

Programming Problems

20th Annual Computer Science Programming Contest

Department of Mathematics and Computer Science
Western Carolina University
31 March 2009

The contest is 10 to noon. The award ceremony is 1:45 to 2:00 in 434 Stillwell.

Problem 1 : Stock Picking

The price of a single stock can vary greatly from day to day. Your program should input the price of a stock for several consecutive days, then output the best day to buy and sell the stock in order to make the most profit. Note that the program cannot merely find the day with the highest price and the day with the lowest price; the sale of the stock must come *after* the purchase of the stock, not before. Your program should output the day of the buy, the buying price, the day of the sale, the selling price, and the profit from the sale. Your program must allow the user to enter as many days worth of data as the user wants to. (Note: Given the stock market over the last year, we thought this problem was especially appropriate!)

Here is an example of the program input and output:

```
program outputs: Please enter the price for each day.  
program outputs: End your input with a 0.  
user inputs: 10  
user inputs: 2  
user inputs: 5  
user inputs: 3  
user inputs: 7  
user inputs: 4  
user inputs; 0  
program outputs: You should buy on day 2 at a price of 2  
program outputs: You should sell on day 5 at a price of 7  
program outputs: You will earn 5 dollars for this sale
```

Problem 2: ISBN Verifier

Books have ISBNs (International Standard Book Number) which uniquely identify them. They have a built-in check digit to verify that the scan of the book was correct. Today a standard ISBN is 13 digits long, with the 13th digit being the check digit.

To calculate the check digit of the ISBN $a_1a_2a_3-a_4-a_5a_6a_7-a_8a_9a_{10}a_{11}a_{12}-?$, the following algorithm is used (for the first 12 digits):

$$(10 - [(1a_1 + 3a_2 + 1a_3 + 3a_4 + 1a_5 + 3a_6 + 1a_7 + 3a_8 + 1a_9 + 3a_{10} + 1a_{11} + 3a_{12}) \bmod 10]) \bmod 10$$

For example the ISBN 978-0-451-52493-5

$$\begin{aligned} & (10 - [(1(9)+3(7)+1(8)+3(0)+1(4)+3(5)+1(1)+3(5)+1(2)+3(4)+1(9)+3(3)) \bmod 10]) \bmod 10 \\ &= (10 - [105 \bmod 10]) \bmod 10 \\ &= (10 - [5]) \bmod 10 \\ &= (5) \bmod 10 \\ &= 5 \end{aligned}$$

has a correct check digit.

Write a program that:

- Takes a user input string and checks that it consists of 12 or 13 digits (and possibly dashes)
- If there are only 12 digits, it computes the check digit and outputs it along with the ISBN in standard format (xxx-x-xxx-xxxxx-x)
- If there are 13 digits, it computes the check digit and checks it against the input then notifies the user if the ISBN is valid.
- Your program should ask if the user wants to input another ISBN or quit.

Some example runs (user input is in boldface).

```
Enter an ISBN with or without the check digit (xxx-x-xxx-xxxxx-?): 978-0-451-52493-
The ISBN with Check Digit : 978-0-451-52493-5
```

```
Enter an ISBN with or without the check digit (xxx-x-xxx-xxxxx-?): 978045152493
The ISBN with Check Digit : 978-0-451-52493-5
```

```
Enter an ISBN with or without the check digit (xxx-x-xxx-xxxxx-?): 978-0-451-52493-
5
The ISBN 978-0-451-52493-5 is valid.
```

```
Enter an ISBN with or without the check digit (xxx-x-xxx-xxxxx-?): 9780451524935
The ISBN 978-0-451-52493-5 is valid.
```

```
Enter an ISBN with or without the check digit (xxx-x-xxx-xxxxx-?): 978-1-586-48556-
7
The ISBN 978-1-586-48556-7 is NOT valid.
```

Problem 3: Simple Calculator

Write a simple calculator that reads in a number, an operation, a second number, and finally outputs the answer. For full credit your solution needs to validate the input as shown in the sample run.

Sample run (user input is in boldface):

```
Enter a number: 134  
Enter an operation (+, -, *, /): 4  
Invalid operation.  
Enter an operation (+, -, *, /): *  
Enter a second number: 178  
134*178 = 23852
```

Bye!

Problem 4: Fibonacci Power Numbers

There is a famous sequence of numbers called the Fibonacci sequence, where the next number in the sequence is the sum of the previous two. It starts off with 1 and 1.

1, 1, 1+1=2, 1+2=3, 2+3=5, 3+5=8, 5+8=13, 8+13=21, ... and so on

We can create our own sequence similar to this where the next number in the sequence is the product of the previous two. We will start off the sequence with 2 and 2 (since $1*1=1$, starting off with 1 and 1 would get us nowhere).

2, 2, 2*2=4, 2*4=8, 4*8=32, 8*32=256, 32*256=8192, ... and so on

Notice that we could also write this sequence as powers of two:

$2^1, 2^1, 2^2, 2^3, 2^5, 2^8, 2^{13}, \dots$ and so on

So we really just have twos raised to Fibonacci numbers.

Write a program that calculates and outputs the n th number in this sequence. These numbers get big fast, so you need to use the `BigInteger` class. We do not recommend inputting something larger than 30 (which takes 70 seconds to run on the development machine). For full credit your solution needs to validate the input as shown in the sample runs.

Notice that the first value in the sequence is the zeroth number in the sequence.

Some example runs (user input is in boldface) .

```
Enter a positive integer value: 0
Enter an integer value greater than zero!
Enter a positive integer value: -2
Enter an integer value greater than zero!
Enter a positive integer value: df
Enter an integer value greater than zero!
Enter an integer value greater than zero!
Enter a positive integer value: 2
4
```

```
Enter a positive integer value: 10
618970019642690137449562112
```