

# CS 161 Program Four

9 Points

## Purpose

This assignment introduces the use of functions in programming. When you completed this program you will have an understanding of how to break down a problem solution into logical pieces and how to identify code that is used in several places and therefore belongs in a separate function. You will know how to write functions and interpret and debug a program comprised of several functions.

## Scenario

As you probably know, eggs are graded before going to market. The main consideration in grading is freshness, with the freshest eggs being graded “AA.” For the purposes of this assignment we will assume that freshness is the *only* consideration, and that freshness is determined *only* by the time that has passed since the egg was laid. (In reality egg grading is much more complicated—google “egg grading” for a more complete, and thoroughly confusing explanation.)

## The Problem

Imagine that you are working in a facility that grades lots of eggs. Some of the lots are coming from egg farms at various distances, and some are coming from supermarkets where the expiration date for their current grade has expired. (When grade AA eggs reach their expiration date, they are often re-graded for sale as grade B or for institutional use.)

Your program asks the user to enter today’s date, and then asks for the lot number and “birth date” of lots of eggs. When the user enters “Q” or “q” for a lot number, the program produces a report showing the current date and the lot number, “birth date” and grade for each lot of eggs.

This task has several components, some of which are needed in more than one place in the code. Outline the process in general terms to identify the logical chunks of code to make into functions.

## Determining Age

To determine the age of a lot of eggs, you need the Julian date for the current date and for the date the eggs were laid. The Julian date is derived by adding the “day” part of a date to the number of days that preceded the month. So the Julian date for any day in January is the same as the day; in February it is the day added to 31; in March it is the day added to 59, and so on. The only trick here is that if the year is a leap year and the month is after February, then the Julian date is one more than it would normally be.

Calculating age is simply a matter of subtracting the Julian birth date from the Julian current date—and taking into account the years.

## Program Requirements

As with the previous two assignments, this one has different stage of completion worth various amounts of credit. Start by solving the simplest version, and then save a backup and work on the more complex versions. Turn in the best *complete* version you finish.

## CS 161 Program Four

For any version, your program should show the correct header and report all lot numbers, and “birth dates.” The intermediate and advanced versions will also show ages in days, and correct grades.

All versions will use functions to eliminate redundant code and to streamline the program.

Lot numbers are strings of up to six characters.

The header includes the current date in its second line.

The dates as shown in the report are strings formatted as MM/DD/YYYY.

Prompts for input should not be vague. See the example input sequence below.

The program **and** each function should be documented. Function documentation should identify any parameters and the value returned (if any). Inadequate or sloppy documentation will result in a loss of up to 1/3 credit.

### *Basic Version—Complete for 4-1/2 Points Maximum*

This version does not compute the ages of the eggs or grade them. It produces a report showing the lot numbers and their birth dates, preceded by the formatted header.

### *Intermediate Version—Complete for 7-1/2 Points Maximum*

This version computes and reports the age in days and the grade for each lot of eggs. Its computation of age should be correct even if the eggs were laid in the previous year. The age of eggs laid more than a year earlier may be a day or two off due to intervening leap years, but since they will be really gross and graded “unfit” it doesn’t matter.

Grades: if the eggs are less than 21 days old, they are grade AA, if they are 21 to 39 days old, they are grade A, if they are more than 39 days but less than 60, they are grade B, and if they are 60 or more days old they are “Unfit for human consumption.” (Remember this is a bogus representation of egg grading.)

The formatted date for this version has a leading zero for any month or day less than ten, *ie.* “01/01/2007” and not “1/1/2007.”

### *Advanced Version*

One assumption all of our programs have made so far is that users don’t make data entry errors. This is taking a lot on faith. This version of the program will not accept unreasonable month or day entries, but will instead require the user to re-enter the value until it is reasonable. Note that if the user has entered **4** for the month, **31** is not reasonable for the day.

Do not worry about handling leap year for the date validation, as this would require the user to enter the year first—something users would not adapt to easily.

### **Sample Input Sequence**

```
Please enter the month for Today's Date2
Please enter the day for Today's Date16
Please enter the year for Today's Date2007
Please enter the next lot number: Z23-14
Please enter the month for the date lot Z23-14 was laid: 1
Please enter the day for the date lot Z23-14 was laid: 25
Please enter the year for the date lot Z23-14 was laid: 2007
```

CS 161 Program Four

**Sample Output**

===== Egg Grading Report =====

02/02/2007

Lot	Date	Age	Grade
Z23-14	01/25/2007	22	A
Z23-35	02/01/2007	15	AA
Z17-32	01/02/2007	45	B
Y17-42	12/21/2006	57	B
Y15-27	11/30/2006	78	Unfit for human consumption