2011 Winter Entrance Examination

Department of Creative Informatics
Graduate School of Information Science and Technology
The University of Tokyo

Programming

INSTRUCTIONS

1. Do not open this problem brochure until the signal to begin is given.
2. Write your examinee ID below on this cover.
3. A draft sheet is provided. Write down your examinee ID on the sheet.
4. You may choose your favorite programming languages.
5. You may consult only one printed manual of a programming language during the examination. *You can use or copy any libraries or program segments existing in your PC, but you cannot connect to the Internet.*
6. By the end of the examination, make a directory/folder on your PC, whose name is the same as your examinee ID, and put your program files and related files into the directory/folder. Copy the directory/folder onto the delivered USB memory.
7. At the end of the examination, the USB memory and the draft sheet are collected.
8. After these are collected, stay at your seat, until all examinee program results have been checked briefly by the test supervisor.
9. After the brief check, try to save your program execution environment on the PC so that you can run your program as soon as possible during the oral examination in the afternoon.
10. *Leave your PC and this brochure together in the room* for the oral examination and leave the room until you are called.

Examinee ID _____

This page is empty.

This page is empty.

Make a program that plays Tic-tac-toe (sometimes called Noughts and Crosses) by following steps:

[Rules of Tic-tac-toe] Tic-tac-toe is a two player game. The size of the board is 3 by 3. One player places ○ and the other player places ×. Assume that ○ is the first move and × is the second move, and so on.

Two players alternately put ○ or × on one of empty spaces in the board. As shown in Figure 1, the player who succeeds in placing three respective marks in a horizontal, vertical, or diagonal row wins the game. When all the places are occupied and there is no winner or loser, it is a draw.

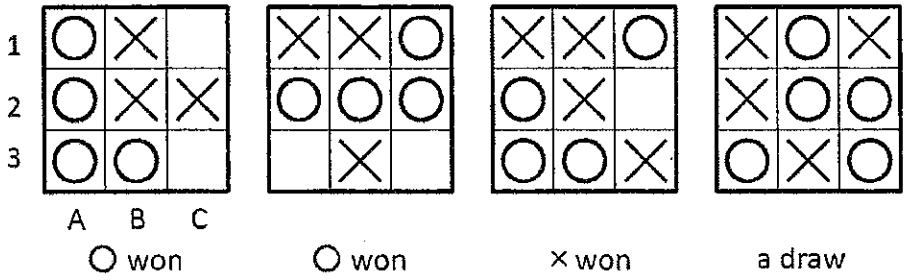


Figure 1. Examples of the board

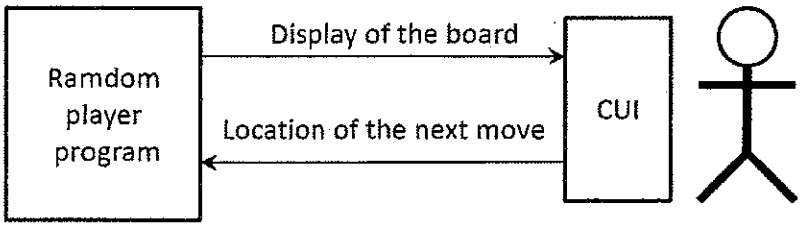
Q1) Make a data-structure for a board and make a program that visualizes the current state of the board. The visualization of the board must be realized by 3-row 3-column ASCII characters (o, x and -). Make a program that creates a pattern of the board as a test program. Show the four cases shown in Figure 1.

```
OX-
OXX
OO-
```

Example of the output (Corresponding to the left of Figure 1.)

Q2) Make a program that receives the location of the new move, updates and displays the board. The program is capable of automatically distinguishing whether the next move is ○ or ×. Then, when the new move results in a win, loss or draw, display that result and terminate the program. The input format of the location of the new move is 2 ASCII characters that specify the location. For example, the upper right location of the left board in Figure1 is specified as 1C.

Q3) Make a program that places the new move on the location randomly selected from empty locations of the current board, updates the board and outputs the newly inserted location for playing with human co-player as shown in Figure 2. When the new move results in a win, loss or draw, display that result and terminate the program. Assume that the human co-player moves first and the program moves second, and so on.



CUI : Character-based user interface

Figure 2. Player program for computer-human playing

Q4) Make an automatically playing program for Tic-tac-toe which incorporates two part taken from Q2 and Q3 shown above (Figure 3). During the automatic playing, insert about one second wait between each move.

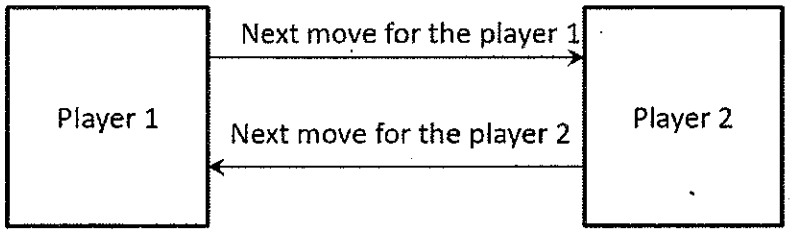


Figure 3. Automatically playing two programs

Q5) Make a program that calculates the final outcome of putting ○ or × on a certain location. The final outcome is either "definite win", "definite loss", "definite draw" or "no definite outcome".

Q6) In the program written in Q3, modify the selection method of the location to place next move to improve strength of the playing program.

Q7) Modify the answer of Q4 to utilize inter-process communication of two processes for play. If the answer of Q4 already utilizes inter-process communication of two processes for play, the answer may be the same program as Q4.

This page is empty.

This page is empty.