Introduction to Computer
Homework 06

2014/05/17 by TA 謝明倫
Due: 2014/06/11

Homework Rules:

Writing homework should be **brought to class** and handed in **before lecture starts**.
If you would late, please bring the writing homework to room BL421 for TA.
(Better to send a mail to TA first in the case that there is no one in room BL421)

As for **programming homework**, you should **upload** it to our course in CEIBA.
Uploading deadline would be the coming midnight at **3:00 am**.
The file you upload must be a .zip file that contains the following files:

README.txt,
HW06_b02901XXX (a folder that contains the .cpp & .h files required),

1. Do not submit executable files (.exe) or files for linker(.o, .obj). Files with names in wrong format will not be graded. You must **remove any system functions,** such as system ("pause"), in your code if you use it.
2. In README.txt file, you need to describe which compiler you choose in this homework and how to compile it (if it is in a "project" form).
3. In your .cpp files, we suggest you write comments as detailed as you can. If the code does not work properly, code with comments can get partial points. It will be good for the TAs to read your code as well as for your future reference and maintenance.

Review Problems (50%)

Chapter 11: Problems 31, 43, 55.
Chapter 12: Problems 4, 15, 30, 44, 45.

Programming Problem (50%)

1. Little Tank War AI

Have you ever play some games like “Tank Wars” with your classmates before? In this homework, we are going to implement an artificial intelligence agent to compete with each other!! Here we **already have a “Little Tank War” game** written in C++. All the tanks are controlled by the class “PolicyMaker”. What you need to do is to **inherit it with your own strategy**. Then enjoy having fun with your friends.

What to do?

(1) Build and run the “LittleTank” project with “Curses” library. **Make sure your environment is well prepared**.

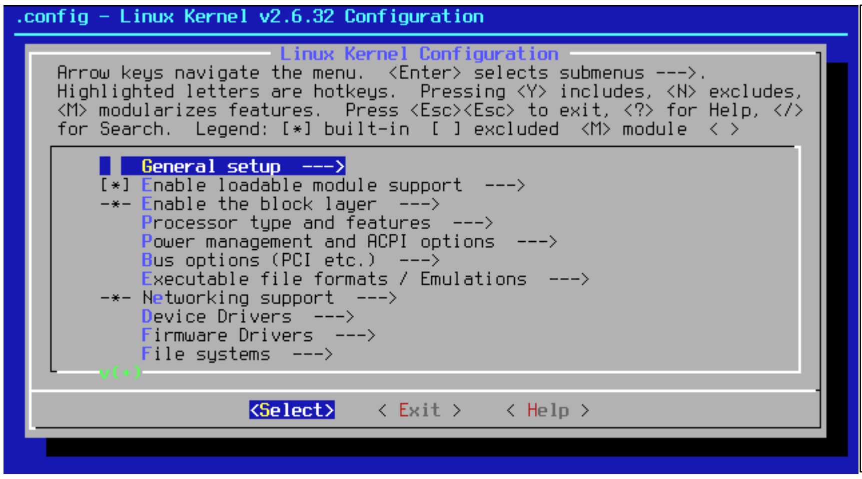
(2) Have some trial, then be more familiar with the rules and game structure.

(3) Modify the file “b02901xxx.h” into your own version. Implement your agent.

(4) Register it in the class “AgentsMgr”. Set your ID for the game and have fun.

How to start? (Project Build and Curses Library)

This game uses the “Curses” library for display controlling. Curses is a terminal control library for the construction of text user interface (TUI) applications.



```
.config - Linux Kernel v2.6.32 Configuration
                       Linux Kernel Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus --->.
  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes,
  <M> modularizes features.  Press <Esc><Esc> to exit, <?> for Help, </>
  for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < >

        General setup  --->
    [*] Enable loadable module support  --->
    -*- Enable the block layer  --->
        Processor type and features  --->
        Power management and ACPI options  --->
        Bus options (PCI etc.)  --->
        Executable file formats / Emulations  --->
    -*- Networking support  --->
        Device Drivers  --->
        Firmware Drivers  --->
        File systems  --->
    v(+)

                <Select>    < Exit >    < Help >
```

← A curses TUI example.

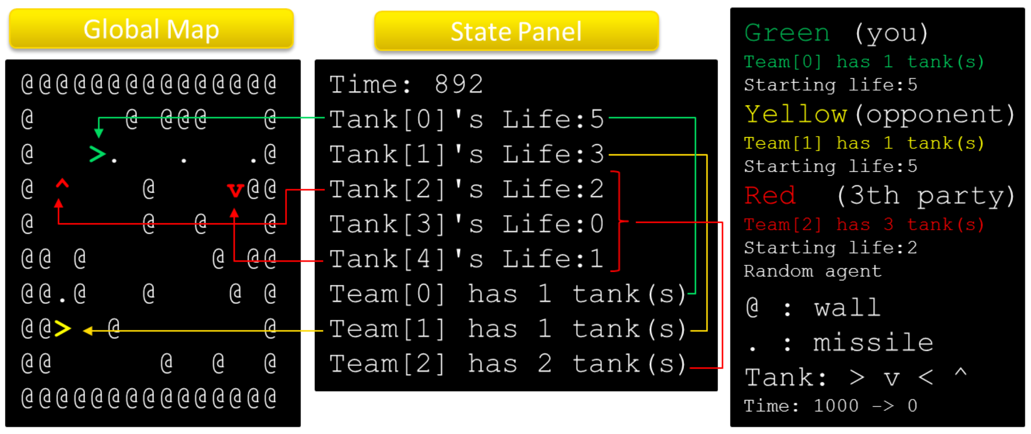
It’s a cross-platform library for C.

For more detail, you can refer to: http://en.wikipedia.org/wiki/Curses_(programming_library)

“Curses” is a cross-platform library, but does not build-in in ANSI C++. You need to download/install it first. For Windows users, we advise you using the “PDCurses” library (It is the light weight version). And we already pack a **Windows** version PDCurses for you in the homework project file. **Just open the Code::Blocks project and build it**. As for **Linux or Mac users, we advise you using the “NCurses”** library, which is maintained by GNU. Both two libraries serve the same in our project. (But you need to download/install NCurses by yourself.)

First glance at the game

First time you build and the LittleTank project, you would play it in the "humanGame" mode (see main.cpp). You are controlling (↑ ↓ ←→ button) the green tank fighting (Space as firing) with other tanks. Here is an example:



```
@@@@@@@@@@@@@@@
@     @ @@@   @
@   >.   .   .@
@ ^    @    v@@
@      @  @   @
@@ @       @ @@
@@.@   @    @ @
@@>  @        @
@@      @  @  @
@@@@@@@@@@@@@@@
```

```
Time: 892
Tank[0]'s Life:5
Tank[1]'s Life:3
Tank[2]'s Life:2
Tank[3]'s Life:0
Tank[4]'s Life:1
Team[0] has 1 tank(s)
Team[1] has 1 tank(s)
Team[2] has 2 tank(s)
```

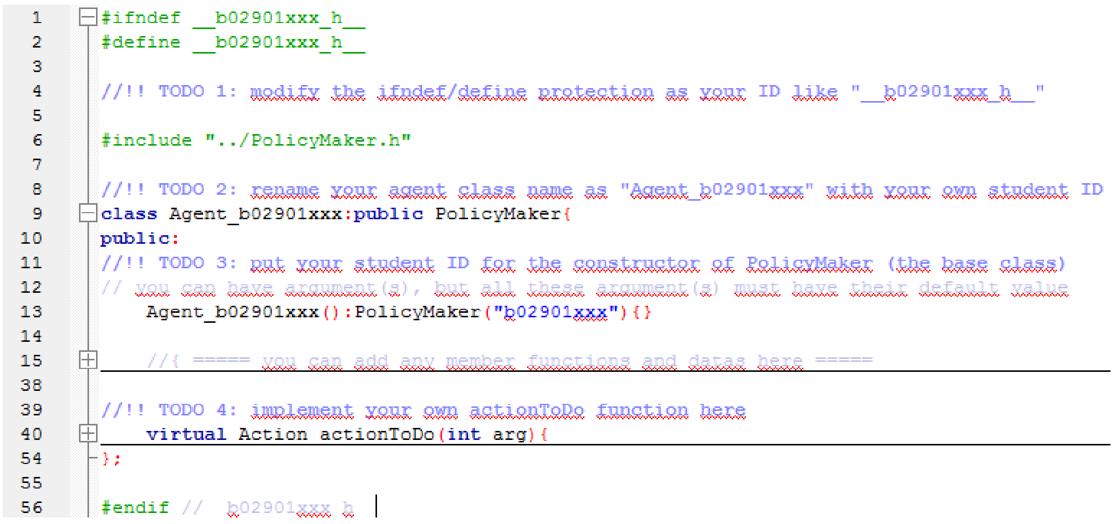
```
Green  (you)
Team[0] has 1 tank(s)
Starting life:5
Yellow(opponent)
Team[1] has 1 tank(s)
Starting life:5
Red    (3th party)
Team[2] has 3 tank(s)
Starting life:2
Random agent
@ : wall
. : missile
Tank: > v < ^
Time: 1000 -> 0
```

The game points is calculating by: (lose: 0, match: 400, win: 900+time remain)

```
tank.isDead() ? 0: (timer<=0)?400:(900+timer);
```

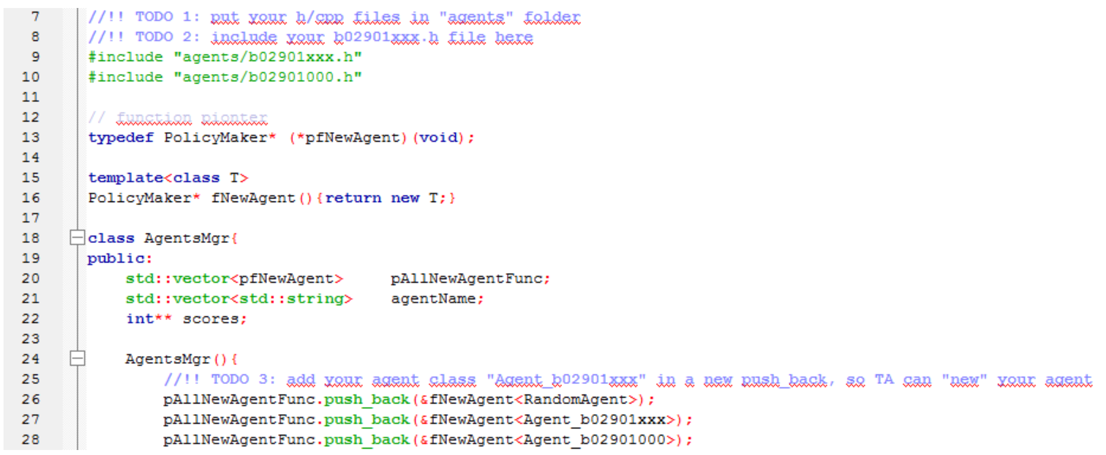
Implement your agent:

Write your AI agent in the file "b02901xxx.h/.cpp" (**rename it as your student ID**) :



```
#ifndef __b02901xxx_h__
#define __b02901xxx_h__

//!! TODO 1: modify the ifndef/define protection as your ID like "__b02901xxx_h__"

#include "../PolicyMaker.h"

//!! TODO 2: rename your agent class name as "Agent_b02901xxx" with your own student ID
class Agent_b02901xxx:public PolicyMaker{
public:
//!! TODO 3: put your student ID for the constructor of PolicyMaker (the base class)
// you can have argument(s), but all these argument(s) must have their default value
    Agent_b02901xxx():PolicyMaker("b02901xxx"){}

    //{ ===== you can add any member functions and datas here =====

//!! TODO 4: implement your own actionToDo function here
    virtual Action actionToDo(int arg){
};

#endif // __b02901xxx_h__
```

Your agent's "actionToDo" function would be called every cycle, you need to return either "noAct", "U_Act"(press ↑), " D_Act"(↓), "L_Act"(←), "R_Act"(→), or "fire". You can get the game information by calling the "getView", "getMissileInView", and "getTankInView" functions inherited from "PolicyMaker". Then, register it in the file "AgentsMgr.h" and remember your agent ID (as the order of push_back()). Last, change the game mode to "singleGame" in main.cpp with the agent ID.



```
//!! TODO 1: put your h/cpp files in "agents" folder
//!! TODO 2: include your b02901xxx.h file here
#include "agents/b02901xxx.h"
#include "agents/b02901000.h"

// function piontер
typedef PolicyMaker* (*pfNewAgent)(void);

template<class T>
PolicyMaker* fNewAgent(){return new T;}

class AgentsMgr{
public:
    std::vector<pfNewAgent>     pAllNewAgentFunc;
    std::vector<std::string>    agentName;
    int** scores;

    AgentsMgr(){
        //!! TODO 3: add your agent class "Agent_b02901xxx" in a new push_back, so TA can "new" your agent
        pAllNewAgentFunc.push_back(&fNewAgent<RandomAgent>);
        pAllNewAgentFunc.push_back(&fNewAgent<Agent_b02901xxx>);
        pAllNewAgentFunc.push_back(&fNewAgent<Agent_b02901000>);
```

(↑ Just rename all the b02901xxx as your student ID is OK. Then your agent ID is 1.)

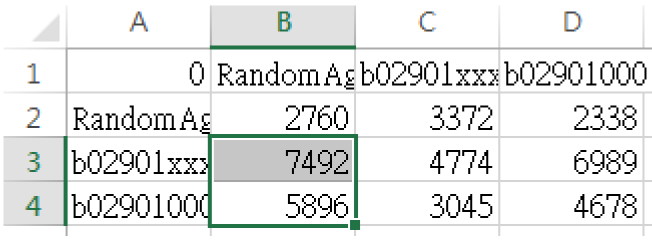
```
int main(){
    Game game;
    game. singleGame(time(0),0,1);
    // parameters:
    // randSeed=4,GreenAgent=0,YellowAgent=1,bool showGame=true
    return 0;
}
```

[Hint] For printing debugging message, you can make use of (TA's) std::string DebuggingMessage, which would be printed automatically by game every cycle.

Grading:

For basic part (50%), your agent needs to get more points than the "Agent_b02901000" does when competing with "RandomAgent", that is, pointSum(You, "RandomAgent") - pointSum("Agent_b02901000", "RandomAgent") >= 250 over 5 games (Game random seed: 4, 9, 80, 7352, 8632). Then you can get the full points.

(that is, in "battleAll" mode, B3-B4>=500)



	A	B	C	D
1	0	RandomAg	b02901xxx	b02901000
2	RandomAg	2760	3372	2338
3	b02901xxx	7492	4774	6989
4	b0290100(	5896	3045	4678

"battleAll" mode plays the game 4, 9, 80, 7352, 8632 and it's reverse (change the Green/Yellow), total 10 games.

Submitting:

Just submit the "b02901xxx.h/.cpp" files (the file name must be "your" student ID). Your code must be compatible with our project, and obeys the following rules.

Important rules:

You **MUST follow these coding rules**:

(1) You can **NOT use const_cast** (or any thing like that) to "set" variables or use non-constant member functions of current tanks.

(2) Do **NOT use any global** variable, global function. Just put them in your class.

(3) Also, you can **NOT use static data member**. Memory across different games are not allowed.

(4) You can write more than one agent in "b02901xxx.h/.cpp", is OK. But **ONLY the class named "Agent_b02901xxx" would be taken for grading**. And other classes must be named as "Agent_b02901xxx_xx…xx" in the case not to conflict with other students.

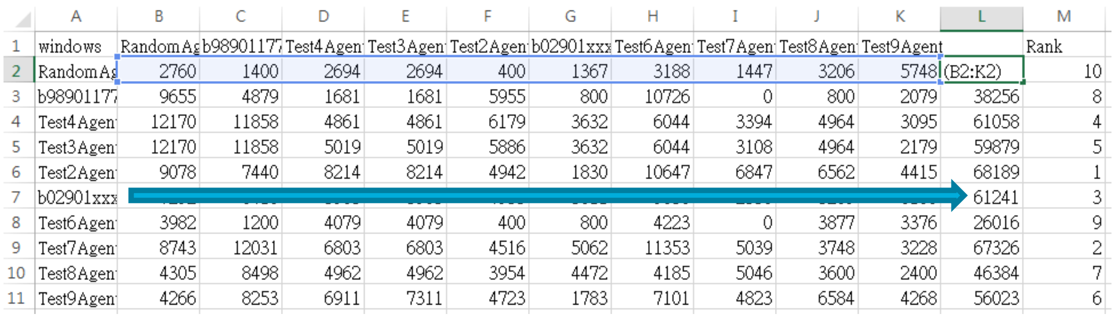
(5) You can NOT use your own #define or macro function in "b02901xxx.h" file. Put them in "b02901xxx.cpp" is OK.

2. [Bonus] 5% Competing with the whole class

TA will run the "battleAll" mode with all students' work. That will be a big competition. You will receive the bonus points as your rank:

Highest	10%	: 5
	10~20%	: 4
	20~30%	: 3
	30~50%	: 2
	50~70%	: 1

The ranking is based on the summation of every game your agent have gone through.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	windows	RandomAg	b98901177	Test4Agen	Test3Agen	Test2Agen	b02901xxx	Test6Agen	Test7Agen	Test8Agen	Test9Agent		Rank
2	RandomAg	2760	1400	2694	2694	400	1367	3188	1447	3206	5748	(B2:K2)	10
3	b98901177	9655	4879	1681	1681	5955	800	10726	0	800	2079	38256	8
4	Test4Agen	12170	11858	4861	4861	6179	3632	6044	3394	4964	3095	61058	4
5	Test3Agen	12170	11858	5019	5019	5886	3632	6044	3108	4964	2179	59879	5
6	Test2Agen	9078	7440	8214	8214	4942	1830	10647	6847	6562	4415	68189	1
7	b02901xxx	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	61241	3
8	Test6Agen	3982	1200	4079	4079	400	800	4223	0	3877	3376	26016	9
9	Test7Agen	8743	12031	6803	6803	4516	5062	11353	5039	3748	3228	67326	2
10	Test8Agen	4305	8498	4962	4962	3954	4472	4185	5046	3600	2400	46384	7
11	Test9Agen	4266	8253	6911	7311	4723	1783	7101	4823	6584	4268	56023	6

[Hint] If you and your classmate(s) want to battle, you can link each other by putting agents into the same project and register them in "AgentsMgr".