Problem A
Access Denied
Time limit: 2 seconds

Computer passwords have been around for a long time. In fact, 60 years ago CTSS was one of the first computers with a password. The implementation of this was very simple. In CTSS the password was stored in plain text in a file on disk. When logging in, the user would enter a password. The computer would then compare the password to the password on disk. If the comparison failed, it would deny access, if it succeeded, access would be allowed. Researchers at MIT were quick to discover several security flaws in this password system. We will explore one of them, the timing attack.



IBM 7090 console (Public Domain)

In a timing attack, we exploit that we can deduce a computation path from the time it takes to do the computation. In CTSS the password check was done using a simple string matching algorithm, similar to this:

```
bool CheckPassword(string pwd1, string pwd2) {
    if (pwd1.Length != pwd2.Length) {
        return false;
    }
    for (int i = 0; i < pwd1.Length; i++) {
        if (pwd1[i] != pwd2[i]) {
            return false;
        }
    }
    return true;
}
```

For the purpose of this problem, we will use a (very) simplified timing model and the above algorithm. The timing model looks as follows:

- A branching statement (if or for) takes 1 ms.
- An assignment, or update of a memory address takes 1 ms.
- A comparison between two memory addresses takes 3 ms.
- A return statement takes 1 ms.

The password consists of between 1 and 20 English letters, upper or lower case, and digits.

Interaction

This is an interactive problem. Your submission will be run against an *interactor*, which reads the standard output of your submission and writes to the standard input of your submission. This interaction needs to follow a specific protocol:

- Your program first sends a password string, consisting of between 1 and 20 English letters, upper or lower case, and digits.

- Depending on if the password is correct, the interactor then responds with either:
  - If the password is correct; "`ACCESS GRANTED`". Your program should then exit cleanly.
  - If the password is incorrect; "`ACCESS DENIED (`t `ms)`", where t is the time it took to verify the password in ms. Your program can then make another guess.

Make sure you flush the buffer after each write. You can guess at most $2\,500$ times. A testing tool is provided to help you develop your solution.

Read	Sample Interaction 1	Write
		`A`
`ACCESS DENIED (5 ms)`		
		`HunFhun`
`ACCESS DENIED (41 ms)`		
		`Hunter1`
`ACCESS DENIED (68 ms)`		
		`Hunter2`
`ACCESS GRANTED`		