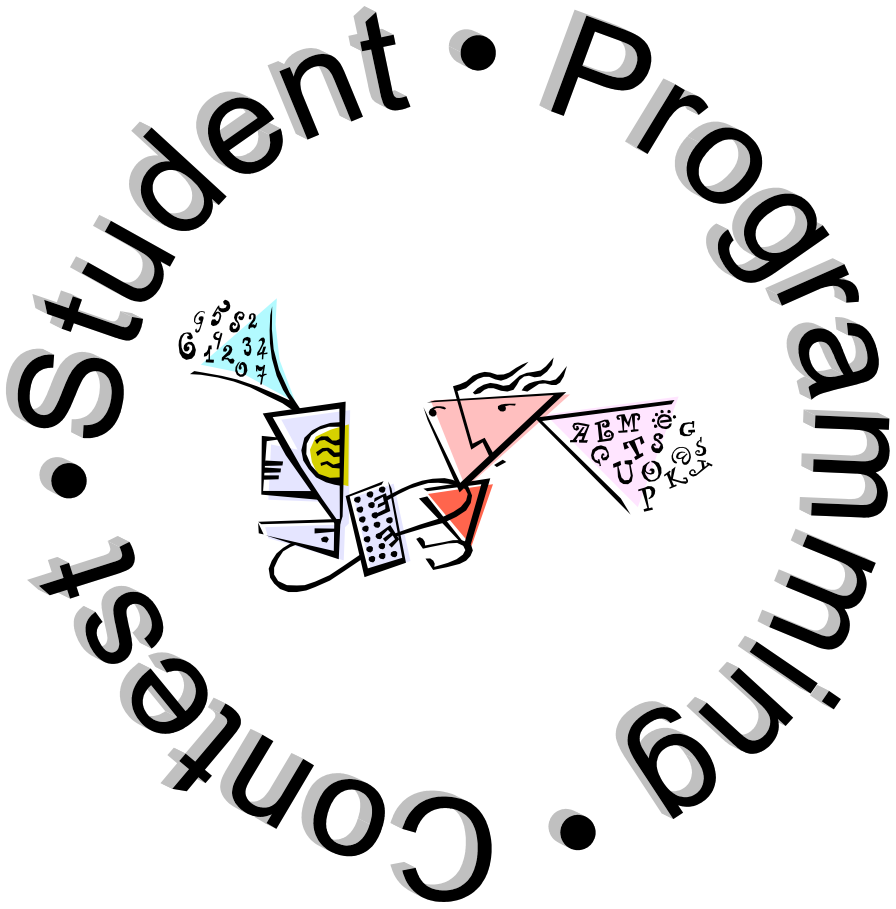
The Ninth Annual

Student Programming Contest

of the
CCSC Southeastern Region



Saturday, November 9, 2002
8:00 A.M. – 12:00 P.M.

The Sequence Formerly Known As Fibonacci

The Problem

We all know what the Fibonacci sequence is: a series such that the ith term is the sum of the previous two terms, starting with 0,1. Thus the sequence starts 0,1,1,2,3,5,8,13,21... Each number is the sum of the preceding two. But the Fibonacci sequnce is old, it's had its day and now it's time for bigger and better things. You, being an astute observer of current trends and fashion, have noticed that the Fibonacci sequence, while once all the rave, is no longer spoken of at happening events.

However, you feel that sequences in themselves are far from dead, and, because you lust after notoriety and glory, you have decided that the next big sequence to hit the scene should be named after you. You decide that your sequence should be reminiscent of the Fibonacci sequence because it seemed to have a good formula for success, but also unique enough that it will not receive the same lackadaisical response.

Thus, after many sleepless nights you come up with a brilliant idea. Your sequence will be like the Fibonacci sequence, except that each number will be the sum of the previous three numbers, not the previous two. Thus, you write a computer program that will generate the sequence for a given three numbers.

Create a program that will take as input four integers. The first three are the first three terms of the sequence (in order). The fourth input, N, is the length to which your sequence should be calculated. Your method should return the sum of the first N terms in the sequence.

Judges will ensure the validity of the inputs. Inputs are valid if all of the following criteria are met:

- N is between 1 and 10000, inclusive
- a, b, and c are all between -10000 and 10000, inclusive
- For all i less than N, the absolute value of the ith term and the absolute value of the sum of all terms from 1 to i will be less than 2^15 = 32768

Example

Input: a=0,b=0,c=1,N=7
Output: 15
Explanation: The sequence starts with (0,0,1) and the first 7 terms are as follows:
0,0,1,1,2,4,7 - Each term is the sum of the previous 3.
The sum of all these is 15, so the method should return 15.

Sample Input

Your program should take its input from a file. Three examples of input (A, B and C) are shown below:

(A) 0,0,1,7
(B) 1,2,3,8
(C) 1,0,0,10

Sample Output

Your program should direct its output to the screen. Appropriate output for the sample inputs (A, B and C) is shown below: **NOTE: Your output should look EXACTLY like what is displayed here!**

(A) The Answer is: 15
(B) The Answer is: 148
(C) The Answer is: 53

Insomnia

The Problem

When I can't sleep I count sheep. During the first hour I count 900 sheep (I start at 1 and count up: 1, 2, 3, ..., 899, 900). During the next hour I count them down (so as to avoid having my numbers of sheep grow too large), that is, I count 899, 898 etc. As I count down slower than up, I count 600 sheep down during the second hour. During the third hour I count up again starting where I just finished counting down. For instance, the end of the second hour would go something like this (302, 301, 300, 301, 302, …). The pattern continues alternating counting up 900 and counting down 600.

Your task is, given the number of minutes I spend trying to get to sleep, return the final number of sheep where I stop counting.

Notes:

- For the purposes of this problem, assume that every hour contains exactly sixty minutes.
- Assume that each hour I count evenly: either 15 sheep a minute up, or 10 sheep a minute down.

Judges will ensure the validity of the inputs. Inputs are valid if all of the following criteria are met:

- minutes are between 0 and 720 inclusive

Examples

1. minutes = 0, return 0
2. minutes = 185, return 900 - 600 + 900 - 50 = 1150

Sample Input

Your program should take its input from a file. Three examples of input (A, B, C and D) are shown below:

(A) 0
(B) 120
(C) 2
(D) 185

Sample Output

Your program should direct its output to the screen. Appropriate output for the sample inputs (A, B, C and D) is shown below:

(A) 0
(B) 300
(C) 30
(D) 1150

Biodiversity

The Problem

We are interested in how much biological diversity can be generated by very simple life and death rules. The environment is a line of cells. Each cell may either be alive or dead.

We will define the "diversity" in a line of cells to be the number of different sizes of alive groups that are present. An alive group is a contiguous set of alive cells. So a line of cells AADADDDADDDAAAADDADDA has diversity equal to 3 since it contains alive groups of sizes 2, 1, and 4.

Your job is to take a configuration of cells as input and return its diversity.

Notes:

- A line will contain only A (alive) and D (dead).

Judges will ensure the validity of the inputs. Inputs are valid if all of the following criteria are met:

- line contains between 1 and 50 characters inclusive
- each character in line is uppercase A or uppercase D

Example

(quotes shown for clarity only)
"ADDAADD": return 2

Explanation

This line has a group of size 1 and a group of size 2

Sample Input

Your program should take its input from a file. Three examples of input (A, B and C) are shown below:

(A) ADDAADD
(B) ADDAAADAAAD
(C) DDD

Sample Output

Your program should direct its output to the screen. Appropriate output for the sample inputs (A, B and C) is shown below:

(A) 2
(B) 2
(C) 0

Tickets, Please

The Problem

It is Saturday morning of the opening weekend for "Lord of the Rings". There is a huge line at the movie theater. Each person in line buys one ticket. The ticket price is $4.50. Each person has either four one dollar bills and two quarters or a 5 dollar bill. The cashier doesn't accept credit cards, and starts selling tickets having only 2 quarters at hand.

Your task is, given a description of the line, return the number of the first customer who will not get change back.

Notes:

- the line will be represented by a String of 'e's for a customer with exact change and 'f's for a customer with a 5 dollar bill. For example, "eeefef"
- customers are represented from left to right, that is, the first character on the left represents the first customer and so on
- customers are served in the order they are standing in line
- if change can be given to all customers that need it, return -1
- in your answer, customers should be numerated starting from 1, where 1 means the first customer.

Judges will ensure the validity of the inputs. Inputs are valid if all of the following criteria are met:

- line contains between 0 and 50 characters inclusive
- each character in line is either 'e' or 'f'

Examples

1. line = "ff" returns 2, as the cashier has change only for the first customer
2. line = "eeeeeeeeeeeeeeeeeeeeeeeeeeeee" returns -1, as nobody needs change
3. line = "efefff" returns 6

Sample Input

Your program should take its input from a file. Three examples of input (A, B and C) are shown below:

(A) fefeeffffffffffffff
(B) eeeeeffffff
(C) e

Sample Output

Your program should direct its output to the screen. Appropriate output for the sample inputs (A, B and C) is shown below:

(A) 8
(B) 12
(C) -1

The Task Master

The Problem

Two friends are given a list of tasks to complete for work. The faster they complete the work, the more they get paid. Since they split the money, they can do any task from either list in any order, but tasks cannot be subdivided (i.e. they can not cut up a 10 into a 7 and a 3).

Your job is to take as input two task lists and return the minimum time (in minutes) required to finish all tasks assigned.

Judges will ensure the validity of the inputs. Inputs are valid if all of the following criteria are met:

- task1 and task2 will each have 0 to 5 elements (inclusive)
- each element of task1 and task2 will be an integer between 0 and 200 minutes (inclusive)

Example

1. task1 = {5,2,9} and task2 = {1,1} Method returns 9

Explanation

The first friend does the first and second task from task1, and both tasks from task2, which takes 5+2+1+1 = 9 minutes. The other friend does the last task of task1 in 9 minutes. This is the shortest amount of time possible to complete all tasks.

Sample Input

Your program should take its input from a file. Three examples of input (A, B, C and D) are shown below:

(A) {5,2,3} {5,2}
(B) {10,20,40} {1}
(C) {5,2} {5,2}
(D) {5,5} {2,2}

Sample Output

Your program should direct its output to the screen. Appropriate output for the sample inputs (A, B,C and D) is shown below:

(A) 9
(B) 40
(C) 7
(D) 7

Breakdown & Classify

The Problem

The main purpose of this problem is to accept a sentence as input, it will then print out each word in that sentence on a separate line along with a word number. For Example:

Your Sentence Please:
Kirk, commands the Enterprise.
The words are:

1.) Kirk
2.) commands
3.) the
4.) Enterprise

The program will then consult a list of three types of words - Verbs, Nouns and Others (attached is the list of these words) and will attempt to classify the sentence as one of the following:

Simple: This sentence has only one Noun, one Verb, and zero or more Other words in it, and the noun must precede the verb. Examples include

- Picard is
- Uhura dances quickly
- The Enterprise flies slowly

Complex: This sentence has two Nouns, one Verb, and zero or more Other words in it, One noun must precede the verb and the other follows it. Examples include

- Picard is Captain
- Uhura dances quickly on ice
- The Enterprise flies slowly through space

Other: Anything that is not simple or complex

Notes:

- Your program should be case insensitive, that is to say if the database contains the words spock, Spock, SPOCK and SpOcK would all match on that word.
- Your program will accept and ignore the following punctuation marks in a sentence:
  . ; , ? !
- The list of verbs, nouns and other words should be stored in a file that is read and written by the program. The user can add words by editing the file to include the new word. The user can delete words by removing words from the file.
- After a sentence has been analyzed its type should be printed out.
- Your program should accommodate sentences up to 80 characters long.
- A word is defined as a group of adjacent characters.
- Word breaks are indicated by spaces or tabs.
- A sentence is terminated by a period, question mark or exclamation point.
- Your program should be able to handle a database of up to 20 Nouns, 20 Verbs and 40 Others.
- Each word will be at most 12 user input characters long.

Data:

The following words should be in the database to start with:

Nouns: kirk, spock, mccoy, enterprise, planet, space, shuttle, console
Verbs: flies, uses, orbits, commands, argues
Other: with, on, in, the, large, small, a, an, loudly

Sample Input

Your program should take its input from a file. Six examples of input (A, B, C, D, E and F) are shown below:

(A) Bob is cool
(B) Spock, uses Console.
(C) A a spock ! a a a uses a a a console a a a
(D) spock, Spock uses!
(E) Spock, uses a Spock Spock
(F) Spock uses uses

Sample Output

Your program should direct its output to the screen. Appropriate output for the sample inputs (A, B, C, D, E and F) is shown below:

(A) contains unknown word
(B) complex
(C) complex
(D) other
(E) other
(F) other