A. Petya and Java

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Little Petya has recently started attending a programming club. Naturally he is facing the problem of choosing a programming language. After long considerations he realized that Java is the best choice. The main argument in favor of choosing Java was that it has a very large integer data type, called BigInteger.

But having attended several classes of the club, Petya realized that not all tasks require using the BigInteger type. It turned out that in some tasks it is much easier to use small data types. That's why a question arises: "Which integer type to use if one wants to store a positive integer n?"

Petya knows only 5 integer types:

1) **byte** occupies 1 byte and allows you to store numbers from -128 to 127

2) **short** occupies 2 bytes and allows you to store numbers from -32768 to 32767

3) **int** occupies 4 bytes and allows you to store numbers from -2147483648 to 2147483647

4) **long** occupies 8 bytes and allows you to store numbers from -9223372036854775808 to 9223372036854775807

5) **BigInteger** can store any integer number, but at that it is not a primitive type, and operations with it are much slower.

For all the types given above the boundary values are included in the value range.

From this list, Petya wants to choose the smallest type that can store a positive integer n. Since BigInteger works much slower, Peter regards it last. Help him.

Input

The first line contains a positive number n. It consists of no more than 100 digits and doesn't contain any leading zeros. The number n can't be represented as an empty string.

Please, do not use `%lld` specificator to read or write 64-bit integers in C++. It is preffered to use `cout` (also you may use `%I64d`).

Output

Print the first type from the list "`byte`, `short`, `int`, `long`, `BigInteger`", that can store the natural number n, in accordance with the data given above.

Examples

input
```
127
```
output
```
byte
```

input
```
130
```
output
```
short
```

input

```
123456789101112131415161718192021222324
```

output

```
BigInteger
```