CPE 101
Turner
Fall 2009

PRACTICE LAB EXAM #3
Pass/Fail
PLEASE work this out and use all resources at your disposal to become comfortable with all the concepts and the commands that may be new to you. You will not have time during the quiz to do any research, just to check a few things.

Instructions as they will appear on the actual exam:

- You have 40 minutes to complete this exam
- The exam is to be completed ***by you alone***
- The exam is open book
- The exam is closed to all other sources
- The only programs you are allowed to have open on the computer during this exam are:
  - a terminal window
  - the editor of your choice
  - ssh session on vogon
  - gcc
    - ♣ you must compile using gcc –ansi –Wall –pedantic
    - ♣ your code must run with ***no errors or warnings***
    - ♣ your code must pass all tests you're given
- ***Do NOT use any code you've already written*** for other courses or this course
- ***Create a directory called LabExam3 to do your editing and compiling – this is the only directory I want to see displayed on your terminal during the quiz.***
- The problem description contains all necessary information to solve the problem. If you have questions, ask the instructor or lab assistant.

**Problem**: You are to write a program named LabQuiz3.c to read in up to 25 non negative integers from stdin. It will read in the integers, one at a time until the sentinel of -1 is reached. Your program must store those integers in an array in your program. You will also write a function designed to determine whether an integer is “odd” or “even” to utilize in your program (this must be written as a function and called as a function). Your program will print out each element of the array you’ve stored, one at a time, and you’ll print out whether the given element is “odd” or “even” on the same line, then go to the next line for the next element, until the end of the array is reached.

***Your output must be precisely as the following example, words, spacing and all. You will have a chance to test your output to match these specifications during the quiz to be sure it matches correctly. See instructions below.***

Sample input from stdin:

```
2 4 6 35 44 2 63 65 77 -1
```

Sample output to stdout:

```
Array [0] has the value 2 and it is Even
Array [1] has the value 4 and it is Even
Array [2] has the value 6 and it is Even
Array [3] has the value 35 and it is Odd
Array [4] has the value 44 and it is Even
Array [5] has the value 2 and it is Even
Array [6] has the value 63 and it is Odd
Array [7] has the value 65 and it is Odd
Array [8] has the value 77 and it is Odd
```

**INPUT**: You will read from stdin (the keyboard or a file using unix redirection)

**OUTPUT**: You will write to stdout (the screen or a file using unix redirection)

You will compare your output with my “Oracle.txt” file. Reminder, unix redirection uses the “<” and “>” symbols to redirect input from stdin and output to stdout. You can replace keyboard input to your program by the command

```
./myprogram < Input.txt
```

and you can redirect the output that would normally go to the screen to a file named “Output.txt” by the command

```
./myprogram > Output.txt
```

Testing your solution:

Please copy the test case for your exam from ~csturner/101LABTEST/X3TEST/
Use the command            cp  ~csturner/101LABTEST/X3TEST/*.txt .
(Don't forget that last "dot" at the end after the .txt - one space is required between the '*' and that '.', the last '.' refers to *your current directory* where the files will be copied for you.)   After you execute this command you should have the test files "Input.txt" and "Oracle.txt" in your directory,  use the unix command "ls" to see if they are there (you should be inside your LabExam3 directory).  You'll need to test your program using the file Input.txt and redirect the output to a file called "Output.txt"    (or some other name of your choosing.)

I expect that if you name your program LabQuiz3.c and compile it for use with keyboard input and screen output, you will execute the command

```
./LabQuiz3 < Input.txt > Output.txt
```

Then you can use the "more" or "cat" commands or an editor to see whether the file Output.txt contains what you expect for the Input.txt file as input.

That output file of your program, in this case Output.txt will then be compared to my output that was redirected to the file named Oracle.txt

Use the unix "diff" command to see if your Output.txt matches the output of my oracle program:  Oracle.txt   -  if there is any output at all from diff, you must debug your program so that your output matches mine.

I expect you'll execute the command

```
diff Output.txt Oracle.txt
```

If you see any output on the screen after this command then the files differ and you need to make some adjustments to your output to match mine.

WHEN YOU are finished, use handin to hand in your (now correct after testing ….) program.

```
handin gradercst LabQuiz3 LabQuiz3.c
```

where LabQuiz3.c is your source code and it resides in the directory from which you execute the command.