Assignment 4 Description

What's your function?

The secret of life is to have no fear; it's the only way to function.
Stokely Carmichael

Due date: February 24th 2016, 6:00 pm
(please note that you shouldn't work on this assignment during the reading week)

Notes:

- **This assignment is individual work**. You may discuss the questions and problems with anyone, but the work you hand in for this assignment should be your own work.
- **Each question indicates what to hand in.** Often, it will be a single document for **each question**. You must give your document the name we prescribe for each question, usually in the form **aNqM,** meaning Assignment N, Question M. You should make sure your name and student number appear at the top of every document you hand in. These conventions assist the markers in their work. Failure to follow these conventions will result in needless effort by the markers, and a deduction of grades for you. Do not submit folders, zip documents, even if you think it will help.
- **Hand in your work using Moodle** It is advisable to hand in each answer that you are happy with, as you go. You can always revise and resubmit as many times as you like before the deadline; only your most recent submission will be graded.
- Questions are annotated use descriptors like "easy" "moderate" and "tricky". All students should be able to obtain perfect grades on "easy" problems. Most students should obtain perfect grades on "moderate" problems. The problems marked "tricky" may require significantly more time, and only the top students should expect to get perfect grades on these. We use these annotations to help students be aware of the differences, and also to help students allocate their time wisely. Partial credit will be given as appropriate, so hand in all attempts.
- **How to find your C++ programs to submit to Moodle for grading.**
    - When you hand in your C++ programs, hand in the .cpp file only, not the whole Eclipse project.
    - To find your .cpp files that you create as part of your projects, open up Explorer/Finder/Dolphin and navigate to the folder/directory that you selected as your 'workspace.' Within that folder you'll see folders for each of the projects that you created. Navigate into the appropriate one, and then into the "src" folder; you'll see your .cpp file there.

Question 1 (13 marks)

Purpose: Practice with getting data to and from functions.

Degree of Difficulty: Moderate

At the local supermarket, hotdogs are sold in packages containing a dozen (12) dogs, and yet hotdog buns are sold in packages containing 8 buns. One question we can ask is this: given some number of packages of dogs and buns, how many hotdogs can we make (one dog per bun)? In this problem, you'll answer that question.

Here is a description of **what** your program should do. After that is a description of **how** you should write the program.

1. Ask the user for the number of hotdog *packages* purchased.
2. Ask the user for the number of hotdog bun *packages* purchased.
3. If there are leftover dogs or buns, display what is left over, and how many.
4. Display to the console the number of hotdogs you can make (one dog per bun) without buying more dogs or buns.

For example, if you have 2 packages of dogs (24 dogs total), and 3 packages of buns (24 buns total), you can make 24 hotdogs, and nothing left over. But if you have 1 package of dogs, and 2 packages of buns, you can only make 12 hotdogs, but you will have 4 buns left over.

To solve this problem, you should write a function called **hotdog**(), which takes as input (e.g. function arguments):

1. The number of *packages* of dogs (12 dogs per package)
2. The number of *packages* of buns (8 buns per package)

Your **hotdog() function** should do the following steps:

1. Calculate how many complete hotdogs that can be made (1 dog per bun)
2. Calculate how many dogs or buns are left over (you can have zero leftovers)
3. If there are leftover dogs or buns, **display** (to the console) what is left over, and how many (display nothing if there are no leftovers).
4. Return the **number of complete hotdogs** you can make.

In **main()**, you should write code that does the following:

1. Ask the user for the number of hotdog *packages* purchased.
2. Ask the user for the number of hotdog bun *packages* purchased.
3. Call the function hotdog(), providing the correct information, and storing the result

4. Display to the console the number of hotdogs you can make (one dog per bun) without buying more dogs or buns.

Here's an example of how the program could work:

```
Wasteful Bro's Hotdog Restaurant Kitchen app!
Please enter how many packages of dogs you have: 1
Please enter how many packages of buns you have: 2
We have 4 buns leftover!
We can make 12 hotdogs!
```

Or this:

```
Wasteful Bro's Hotdog Restaurant Kitchen app!
Please enter how many packages of dogs you have: 2
Please enter how many packages of buns you have: 3
We can make 24 hotdogs!
```

You should prepare 6 examples to demonstrate that your program works, as follows:

1. Two different examples where there is nothing left over
2. Two different examples where there are dogs left over, but no buns left over
3. Two different examples where there are buns left over, but no dogs.

You should prepare these in advance, so that you can judge whether your program gets the answers right! It is of course *impossible* to have dogs **and** buns left over at the same time.

What to hand in:

1. Your C++ code in a file called **a4q1.cpp**. Be sure to include your name, NSID, Student Number, lecture section and tutorial section in comments at the top of your file.
2. Copy/paste your 6 examples of your testing, in a text file called **a4q1_examples.txt** (.rtf, and .doc are also okay).

Evaluation:

- 0/13: Nothing submitted, or files cannot be opened.
- -1 marks: for lack of identification in the files (Name, NSID, Student number, Tutorial section, and lecture section).
- 8 marks for the function:
  - 3 marks: a correct header;
  - 2 marks for the correct calculation of hotdogs;
  - 2 marks for identifying leftovers
  - 1 mark for returning the number of hotdogs
- 2 marks: Appropriate function call in `main()`

- 3 marks: You demonstrated that your program works correctly by copy/pasting the output of your program using the 6 examples you prepared.