

**Note:** The following is a trivial sample problem intended for instructional purposes only. Please see the back of the page for a solution and how to test and submit the solution to the judges. Feel free to use the sample solution as a template for your own programs when solving the other problems.

## Problem 0: Just Add Water

A weather station keeps track of rainfall for a local area by electronically recording the amount of rain (in mm) that falls during every 24 hour period. Due to a flaw in the sensor, corrupt readings may also be recorded by the weather station. These errant readings are always recorded as negative numbers. Write a program which will determine the total rainfall measured by the weather station.

Each input file contains a single row of numbers representing all the recordings (both legitimate and corrupt) made by the weather station. You may assume that all input files are formatted correctly.

**Sample input #1** (available as file “test0a.dat”):

```
5 -44 2 12 4 9
```

**Sample output #1:**

```
76mm
```

**Sample input #2** (available as file “test0b.dat”):

```
-8 4 -23 12 34 12
```

**Sample output #2:**

```
62mm
```

**Sample input #3** (available as file “test0c.dat”):

```
872 -621 266 563 -394 0
```

**Sample output #3:**

```
1701mm
```

Please turn over  $\implies$

# Submitting a Solution to the Problem

1. You've determined that this is the easiest problem, so you choose to solve it first. This is Problem 0, so you create a file called `problem0.java` in the `apics1-a0` directory and write the following program:

```
import java.util.*;
class problem0 {
    public static void main(String args[]) {
        Scanner s = new Scanner(System.in);
        int    total = 0;

        while (s.hasNextInt()) {
            int num = s.nextInt();
            if (num > 0)
                total += num;
        }
        System.out.println(total + "mm");
    }
}
```

FYI: You can use `s.hasNext()/s.next()` to read words, and `s.hasNextLine()/s.nextLine()` to read entire lines. See the Java documentation for the `Scanner` class for more information.

NOTE: For Problem 1, Problem 2 and Problem 3, you would use the file names `problem1.java`, `problem2.java` and `problem3.java`, in the directories `apics1-a1` `apics1-a2` and `apics1-a3` respectively.

2. Compile the program on the command line using `javac`:

```
$ javac problem0.java
```

Please *do not* submit a program that cannot be compiled.

3. If the compiler doesn't generate any warnings or errors, test the program on **all** the test data corresponding to that problem. This data is located in the `/local/pub/cs/apics/` directory. For example:

```
$ java problem0 < /local/pub/cs/apics/test0a.dat
76mm
$ java problem0 < /local/pub/cs/apics/test0b.dat
62mm
$ java problem0 < /local/pub/cs/apics/test0c.dat
1701mm
```

Make sure that the output generated by your program is similar to that in the corresponding "Sample output" section of the problem description before you submit it.

4. Submit your solution using the `submit` program.

```
$ cd
$ submit submit apics1-1 a0
<enter your password, when asked.>
```

Instead of `a0`, use `a1`, `a2` and `a3`, for problems 1, 2 and 3, respectively, during submission.

5. Write your team letter (*e.g.*, A, B, C, D, ...), the problem number and current time (as shown on the Competition Scoreboard) on the submission slip. Give the slip to a runner who will deliver it to the judges.

The judges will run your program using the same test data. They will also run your program using test data that you don't have access to so as to ensure that your program works as expected. If there are no major discrepancies between the output generated by your program and the corresponding sample output, your solution is deemed to be correct.