Requirements

Write a Java program to compute large Fibonacci numbers. Part of this program should be a Java class that implements unlimited-precision integers, and their basic arithmetic, using a linked-list implementation.

As you know, the ordinary Java "int" is 32 bits and stores values up to about 2 billion. In some domains, such as cryptography or number theory, this is too small. If you were interested in large Fibonacci numbers, you might like to know that the 10,000th one is

```
33644764876431783266621612005107543310302148460680063906564769974680081442166662368
```

. . . which is more than 2 billion. Your task in this first assignment is to implement an integer abstract data type with no predetermined limits on precision, so that (as a demonstration) you can compute large Fibonacci numbers. EDIT (17 June) -- we are restricting the type to represent only nonnegative values.

Goals: Review linked lists, think about numeric data types, familiarize yourself with the programming practices for this class.

Deadline: 25 May, 6:00 AM. (I recommend you turn it in the night before.)

Detailed requirements:

- Your assignment should include a Java class called *LLInt* (for linked-list integer), implementing the integral data type using a linked list, where each linked list node will store a limited-range value (like a single decimal digit). EDIT (17 June): nonnegative values only!
- Don't assume each node stores a single digit; instead, your class must contain a positive *static final int RADIX* value, which you should set to 10. Then, assume each node will store an integer value between 0 and *RADIX* - 1 (but see exceptions below).
- The class should be able to instantiate a *LLInt* from an ordinary *int*, from a *String* of decimal digits (e.g., "5280" interpreted the usual way), or by copying the contents of another *LLInt*. The constructor accepting a *String* may throw an exception if *RADIX* does not equal 10. (Or, make it work for arbitrary RADIX, if you are ambitious.) EDIT (17 June): You may throw an exception for bad inputs, and that includes negative inputs.
- The class should contain a *toString* method that produces a decimal representation in the form of a string. This also may throw an exception if *RADIX* is not equal to 10. (Again, ambitious programmers are welcome to implement this for arbitrary RADIX.)
- The class should support addition, ~~subtraction,~~ and multiplication. EDIT (17 June): you need not do subtraction.
- Outside the *LLInt* class, write a function that returns the *n*-th a Fibonacci number, given *int n*. The function should be efficient enough to produce the 20,000th Fibonacci number in less

Download Print

Activity Details

You have viewed this topic

Last Visited May 18, 2016 8:30 PM