Winter 2011 **CPE 101:** Fundamentals of Computer Science I Alexander Dekhtyar

Lab Test Example #2

Instructions

- Time to complete the exam: 50 minutes.
- This exam is individual
- This exam is open book
- This exam is closed everything else, including lecture notes, electronic devices, etc
- The only programs you are allowed to have open on the computer during this exam are:
  - The text editor of your choice
  - Terminal window(s)
  - ssh connection to vogon
- Do not use any code you may have access to from earlier in this course or from other courses
- Before starting your work, open a terminal session, and create a directory for the exam. It will be referred to as `labexam01` below.
- The problem description includes all assumptions necessary to answer the problem. Please raise your hand, or approach the instructor if you have any questions.
- Please, collect candy wrappers after you! Additional candy is available, at the instructor's desk: help yourself!
- GOOD LUCK!

Assignment

The input to your program will consist of two parts. First, your program will take as input a sequence of numbers. It will count how many times each number was found in the input and store this. The second part of the input will consist of queries. Your program shall correctly answer each query and print the output.

Input: sequence of numbers. The first part of the input is a sequence of integer numbers — one number per line. The numbers can be repeated. The sequence of numbers will contain up to 20 unique numbers. It is terminated by a sentinel line contianing `-1`, which **should NOT count** as part of the sequence of numbers.

Input:queries. The rest of the input to your program will consist of queries. Each query is an integer number occupying a single line. The number of queries is arbitrary. The queries will be terminated by a sentinel line containing the value `-1`.

The English version of the query n is *"How many times does the number n appear in the sequence?"*

For example, the following *complete* input,

```
2
3
4
5
3
4
3
-1
2
3
4
7
-1
```

represents a sequence of numbers that contains one copy of number 2, three copies of number 3, two copies of number 4 and one copy of number 5. There are three queries ask your program to find how many times numbers 2,3, 4 and 7 appear in the input sequence.

Program behavior. Your program shall read the first part of the input (the sequence) and store it in an array (or collection of arrays). For each number in the input sequence, your program shall store the number of times it was encountered in the sequence.

Your program shall then read, one-by-one, each query and produce an answer to it.

For each query, your program shall search the array (collection) for the query number. If the number is found, your program shall output

```
NUMBER is found X times
```

Here, `NUMBER` is the number from the query and `X` is the number of times `NUMBER` occurs in the input sequence.

If the number is not found, your program shall print

```
NUMBER is found 0 times.
```

For example, for the input above, your program shall produce the following output:

```
2 is found 1 times
3 is found 3 times
4 is found 2 times
7 is found 0 times
```

Program structure. You get to decide how you want to store the data in your program (parallel arrays, structure, multidimensional array, etc.)

Your program shall include function `int find()` which takes as input the number from a query, the total number of unique numbers found in the sequence and the necessary array(s) and outputs the index of array storing the provided number, or -1 if the number is not found in the array.

Error checking. No error checking is necessary, all input files will contain correct inputs.

Program name. Name your program `repeats.c`.

Testing your solution

Compile your program using the following command:

```
> gcc -ansi -Wall -Werror -lm -o repeats  repeats.c
```

You have access to my full set of test cases for this program, my executable of it and scripts to test my executable and your executable. Copy all the files to your `labexam03` directory with the following command:

```
cp ~dekhtyar/www/101-Winter2011/nrem09/* .
```

Instructor's executable is called `repeats-alex`. Do `ls -al` to ensure that it is executable. If not, run `chmod u+x repeats-alex`. Perform the same check for `repeats-test.csh` and `repeats-alex-test.csh` files that are also copied. Test file names are `repeats-test01` through `repeats-test10`.

To test your program first compile it into an executable named `repeats`. Then run the following command:

```
> repeats-test.csh
```

The script will run your program on all test cases and produce output. To check it against the instructor's output, open a second terminal session, change to `labexam03` directory and run the command:

```
> repeats-alex-test.csh
```

Compare the two outputs.

Note. You can run the following commands:

```
> repeats-test.csh > my.out
> repeats-alex-test.csh > alex.out
> diff my.out alex.out
```

If the last command does not produce any output, your results exactly match the instructor's results.

Submitting your solution

Once you are satisfied your program meets all the requirements you may turn it in and leave the exam. Run the following command on vogon to submit your program:

```
handin dekhtyar  labexam03 repeats.c
```