

ENGINEERING H192
DAILY ASSIGNMENT B14

In this assignment you will write a program that uses pointers and arrays to perform two fundamental mathematical operations (the dot product and the cross product) on 3D vectors.

A 3D, or spatial, vector is comprised of three components, commonly referred to as the x , y , and z directions, which provide the magnitude and direction of the vector. In proper vector notation, these are written as $[x, y, z]$. 3D vectors may be multiplied together using either a **dot operator** (\cdot) or a **cross operator** (\times). The dot product results in a scalar while the cross product results in a vector orthogonal to the two original vectors. Specifically, for two vectors \mathbf{U} and \mathbf{V} :

$$\begin{aligned} \mathbf{U} &= [x_1, y_1, z_1] \quad \mathbf{V} = [x_2, y_2, z_2] \\ \mathbf{U} \cdot \mathbf{V} &= x_1x_2 + y_1y_2 + z_1z_2 \\ \mathbf{U} \times \mathbf{V} &= [(y_1z_2 - z_1y_2), (z_1x_2 - x_1z_2), (x_1y_2 - y_1x_2)] \end{aligned}$$

Write a complete C program, that includes a **main()** function, to collect user input and control the overall flow of your program, and two user-written functions, **mydot()** and **mycross()**, to perform the dot product and cross product operations, respectively. Arrays with three elements will be used for the vectors, such that **vector1[0]** is the x-component of **vector1**, **vector1[1]** is the y-component, and **vector1[2]** is the z-component.

The finished program must:

- 1) Prompt the user to input two vectors, component by component
- 2) Read the values entered by the user into the vector arrays
- 3) Allow the user to choose to calculate either the dot product ('d'/D) or the cross product ('c'/C)
- 4) Use addresses to pass the vectors to **mydot()** and **mycross()** in the format specified by the function prototypes below
- 5) Display the original vectors and either the resultant dot or cross product, depending on the user's selection
- 6) Run indefinitely until the user chooses to quit

The result of **mydot()** will be a single scalar value returned to the calling function with a **return** statement. The function **mycross()**, however, will require that the resultant vector be returned to the **main()** function using an address, as indicated by the third parameter in the **mycross()** prototype below. The **required** function prototypes are:

```
float mydot (float [], float []);  
void mycross (float [], float [], float *);
```

Save your program in a file named **b14.cpp**. Compile, link, test, and debug it. When it is running without error and producing correct results, modify **b14.cpp** so that everything that is being written to the screen is also written to a result file, **b14result.txt**. When the modified **b14.cpp** is running correctly, submit **b14.cpp**, **b14result.txt**, and this sheet.

Name _____ Instructor _____ Seat _____ Hour _____