APS 105 — Computer Fundamentals

Lab #4: Making Decisions and Repetition
Fall 2013

Important note: The course material necessary for completing this lab will be found in your lecture notes and in the Carter text up to the end of Section 4.6.

You must use the `submitaps105f` command to electronically submit your solution by 11:59 p.m. on **Saturday, October 19, 2013**.

Objective

In this lab you will be writing more complex C programs that use what you have learned in *Chapter 3: Decision Making* and *Chapter 4: Repetition*. Input and output must be done using the functions `scanf` and `printf`.

Part 1 — Rating Movies

The website Rotten Tomatoes (http://www.rottentomatoes.com) is a movie reviewing web site that summarizes the reviews of recent movies by looking at all of the reviews written by critics in newspapers and online. They calculate the percentage of movie critics that liked the movie. That percentage gives a better sense of how good the movie is compared to looking at just one review because it reflects the combined opinion of many people. (At least I tend to think so; though for some reason Iron Man 3 was highly rated with 78% of critics liking it!). The website categorizes any movie that receives a rating of 60% or higher as "FRESH" and anything less than that as "ROTTEN."
You are to write a C program (in a file called `Lab4Part1.c`) that reads in the ratings for a series of movies, as integers, and outputs the following three things:

1. The movie number - beginning with 1, and adding 1 for each movie entered.
2. A conversion of the percentage rating into a number of 'stars' as described in Table 1 below, printing that number as an integer.

Rating Range	Number of Stars
0-19	0
20-39	1
40-59	2
60-79	3
80-100	4

Table 1: Rating to Star Conversion

3. In addition, you should also print out the number of stars (*) explicitly, in the manner shown in the examples below.

Valid input to the program should be a positive integer from 0 to 100, which represents the rating percentage of each movie. The program should repeatedly prompt the user for a positive integer value, and in response to user input, should print the movie number, original rating, star number and a number of asterisks (*) corresponding to that number, as illustrated below.

The program should repeatedly prompt the user for additional user input, until an invalid input is entered, after which the program should exit and print ”Goodbye.” An invalid input would be a negative number or one that is greater than 100. It can be assumed that the user always enters integer values when prompted.

The source code of your program should be put in a file called `Lab4Part1.c`.

An example run of the program is shown below. The values `23`, `50`, `75`, `85` and `-1` are entered by the user in the example run. Given the same user input, your output should match the following output *exactly*, including all punctuation, and all wording (including the singular or plural form as needed). Any variation from this will result in a loss of marks.

```
Enter movie rating from 0 to 100: 23
Movie number 1 rated at 23% has 1 star: *
Rotten Tomatoes says this movie is ROTTEN

Enter movie rating from 0 to 100: 50
Movie number 2 rated at 50% has 2 stars: **
Rotten Tomatoes says this movie is ROTTEN

Enter movie rating from 0 to 100: 75
Movie number 3 rated at 75% has 3 stars: ***
Rotten Tomatoes says this movie is FRESH

Enter movie rating from 0 to 100: 85
Movie number 4 rated at 85% has 4 stars: ****
Rotten Tomatoes says this movie is FRESH

Enter movie rating from 0 to 100: -1
Goodbye.
```

Part 2 — Semiprime Numbers

As you will recall, a prime number is a number is that is divisable only by the number 1 and itself. A *semiprime* number is positive integer that is the product of two (not necessarily different) prime numbers. The first few semiprimes are 4 ($= 2 \times 2$ and 2 is prime), 6 ($= 2 \times 3$ both prime), 9, ($= 3 \times 3$), 10, ($= 2 \times 5$), 14, 15, 21, 22, 25, 26. Semiprimes have applications in cryptography (in the public key system, if you've heard of that) and were used in a message sent by Earth to Aliens! (The 1974 Arecibo message was sent with a radio signal aimed at a star cluster. It consisted of 1679 bits intended to be interpreted as an image of size 23 pixels by 73 pixels. The number $1679 = 23 \times 73$ was chosen because it is a semiprime and therefore can only be broken down into 23 rows and 73 columns, or 73 rows and 23 columns. See `http://en.wikipedia.org/wiki/Arecibo_message` for details.)

In this assignment, it will be useful to know that the factors of semiprime numbers are *only* 1, the number itself, and the two other prime factors. For example, the number 26 is semiprime because its only factors are 1, 2, 13, and 26.

You are to write a C program that receives as input a series of integers and determines if each is a semiprime number. If it is semiprime, the program will output the prime numbers, that when multiplied, give the input number. If it is not a semiprime number, your program should simply state that it is not a semiprime.

If the input to the program is zero or a negative number, your program should exit and print "All Done." It can be assumed that the user always enters integer values when prompted.

In this part of the lab, you are not allowed to use any math library functions other than `sqrt`, or to use material after Section 4.6 of the Carter text.

The source code of your program should be put in a file called `Lab4Part2.c`.

Shown below are example runs of the program. Given the same user input, your output should match the output *exactly*, including all punctuation, and all wording. Any variation from this will result in a loss of marks. The values `4`, `5`, `10`, `...` are entered by the user in the example run.

```
Enter input integer above 0: 4
The Integer 4 is Semiprime, with the two primes being 2 and 2
Enter input integer above 0: 5
The Integer 5 is not Semiprime
Enter input integer above 0: 10
The Integer 10 is Semiprime, with the two primes being 2 and 5
Enter input integer above 0: 24
The Integer 24 is not Semiprime
Enter input integer above 0: 25
The Integer 25 is Semiprime, with the two primes being 5 and 5
Enter input integer above 0: 187
The Integer 187 is Semiprime, with the two primes being 11 and 17
Enter input integer above 0: 1234568609
The Integer 1234568609 is Semiprime, with the two primes being 103 and 11986103
Enter input integer above 0: 0
All Done.
```

Marking

This lab will be marked out of 10. Part 1 count for 3 marks and Part 2 will count for 7 marks. Full marks are given if your program works correctly, fewer if not, and zero if it cannot be compiled. Late submissions or submissions with an incorrect filename will result in a mark of 0 for the entire lab. The deadline is strictly enforced, so avoid last minute submissions.

You can run a testing program, called `tester`, yourself to test the correctness of your solution. At the command line in your ECF account, run:

```
/share/copy/aps105f/lab4/tester
```

in the same directory as your solution file. The testing program will use a number of test cases to test your solution, and report `success` if your solution produces output that is **identical** to the expected output.

A marking program, called `marker`, will be used to automatically mark your lab. The marking program will use **the last version** of the lab files you submitted. Some of these test cases will be used by the marking program as well, but the marking program will also be using other test cases that are not included in the testing program to test the correctness of your program. This implies that even though you do not have access to the marking program before the submission deadline, you will obtain at least partial marks if all of the test cases in the testing program report `success` with your solution. After the submission deadline, you can access the marking program by running the following command at the command line in your ECF account:

```
/share/copy/aps105f/lab4/marker
```

The mark you receive for each part will be the same as reported by the marking program for the file you submit. Your mark for this lab will contribute 3% to your mark in the course.

What To Submit

When you have completed the lab, use the command

```
    % submitaps105f 4 Lab4Part1.c Lab4Part2.c
```

within the directory that contains your solutions to submit your files. Make sure you name your files exactly as stated (including lower/upper case letters). Failure to do so will result in a mark of 0 being assigned. You may check the status of your submission using the command

```
    % submitaps105f -l 4
```

where `-l` is a hyphen followed by the letter '*ell*'.

You can also download a copy of your submission by running the command:

```
/share/copy/aps105f/lab4/viewsubmitted
```

Good Luck!