++){
        Map[i][j]=tempMap[i][j];
    }
}
gameMap gMap(Map,r,c);
```

Function `printMap()` will print the map on console like a table with "|" as horizontal bars and "_" as vertical bars.

The size of a cell is at least 2 "|" height and 7 "_" lengths (see Figure 1).

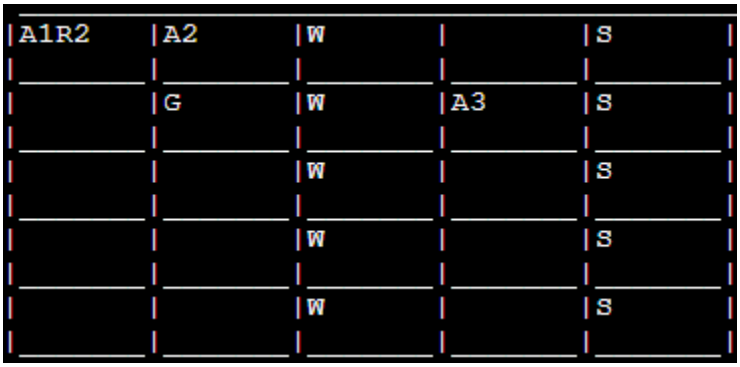


FIGURE 1: EXAMPLE MAP PRINT

In case of a table cell have more than 6 characters, cut the string into the 2nd line. The 1st line will have first 5 characters and a "_" while the rest of the string go to 2nd line (see Figure 2).

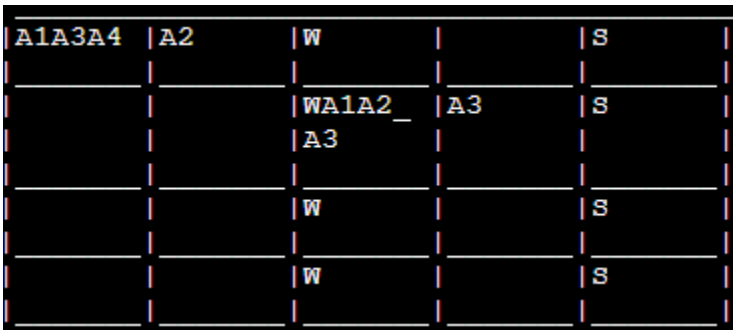


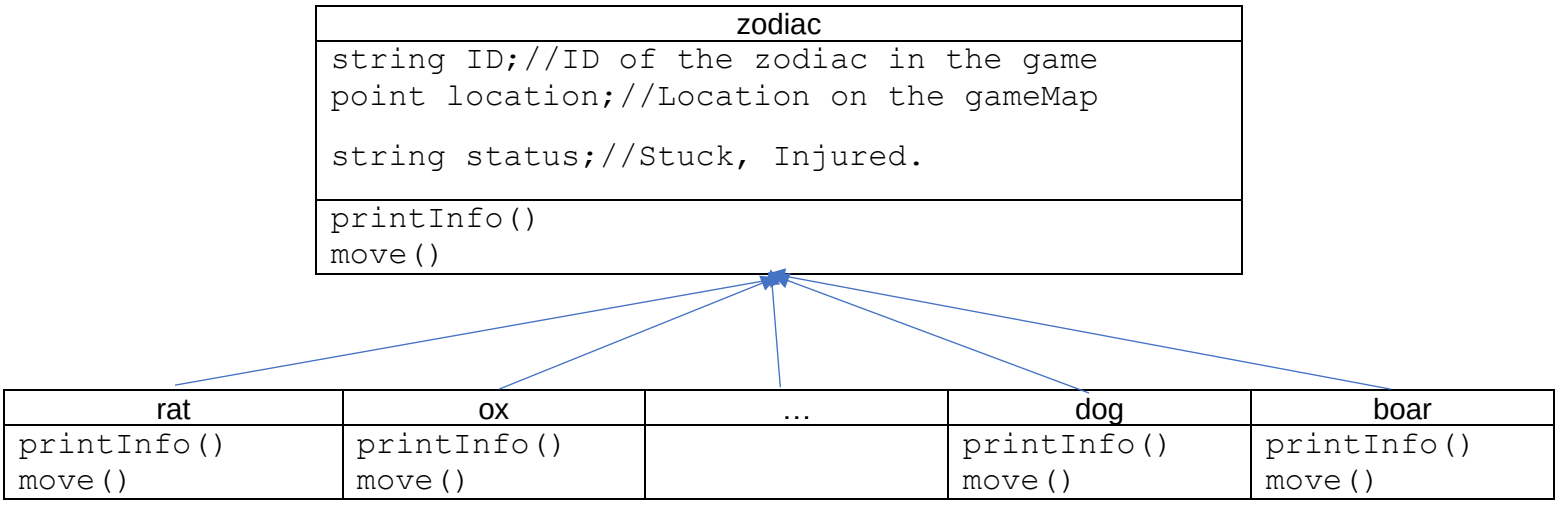
FIGURE 2: EXAMPLE 2-LINE MAP PRINT

*Note: In this assignment the maximum will be 2 lines

2.2 Zodiac – Zodiac list

2.2.1 Class zodiac and Sub-classes

This race among the zodiac has 12 different classes (rat, ox, tiger, cat, dragon, snake, horse, goat, monkey, rooster, dog, boar) all inherited from a base Zodiac class. These classes' structure is described with the following figure:



Class/Struct `point` is a class/struct with int x and int y to interpret the location of a zodiac in the gameMap.

Constructor with 2 inputs for ID and location will be called like:

```
int main(){
    zodiac *pR= new rat("R1",point(2,3));
    rat r("R2",point(3,3));
}
```

Method `printInfo` is a virtual method, print the ID, what kind they are and their current location like:

```
rat R1 at (2,3)
rat R2 at (3,3)
```

Method `move` is a virtual method, move the zodiac according to the introduction. This method receives the goal location and the gameMap to move.

2.2.2 Class zoList

```
add();//add a new zodiac to the list
~zList();//clear the zodiac list
zodiac*& operator[](int
i);//overloading the [] operator
```

The constructor

`zoList()`

Initialize the zList with at most 12 elements.

2.3 Game

Game
`gameMap mapMat;` `zoList zList;`
`void addZo();` `void startGame();`

The constructor

`Game(const gameMap& m)`

copy the m to mapMat;

Function `addZo` to add a zodiac pointer k to the zList, place them on their appropriate location on the map.

Function `startGame` will do the following:

Receive and place the goal on the map.

Receive the bool printMap flag; printMap default is 0.

Step 1: Print the current map with all the zodiacs and “G” for the goal.
Step 2: Move all the zodiacs in zList appropriate to their class. If a zodiac cannot move apply status “Stuck”.
Step 3: If the printMap flag is 1, print the map.
Step 4: If all the zodiacs are “Stuck” or at least one of the zodiacs reached the goal, print congratulate the winner and the ranking list; else, return to Step 2.

The turn printing and congratulation can be see in the example on BKEL site quiz.

3 Points Allocation (Deploy on BKEL theory site):

1. Implement class gameMap as described in section 2.1 (2 points)
2. Implement the class/struct point; constructor, printInfo for all the zodiac as described in section 2.2.1 class (2 points)
3. Implement the zoList class as described in section 2.2.2 (2 points)

4. Implement the Game class and the move of zodiac as described in section 2.3 and section 1 (4 points)
5. Submit your files on the last day of the assignment deadline (The submission only opens on the last day). Without files submission your point will be 0. The file should be named <ID>.cpp, lib.h and lib.cpp where: <ID>.cpp is the main file with only main function and library include. The main function should run a game of 10x10 size map with at least 5 runners of any class you choose (<ID> is your student ID).
   lib.h: classes and functions declarations.
   lib.cpp: functions definition.

*Note: Any cases of copying or letting others copy your work will result in 0 point of the assignment and further discipline if necessary.