DISASSEMBLER

Write a program which does the disassembly process i.e. functions exactly opposite to Assembler.

Input Format

Input is provided through STDIN. Each line is a 16-bit HACK binary code. No Indentation, No comments **NOTE:** Input ends with new-line character. Eg:

```
0000000000001111 // line
          // new-line
```

Constraints

- Firstly, there are 3 Sample test cases (visible) so that you can *RUN* and debug your program. These are worth ***0 Points*** in total.
- Then, there are 3 Graded test cases (hidden) on which your program will be evaluated after clicking *SUBMIT*. These are worth ***30 Points*** in total.
- Finally, there are 3 Bonus test cases (hidden) to check syntax errors (if any) and your program will be evaluated on these bonus tests cases after clicking *SUBMIT*. These are worth ***4 Points*** in total. We are looking to detect only two kinds of syntax errors (if any) as follows. As soon as your program detects either of the syntax error first, it must immediately exit the program with the corresponding String type error message on STDOUT.

ERROR: illegal HACK instruction

ERROR: instruction bit size not 16

Output Format

1. Output must be *symbol-less* assembly statements as per HACK assembly reference sheet.
2. Recall, that your disassembler program has to output explicit addresses rather than displaying symbols. Why? Because, symbols is simply programming gimmick which we use to make our asm readable. For disassembler, it really has nothing to do with symbols in the reverse translation.

3. Output must be written onto STDOUT

Sample Input 0

```
0000000000000001
0000000000010000
```

Sample Output 0

```
@1
@16
```

Sample Input 1

```
0000000000000110
1110110000010000
0000000000000010
1110101010000111
```

Sample Output 1

```
@6
D=A
@2
0;JMP
```

Sample Input 2

```
0000000000010100
1110011111010001
```

Sample Output 2

```
@20
D=D+1;JGT
```