Quiz Section for Program Design (II)

Exercise #10

This week, we're going to build a **2-bit counter table**. Sometimes, we will need a table to record users‘ usage, such as the number of log-ins. The most familiar method is to use **int type table (array)** to record. However, if the number of record is extremely small, using an int **type table** to record it will waste lots of space.

Today, let’s assume that we want to check if the user has logged in more than 3 times or not. It can be achieved by using only **2-bit (00, 01, 10, 11)**, but unfortunately, the smallest declared size of the types supported by the C language is 1 byte (char type, boolean type).

Therefore, if you want to implement the 2-bit recording, you need to use the technique of low-level programming to split the 1-byte type (or other sizes) into a **2-bit** group to record the number of log-in times.

For example, as shown below, a 1-byte char type can be divided into 8 bits. In this byte, if we treat the rightmost bit as the 0 bit, the bit field containing 0 and 1 bits (i.e., (0,1) bit) can represent the log-in times of the user 0. The (2,3) bit can represent the log-in times of the user 1; the (4,5) bit can represent that of the user 2, etc. The figure below is the simple concept of the 2-bit table:

7	6	5	4	3	2	1	0
1	1	1	0	0	1	0	0
User 3 (login 3 times)		User 2 (login 2 times)		User 2 (login 1 times)		User 1 (login 0 times)	

Your task is to build a **2-bit table.** First, you need to find the bit field of the specific user in the table. Second, you need to check whether the user logs in more than 3 times or not. If the user logs in less than 3 times, you need to print the message “PASS” as the output. Otherwise, you need to print “GOOD BYE.”

Notice: the teaching assistants will check **whether your declared size of the 2-bit table exceeds the limit or not**. For example, if the number of a user is 80 (n = 80), **you can only declare 10 bytes to build the 2-bit table. You must follow the rule. Otherwise, you will get no score.**

The table below shows the example input and output. The integer **n** in the first line represents the number of the users (**n** must be divisible by 4). After the first line, we will input the User ID which represents login until EOF. The value of User ID is between **0 to n – 1.**

Input	Output
4 1 2 1 1 3 1 3	PASS PASS PASS PASS PASS GOOD BYE PASS