Classes and Objects

Write a program, which will print on standard output `System.out` a prime factorization for numbers given as command-line arguments. Factorization of each of the numbers should be printed in a separate line: first the given number, then the = sign, then the factors separated by *.

Command-line parameters of the program invocation should be integer numbers. Invalid argument format should be reported on the standard error stream `System.err`. If the program was called without any argument, it should print a help message on the error stream.

To solve the task, you shuld define a helper class `PrimeNumbers`. The class should contain only two, static, methods: `isPrime()` and `toPrimeFactors()`. Use the *sieve of Eratosthenes* algorithm, modified to store the smallest factor of each number (it will be helpful when computing the factorization). The functions should work for all numbers of the `long` type (remember that you cannot create such a huge sieve, so find an algorithmic workaround). The `PrimeNumbers` class should be defined in such a way that its instances cannot be built.

```
public class PrimeNumbers
{
  protected final ststic int POWER_OF_2 = 21 ;
  protected final static long[] SIEVE = new long[1<<POWER_OF_2] ;
  // ...  initialization block for the sieve needed ...
  public static boolean isPrime (long x)
  { /* ...  */ }
  public static long[] toPrimeFactors (long x)
  { /* ...  */ }
}
```

Factorization of a negative number should start from the factor -1. The program should handle the numbers -9223372036854775808 and 9223372036854775783 correctly. The prime factorization of -1 and 1 should be -1 and 1 respectively.

The program should be compiled and run from the command line.

Hint. A compound number n has at least one prime factor, which is $\leqslant \sqrt{n}$.