Week 3 Exercises : Expressions and Functions

Jan 30th 2012

This week we build on the experience in writing expressions from last week. The focus is on translating mathematical formula into code. The new tool that you use this week is the standard set of maths functions provided in C++. We have tried to pick examples that will be familar to you, but the important thing is not what the equations calculate, it is the process of writing code to evaluate them.

1 Standard Exercises

1. The Hamming Weight of a value is the number of bits that are set to one in its binary representation. For example $H(3) = 2, H(14) = 3, H(16) = 1$. Write a program that inputs an integer (assuming it is in the range 0..255) and outputs the Hamming Weight. Think about how this problem can be split into simpler parts. There is a counting part of the problem, and a bit-checking part of the problem. You should consider masking and shifting approaches.

2. a) A vector is a collection of real numbers such as [2.34 5.12]. Pick a way to represent a vector with two components (a 2-vector) in a program. Write a program that inputs the two components and outputs the normalised vector. Remember that the normalisation of a 2-vector
$$|[x\ y]| = \frac{[x\ y]}{\sqrt{x^2 + y^2}}$$
b) Convert your program to one that operates on 3-vectors.

3. Entropy is a measure of information content. It is commonly used in communications and compression where the probability of each symbol occuring in a message is used to define the number of bits needed to encode the message. Given a series of n symbols where the probabilities of each symbol occuring is $p_1, p_2 \ldots p_n$, the Shannon Entropy is defined as
$$-\sum_{i}^{n} p_i \cdot \log_2(p_i)$$
e.g. in the case that $n = 2$ the entropy is $-(p_1 \cdot \log_2(p_1) + p_2 \cdot \log_2(p_2))$
a) Write a program that inputs two probabilities and outputs the entropy.
b) Write a program that inputs three probabilities and outputs the entropy.

4. Cartesian coordinates are the common $[x\ y]$ form that measures distance along the x axis and the y axis. Polar coordinates $[r\ \theta]$ measure the angle θ and distance r. The two forms are related by the equations
$$x = r \cdot \cos(\theta)$$
$$y = r \cdot \sin(\theta)$$
$$\theta = r \cdot \tan^{-1}(\frac{y}{x})$$
$$r = \sqrt{x^2 + y^2}$$
Some of these functions map directly into C++, and others have slightly more complicated usage. The reference material from the slides will offer some guidance. Think about simple test cases to check that the programs in this question work correctly.
a) Implement a program that inputs r and θ and outputs x and y.
b) Implement a program that inputs x and y and outputs r and θ.

5. a) Write a program the inputs the values of a 2-vector and a $2\times$ matrix and outputs the product of multiplying them together.
   b) Write a program the inputs the values of a 2-vector and a $2\times$ matrix and outputs the product of multiplying them together.

6. Compound interest is simple in the case that money is desposited in a bank at the beginning and a regular interest payment is made, i.e. x is the initial deposit and r is the interest rate (e.g. $r = 1.03$ for a 3% rate). After n years the account value is $x \cdot r^n$. But things are more complex if money is deposited in the account between interest payments. If d is the amount deposited each year then the balance of the account y_i would be

$$y_0 = x$$

$$y_1 = x \cdot r + d$$

$$y_2 = y_1 \cdot r + d \ldots$$

   a) Write a program that performs the simple case, i.e. input the initial deposit, the interest rate and the number of years and outputs the final total.
   b) Write a program that performs the complicated case when the number of years $n = 5$. Input the initial balance, an interest rate and a desposit amount. The program should calculate the balance of the account after five years. To make this straight-forward the structure of your program should mirror the equations above.
   c) Convert the program to use a single assignment statement to make the same calculation, i.e. substitute the values of the different variables into a single expression.

2 Extension exercises

We have not listed the background material necessary for these problems, or the hints about how to solve them. In order to find the background you will need to search for material on the web. For hints on the technique you can ask in class.

7. * Read the three coefficients of a quadratic polynomial and calculate the real roots.

8. * Calculate the inverse of a 3x3 matrix. Read the 9 values from standard input, and use a double for each matrix element. Try your program with singular and non-singular matrices. You can verify your results in the following website: http://www.jimmysie.com/maths/matrixinv.php

9. * Write a program that inputs two integer values and outputs the product of the two numbers multiplied together. You may assume that both integers are in the range 0..255. Your program is not allowed to use the multiplication operator $*$. This requires some sophisticated use of the binary operators. There are two pieces of information that you will find useful.
   a) Work out the values of powers of two with one subtracted, and look at their binary representation: $2 - 1 = 1$,$4 - 1 = 3$,$8 - 1 = 7$,$16 - 1 = 15 \ldots$ Think of these numbers as masks as described in this week's lecture.
   b) Think about the relationship between multiplying $x \cdot \{0, 1\}$ and x AND$\{0, 255\}$.