**10902 CPP Midterm Exam**

|  |
| --- |
| **Contributor︰SHANG-YAN XUN** |
| **Subject：Ghost Leg** |
| **Main testing concept：**   |  |  | | --- | --- | | **Basics** | **Functions** | | ■ C++ BASICS  ■ FLOW OF CONTROL  ■ FUNCTION BASICS  □ PARAMETERS AND OVERLOADING  ■ ARRAYS  □ STRUCTURES AND CLASSES  □ CONSTRUCTORS AND OTHER TOOLS  □ OPERATOR OVERLOADING, FRIENDS, AND REFERENCES  □ STRINGS  □ POINTERS AND DYNAMIC ARRAYS | □ SEPARATE COMPILATION AND NAMESPACES  □ STREAMS AND FILE I/O  □ RECURSION  □ INHERITANCE  □ POLYMORPHISM AND VIRTUAL FUNCTIONS  □ TEMPLATES  □ LINKED DATA STRUCTURES  □ EXCEPTION HANDLING  □ STANDARD TEMPLATE LIBRARY  □ PATTERNS AND UML | |
| **Description：**  Ghost Leg is a method of lottery designed to create random pairings between two sets of any number of things, as long as the number of elements in each set is the same.  It consists of vertical lines with horizontal lines connecting two adjacent vertical lines scattered randomly along the lines; the horizontal lines are called "legs".  The rule for playing this game is:  Choose a line on the top, and follow this line downwards. When a horizontal line is encountered, follow it to get to another vertical line and continue downwards. Repeat this procedure until reaching the end of the vertical line.    **Input：**  Given the number of vertical lines and the number of "legs".  Given a sequence of indices where the "legs" should be placed. For example, index ‘0’ means the "leg" is placed between the 0th and the 1st vertical line. Note that the indices are sorted by the height of the "leg".  Given an index of the vertical line to start playing.  **Output：**  Print the index of the vertical line you reached at the end of each game.  **Sample Input / Output :**   |  |  | | --- | --- | | **Sample Input** | **Sample Output** | | 4 8  1 0 2 1 1 0 1 2  1 | 2 | |
| **■** **Easy, only basic programming syntax and structure are required.**  **□ Medium, multiple programming grammars and structures are required.**  **□ Hard, need to use multiple program structures or complex data types.** |
| **Expected solving time:**  10 minutes |
| **Other notes:** |