2009 Summer 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 and the delivered envelope until the signal to begin is given. The brochure contains a printed working sheet.
2. Write your examinee ID below on this cover.
3. An answer sheet and a draft sheet accompany this brochure. Write down your examinee ID on both sheets. There are questions to be answered on the answer sheet.
4. The USB memory delivered beforehand to each examinee contains seven ASCII text files: `init-state.txt`, `rotseq.txt`, `data1.txt`, ..., `data5.txt`. Newline is represented by carriage return (CR) followed by linefeed (LF) in these files.

   Before examination starts, copy these files to your PC and browse them. Make sure you can see lines of single alphabetic characters separated by blanks or lines of two alphanumeric characters separated by blanks in these files, and then keep your hands away from your PC. If you cannot read the file properly, consult the test supervisor. The contents of the USB memory are common to all examinees.
5. You may choose your favorite programming language.
6. You may consult only one printed manual of the programming language in the examination. **You can use or copy any libraries or program segments existing in your PC, but you cannot connect to the Internet.**
7. 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.
8. At the end of the examination, the USB memory, the answer sheet, draft sheet and working sheet are collected. You can take the gadget in the envelope.
9. After these are collected, stay at your seat, until all examinee program results have been checked briefly by the test supervisor.
10. 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.
11. **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 is a blank page.

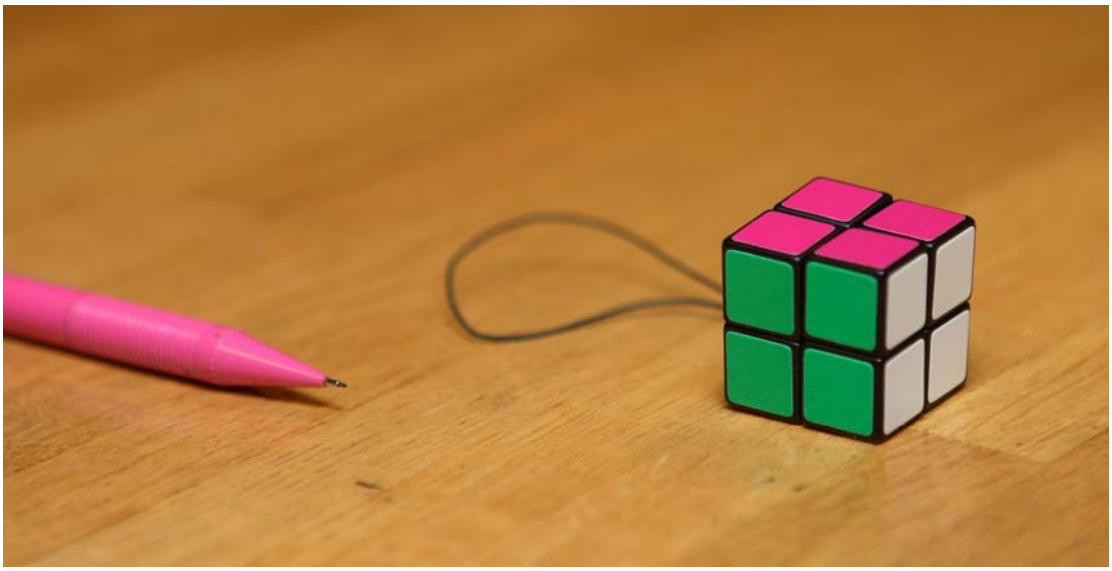

Fig.1　2×2×2 Rubik Cube

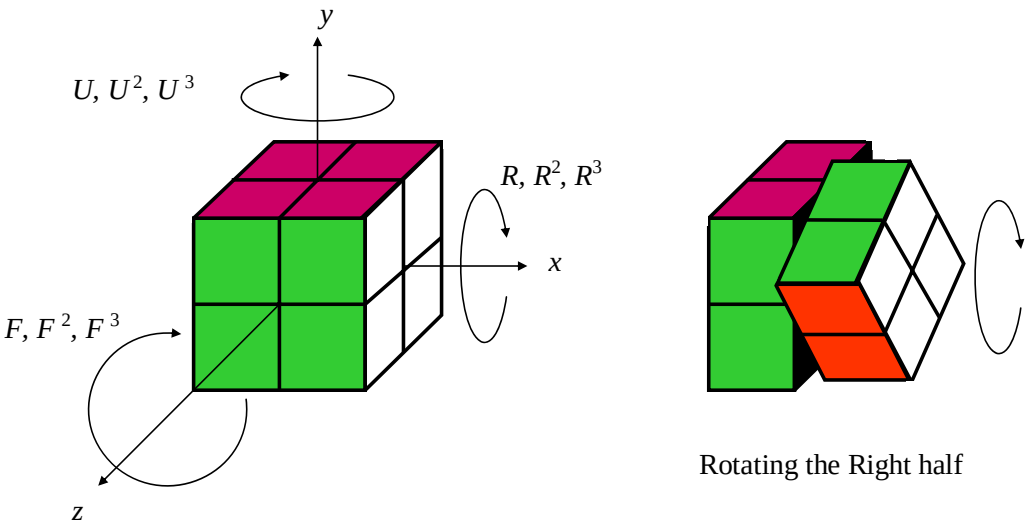

Fig.2　Rotation of the Cube

Consider a Rubik cube, or simply *cube*, included in the delivered envelope (Figure 1). This is a $2 \times 2 \times 2$ cube, whose unit square faces (*facelets*) are colored in pink (p), white (w), green (g), red (r), yellow (y), and blue (b). Each of the eight pieces which comprise the cube is called a *cubicle*. We can rotate any $2 \times 2 \times 1$ part (consisting of four cubicles) of the cube around the center of the 2×2 face (Figure 2). After a few rotations, we can get a pattern of mixed colors on the cube faces. We say the cube is in the *initial state* if each face has the same color and it is placed in the orientation as shown in Figure 1.

The *cube state* is defined as an *opened-up cube* which shows the colors of the 24 facelets of the cube. Figure 3 indicates the initial state. In the opened-up cube, from the upper left to the lower right, six 2×2 facelet arrays for the *Up* face, *Right* face, *Front* face, *Down* face, *Left* face, and *Back* faces are aligned. In Figure 3, we index the same color facelets, say, p_1, p_2, p_3, p_4 clockwise to distinguish them. We use this opened-up cube notation hereafter. Use five sheets of the opened-up cube delivered in the envelope. And if necessary, also use the opened-up cube templates printed on the working sheet.

Let us consider the rotations which rotate the *Up* half, *Right* half or *Front* half of the cube clockwise by 90 degrees, 180 degrees or 270 degrees (or counter-clockwise 90 degrees), while placing the cube on the table as shown in Figure 2. These rotations are described as U, U^2, U^3, R, R^2, R^3, F, F^2, F^3, respectively. There is a cubicle which does not move by these rotations. We say this cubicle is located at the *U-R-F invariant position*.

$$
\begin{array}{cccccccc}
p_1 & p_2 & w_1 & w_2 & & & & \\
p_4 & p_3 & w_4 & w_3 & & & & \\
 & & g_1 & g_2 & r_1 & r_2 & & \\
 & & g_4 & g_3 & r_4 & r_3 & & \\
 & & & & y_1 & y_2 & b_1 & b_2 \\
 & & & & y_4 & y_3 & b_4 & b_3
\end{array}
$$

Fig.3 A cube state with color indeces for the initial state

We call the rotation of the whole cube on the table a *replacement*. To solve the puzzle to return an arbitrarily placed cube back to the initial state, at first we replace the cube so that the cubicle which should be located at the *U-R-F* invariant position in the initial state will be located at the *U-R-F* invariant position in the same orientation. To solve the puzzle, we need only one replacement at first, and then iterate the rotations mentioned above. We need not consider the rotations of the *Down* half, *Left* half and *Back* half: D, L, B and their derivatives.

If two cube states become equal by some replacement, we call the cube sates *equivalent*.

Q1 Write on the answer sheet giving a brief explanation, the number of the cube states equivalent to the initial state, including the initial state itself.

Q2 You applied the rotation U to the initial state. Write the obtained cube state on the answer sheet. Write the color index for this question, and circle the facelets of the cubicle at the *U-R-F* invariant position.

Q3 Let us consider the replacement first. We can represent a replacement as a permutation (or rearrangement) of the 24 facelets written in a cube state. The replacement can be composed by a permutation of the 6 faces U, R, F, D, L and B, and cyclic permutations of the 4 facelets on individual faces. Let us consider here only the permutation of the 6 faces.

A replacement of the cube where the original face Φ_1 is placed at the *Up* face, and the original face Φ_2 is placed at the *Right* face is described as $\Phi_1\Phi_2$. For example, if we replace the cube so that the original *Left* face becomes the *Up* face, and the original *Back* face becomes the *Right* face, we describe the replacement LB. Note that UR does nothing.

Let us compose any replacement of the cube only by the two replacements FR and UB. Both FR and UB are the clockwise entire cube rotations by 90 degrees, with FR around the x-axis, and UB around the y-axis. Axes are shown in Figure 2. You may answer the following questions either with or without writing a program.

(3-1) Write on the answer sheet the compositions of the replacements UL, FU, RU, BU, and LD, only by using FR and UB. For each replacement, there may be more than one possible composition, but you need write only one. A composition should be written in a sequence of replacements separated by blanks. For example, RB can be composed by doing UB and then FR, the composition is written as UB FR.

(3-2) Write on the answer sheet the inverse of every replacement except for UR, where the *inverse* means that if it is composed with the original replacement the result will be UR. Write each replacement and its inverse as a pair.

Q4 We want to write a program that calculates the final cube state obtained by applying a given rotation sequence to a given cube state. The cube state is input and should be output in the opened-up cube format as shown in Figure 3. We do not care about the color indeces hereafter. The initial state of the cube is written in the file `init-state.txt`.

A *rotation sequence* is an arbitrary number of rotations written on the same line, each separated by a blank. In the file, we describe the rotations, say, R, R^2, R^3 as `R1`, `R2`, `R3`, respectively. The following rotation sequence of length 3 is the first of those written in the file `rotseq.txt`. Note that `U3` and `F2` are considered as single rotations.

```
R1 U3 F2
```

(4-1) The rotation of the cube can be represented by a permutation of the 24 facelets in the cube state. Write a short design document either in your program or on the answer sheet, which describes how you design the data structures for the cube state and the permutation representation in your program.

(4-2) Based upon the data structures and permutation representation described above, write a program and output the cube state obtained by applying the rotation U to the initial state. Then, output the results for the cases applying the rotations R and F to the initial state, respectively.

(4-3) Output the obtained cube states respectively for the cases applying the four rotation sequences written in the file `rotseq.txt`, each to the initial state.

Q5 Write a program that outputs a shortest rotation sequence that returns a given cube state back to the initial state, where the given state has already been replaced correctly. Apply the program to the five (already correctly replaced) cube states, each in `data1.txt`, ..., `data5.txt`. The input/output format is the same as that of Q4. The given states in the files *require rotation sequences shorter than or equal to 6*, so that they do not consume too many computational resources.