

# Problem F

## A language for constants

Input file: yacl.in  
Output file: yacl.out

An obscure computer science professor wants to become famous developing a new programming language YACL (“Yet Another Constant Language”). This language is very simple; it only has four constructs:

**C+1** creates the constant 1.

**C-1** creates the constant  $-1$ .

**INCR** adds 1 to the constant being generated.

**DBL** multiplies the constant being generated by 2.

A program in this language is a sequence of these constructs, one on each line, executed sequentially. The professor wants to keep the programs in this language simple, small, and fast. To achieve his goal, he adds the following constraints:

- Every program must begin either with **C+1** or with **C-1**.
- A given constant  $C$  must be generated with the smallest number of instructions possible.
- If a constant  $C$  can be generated by different programs (all with equal number of instructions), then the fastest program should be used. For this purpose, suppose that the instruction **DBL** is executed in  $T$  nanoseconds and the instruction **INCR** in  $2T$  nanoseconds.

You were hired by the professor to write several sample programs, so that he can use them at various conferences to demonstrate his powerful new language. The professor will give you a few constants, and your task is to write programs to generate these constants, obeying the constraints above.

### Input specifications

The input file may contain several instances of the problem. Each instance of the problem is just one line containing the numeric constant to be generated. All numbers are non-zero integers between  $-32768$  and  $32767$ . A line containing the integer zero terminates the input file.

## Output specifications

For each instance of the problem, your program should print one line saying "Constant  $n$ ", where  $n$  is the constant for that instance, followed by the most efficient program to generate that constant, one instruction per line. Insert a blank line between each instance.

## Sample input

```
3
-5
1
7
0
```

## Output for sample input

```
Constant 3
C+1
DBL
INCR
```

```
Constant -5
C-1
DBL
DBL
INCR
DBL
INCR
```

```
Constant 1
C+1
```

```
Constant 7
C+1
DBL
INCR
DBL
INCR
```