

## Program 1: Array Practice

### Program due: Monday, February 18th, 8 p.m.

Write a MIPS program named **prog1.s**. Your program will use an array of integers labeled **numbers**. The number of integers in the array will be stored in a word labeled **numElements**. One additional integer will be present, labeled **skip**. Your program will accomplish three tasks:

- Print the contents of the **numbers** array, with 5 numbers to a line. Put a tab character in front of each number.
- Print the contents of the numbers array again. This time, do not print any zeroes. The other requirements are the same: 5 numbers to a line with a tab character in front of each number.

As an example, if the test case we provide contains:

```
.data

numElements:
    .word 18

numbers: .word 17
           .word -50
           .word 3
           .word -23
           .word -60
           .word 97
           .word 0
           .word -82
           .word 53
           .word 30
           .word -17
           .word 142
           .word 93
           .word 0
           .word -1
           .word 1
           .word 0
           .word 5

originalString:
    .asciiz "The integers are\n"
noZeroesString:
    .asciiz "The integers without any 0's are\n"
tabString:
    .asciiz "\t"
newline:
    .asciiz "\n"

# Your code goes below this line
```

The output of your program for this test case should be:

**The integers are**

```

17      -50      3      -23      -60
97       0      -82      53       30
-17     142     93       0       -1
1        0        5

```

**The integers without any 0's are**

```

17      -50      3      -23      -60
97      -82     53       30      -17
142     93     -1        1        5

```

## Input:

We will provide the first part of your program. This allows us to use different values to test your program. The **.data** section shown on the first page is an example of what we will provide. We will provide additional test cases that you can use.

We guarantee that the number of integers in the array **numbers** will be at least as many as stated by **numElements**. It is possible that there will be more values in the **numbers** array than stated by **numElements**; in this case, you are to use only **numElements** elements of the array (see test case #6, for example). Note that it is possible that **numElements** can be zero; see the provided sample output for test case #9 for how this case is to be handled by your program.

We will provide in each test case, the **numElements** integer, an array named **numbers** that will contain at least **numElements** values, and 4 strings named **originalString**, **noZeroesString**, **tabString**, and **newline**. Use these strings in printing your answers.

Your program must have the comment:

```
# Your code goes below this line
```

Everything that you provide (comments and code) must go below this required comment line. This will include any additional variables that you need (and you will need some additional string definitions)

We use a script in grading your program. The script assumes this line is present in your program. You can write anything below the line that is necessary for your program to work. This would include additional values for the **.data** section, the entire **.text** section, and all comments.

We will provide sample files for you to copy. They will be available to copy on the Unix machines (lectura and the Fedora machines, fd01 to fd08). They will also be on the Windows machines. On the Windows systems, look in the shared Rotis drive: **Rotis->csc252->shared->prog1**. On the Unix machines, look in the directory: **~cs252/spring08/prog1**. In both cases, the files will be named: **test01.s**, **test02.s**, etc. Secure ftp can also be used to access the test cases on lectura from your home system. Please check with us if you have problems accessing these test cases.

The required comment line:

```
# Your code goes below this line
```

must be present in the **prog1.s** file that you turnin! The **prog1.s** that you turnin may contain one of the test cases above the required comment, or the required comment can be the very first line (no test case data above it). Either is acceptable.

## Output:

Your program should produce:

- The string “The original integers are”
- A list of the **numElements** integers from the **numbers** array, five integers to a line. Each number to be preceded by a tab character.
- The string “The integers without any 0’s are”
- A list of the non-zero **numElements** integers from the **numbers** array, five integers to a line.

Do not print any additional information. Do not print any extra blank spaces, any extra blank lines. Print nothing but what is required. We use automated scripts for grading that will look for an exact match on the output!

See the attached notes (also available on the web page) about how to check your program using the provided test cases. We will use the provided test cases in grading your program. We do reserve the right to use additional test cases beyond those that we provide.

## Turnin:

Note: We are asking for submission of programs via D2L only.

- Name your program: **prog1.s**
- Using a web browser, go to: <http://d2l.arizona.edu/>
- Login using the “UA NetID Login” (upper-left corner of the web page).
- You should now be at “My Home” on D2L. At the center bottom of the screen, under the heading “My Academic Courses”, you should find “C SC252 SP08 001-002H Homer” listed under 2008 Spring. Click on this link.
- You will now be at the CSc 252 page for Spring 08.
- There is a row of links just underneath the Wildcat. In this row, you will find the link “Dropbox”. Click on this link.
- You will be at a page that shows three Dropbox Folders: **Program1**, **Program1-Late**, **Program1-Regrade**. Only one of these will be active. The **Program1** folder is for on-time submissions. The **Program1-Late** folder will be available for late submissions. The **Program1-Regrade** folder is used for requesting a re-grade of the program after the grades have been returned. Click on “Program1”. If you are doing a late turnin, click on “Program1-Late”.

- You should now be at the page: “Submit Files - Program1”. Click on the “Add a File” button. A pop-up window will appear. Click on the “Choose File” button. Use the file browser that will appear to select your **prog1.s** file. Click on the “Upload” button in the lower-right corner of the pop-up window.
- You are now back at the “Submit Files - Program1”. Click on “Upload” in the lower-right of this window. You should now be at the “File Upload Results” page, and should see the message **File Submission Successful**.
- You can repeat this process as often as necessary. Each submission will be time-stamped. When we grade programs, we will grade only the most recent submission.

## PC Spim and the exceptions.s file

For those who use PCSpim on your own machine and/or on one of the department’s Windows machines: You may need to set the **exceptions.s** file location.

On the department’s Windows machines, the correct location for **exceptions.s** is

**C:\Program Files\PCSpim\exceptions.s**

You can check that this setting is correct and/or make changes as follows:

- a.) Click on Simulator in the PCSpim Menu Bar.
- b.) Click on Settings...
- c.) In the pop-up window that will appear, the path should be:

**C:\Program Files\PCSpim\exceptions.s**

If it is not, then change the setting.