CS 210 - Fundamentals of Programming I
Fall 2011 – Programming Assignment 6
20 points

Out: November 2, 2011
Due: November 14, 2011 (Monday)

Note: Although this assignment is due after Written Exam 2, the material used in this assignment will be on the exam. Please do not wait until after Written Exam 2 to start this assignment.

Problem Statement
In computer graphics, direction is represented using mathematical vectors of three dimensions written as < x, y, z >. For vectors **u** and **v** represented as < u_x, u_y, u_z > and < v_x, v_y, v_z >, respectively, the following arithmetic operations on 3-D vectors are defined:

Operation name	Operator notation	Result
Addition	**u** + **v**	$\langle u_x+v_x, u_y+v_y, u_z+v_z \rangle$
Subtraction	**u** – **v**	$\langle u_x-v_x, u_y-v_y, u_z-v_z \rangle$
Scalar multiplication	$k \cdot$ **v**	$\langle k \cdot v_x, k \cdot v_y, k \cdot v_z \rangle$
Dot product	**u** · **v**	$u_x \cdot v_x + u_y \cdot v_y + u_z \cdot v_z$
Cross product	**u** × **v**	$\langle u_y \cdot v_z - u_z \cdot v_y, u_z \cdot v_x - u_x \cdot v_z, u_x \cdot v_y - u_y \cdot v_x \rangle$
Magnitude	| **u** |	$\sqrt{u_x^2+u_y^2+u_z^2}$

A *unit* vector is defined as a vector with magnitude of 1. The cross product of two vectors is a vector that is said to be *orthogonal* to both argument vectors.

Assignment
Note: because this assignment will be graded by linking it with a driver program written by the instructor, all names and identifiers given below (in **`Courier`** font) must be used **exactly** as shown. In addition, function prototypes must declare any parameters in **exactly** the order listed. If this is not done, the submission system will not be able to compile your library and test it, and the submission will not be graded.

The assignment is to implement a personal library for a type and functions to model 3-D vectors and their operations. The type definition and function prototypes must be stored in a header file named **`vector.h`**, and the function definitions must be stored in a source file named **`vector.c`**, in the same manner as the rational number library completed in the in-class exercise. The vector library must meet the following design specifications:

1. The type of a vector must be called **`vector_t`**. It must have three double-precision components named **`x`**, **`y`**, and **`z`**.
2. A function **`make_vector`** must be provided. This function receives three double values representing the x, y, and z components of a vector and returns a **`vector_t`** result with those component values.
3. A function **`vec2a`** must be provided. This function receives a **`vector_t`** object and passes back a string

containing the string representation of the vector. The format of the string representation must be **<x,y,z>** with no spaces and three places past the decimal point. E.g. **<3.000,-4.000,5.125>**.

4. Functions for the arithmetic operations defined above must be provided:
   - **vec_add** that receives two vectors and returns their sum
   - **vec_sub** that receives two vectors and returns their difference
   - **vec_scale** that receives a vector and a scaling value, and returns the vector multiplied by the scaling value
   - **vec_dot** that receives two vectors and returns their dot product
   - **vec_cross** that receives two vectors and returns their cross product
   - **vec_magnitude** that receives a vector and returns its magnitude
5. The following additional functions must be provided:
   - **vec_unitvector** that receives a vector and returns a unit vector that represents the same direction as the argument
   - vec_unitnormal that receives two vectors and returns a unit vector that is orthogonal to both of the them.

Notes:

- Although the submission system will provide its own main program, you should write your own driver program that tests your vector library.
- It is recommended that each function be tested thoroughly after it is written. In other words, do not try write all of the library code first, then try to test it.
- Thorough testing is more than one test.

REMINDER: Your program must compile for it to be graded. Submissions that do not compile will be returned for resubmission and assessed a late penalty. Submissions that do not substantially work also will be returned for resubmission and assessed a late penalty.

Follow the program documentation guidelines in the *C Programming Style Guideline* handout. As stated in the syllabus, part of the grade on a programming assignment depends on how well you adhere to the guidelines. The grader will look at your code and grade it according to the guidelines.

What to Submit

Electronically submit a zipfile containing files **vector.h** and **vector.c** as explained in the handout *Submission Instructions for CS 210*. To zip multiple files together, select all files using Ctrl-Left-Click, then Right-Click to zip as usual. The submission system will start accepting assignments on Wednesday, November 9.