

KHBJ Maths2020

by Kai Bentley - jennings

FILE	11982_KAI_BENTLEY_- _JENNINGS_KHBJ_MATHS2020_250776_699801439.DOCX (215.94K)		
TIME SUBMITTED	01-MAY-2020 05:12PM (UTC+0100)	WORD COUNT	3907
SUBMISSION ID	125873910	CHARACTER COUNT	18034



APPLICATION OF MATHS FOR SOFTWARE DEVELOPMENT

Kai Hakkeem Bentley - Jennings
1807048@student.uwtsd.ac.uk

1 Table of Contents

2	Table of Figures.....	2
3	Introduction	3
4	Main Body	4
4.1	Calculating the Length of a Vector & Normalization	4
4.2	Finding the Distance Between Two Points	6
4.3	Calculating the Dot Product of Two Vectors.....	7
4.4	Calculating the Cross Product of Two Vectors	9
5	Conclusion & Appendices	10
5.1	Conclusion	10
5.2	Appendix	11

2 Table of Figures

Figure 1: Program Menu detecting invalid input.....	3
Figure 2: 2D Vector Length Menu with prompt and result.....	Error! Bookmark not defined.
Figure 3: 3D Vector input and output	5
Figure 4: Example of Y coordinates set the same	6
Figure 5: Display of all coordinates different.....	6
Figure 6: Dot Product Menu	7
Figure 7: Dot Product of two 3D vectors	7
Figure 8: Attempt at calculating Dot Product using $\cos(90)$	8
Figure 9: The matrix operation display, input and result	9

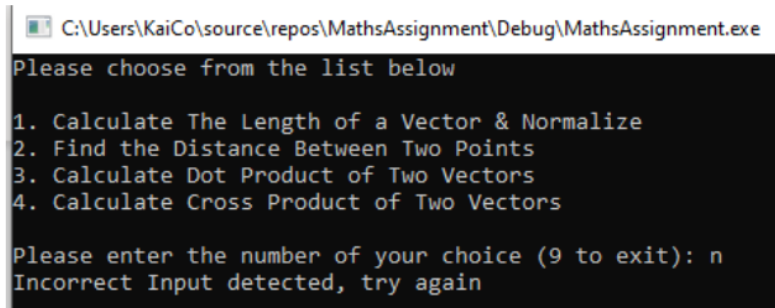
3 Introduction

This report will attempt to fulfil the requirements of creating a maths library software application. Contained in this report are the created functions of the following:

- Vector Length
- Normalization of a Given Vector
- Returning the Distance Between Two Points
- Two Vectors & the Dot Product Result
- Two Vectors & the Cross-Product Result

Screenshot evidence of each process will be made and the description of the process as well. The mathematical theory behind each process will be made clear at the beginning of each appropriate subtitle.

A menu using a switch is implemented to avoid the user from having to run through each exercise until they reach their desired task, and for repeated troubleshooting to test if calculations are correct. The program will also be using validation for each input to prevent the process from looping endlessly, forcing a shutdown. Instead the program will skip and ignore character inputs and ask for you to try again. Values of consideration will be set to only show two decimal points when displaying the double/float.



```
C:\Users\KaiCo\source\repos\MathsAssignment\Debug\MathsAssignment.exe
Please choose from the list below

1. Calculate The Length of a Vector & Normalize
2. Find the Distance Between Two Points
3. Calculate Dot Product of Two Vectors
4. Calculate Cross Product of Two Vectors

Please enter the number of your choice (9 to exit): n
Incorrect Input detected, try again
```

Figure 1: Program Menu detecting invalid input

4 Main Body

4.1 Calculating the Length of a Vector & Normalization

Both the Length of the vector and its Normalization will be combined into one process, making things smoother and more manageable.

In order to begin calculating the Vector Length, the correct formula must be considered. For calculating Vector Length, it comes as: $|a| = \sqrt{x^2 + y^2}$.

And for Normalizing the Vector: $\hat{a} = (x/|a|, y/|a|)$

These formulas however consider Vectors that are only two dimensional. For a 3D Vector, one must add the z axis into the equation as well. The program will be configured to prompt the user what dimension their vector is.

```
Calculating Vector Length & Normalization
Is your Vector...
1. 2D (Two Dimensional)
2. 3D (Three Dimensional)

Please enter the number of your choice (Any other number to exit): 1

Please enter your X coordinate: 9
Please enter your Y coordinate: 4

The Length of Your Vector is: 9.85
And When Normalized is: 0.91, 0.41
```

Figure 2: 2D Vector Length Menu with prompt and result

The above figure will have the screen cleared upon the user's choice at the initial menu. Once in this menu the user will have been prompted to choose the dimension of their vector, where they will then start to input the coordinates of their vector.

Using the above formulas, the program should output the correct length for the vector, alongside the normalized values. An alternate scenario below – Figure 3 – shows entering in the extra z axis for three dimensional vectors.

```
Calculating Vector Length & Normalization  
Is your Vector...  
1. 2D (Two Dimensional)  
2. 3D (Three Dimensional)  
Please enter the number of your choice (Any other number to exit): 2  
Please enter your X coordinate: 9  
Please enter your Y coordinate: 4  
Please enter your Z coordinate: 7  
The Length of Your Vector is: 12.08  
And When Normalized is: 0.74, 0.33, 0.58
```

Figure 3: 3D Vector input and output

4.2 Finding the Distance Between Two Points

For calculating the distance between the two points, the function shall use: $\sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Using this formula will provide the distance between the two sets of coordinates. Theoretically, should one set of the axis rest on the same point i.e. y_1 and $y_2 = 0$, then it is merely a case of taking away one x coordinate from the other.

```
The Distance Between Two Points

Please enter your first set of coordinates
Please enter X1:5
Please enter Y1:0

Please enter your second set of coordinates
Please enter X2:8
Please enter Y2:0
The Distance Between the inputted points is: 3.00
```

Figure 4: Example of Y coordinates set the same

Above shows how the distance between two different points can be as simple as taking away from one set of coordinates should the other be the same value. This display shows that with the inputted formula it works as it should for one set of coordinates. This is still possible from using the same formula, rather than making an exception as they both come to the same result. Below in figure 5 shows the result when each axis is different from a similar axis.

```
The Distance Between Two Points

Please enter your first set of coordinates
Please enter X1:6
Please enter Y1:9

Please enter your second set of coordinates
Please enter X2:3
Please enter Y2:4
The Distance Between the inputted points is: 5.83
```

Figure 5: Display of all coordinates different

4.3 Calculating the Dot Product of Two Vectors

The dot product of vectors can be calculated either by:

- $a.b = (x1 * x2 + y1 * y2...)$
- $a.b = |a| * |b| * \cos(\theta)$

The function will attempt to calculate both, giving the user a bit more option when wanting to calculate the dot product. Figure 6 below will prompt the user to select their choice, and to clarify will show the formula with which the function will use to calculate the dot product for the user.

```
Calculating the Dot Product of Two Vectors

Do you wish to calculate with...

1. a.b = (x1 * x2) + (y1 * y2)...
2. a.b = |a| * |b| * cos(theta)
Please enter the number of your choice (Any other number to exit):
```

Figure 6: Dot Product Menu

If the user should choose option one, they will also receive the option of whether their Vectors are two dimensional or three dimensional. Option doesn't fully show the possible formula but implies that the z axis is part of it with the trailing dots after the y axis multiplication.

```
Calculating the Dot Product of Two Vectors

Do you wish to calculate with...

1. a.b = (x1 * x2) + (y1 * y2)...
2. a.b = |a| * |b| * cos(theta)
Please enter the number of your choice (Any other number to exit): 1

The Type of Vectors is?...

1. 2D Vectors
2. 3D Vectors

Please enter the number of your choice (Any other number to exit): 2

Please enter your first set of coordinates
Please enter X1:5
Please enter Y1:6
Please enter Z1:7

Please enter your second set of coordinates
Please enter X2:9
Please enter Y2:3
Please enter Z2:4
The Dot Product of these Two Vectors is: 91.00
```

Figure 7: Dot Product of two 3D vectors

```
Calculating the Dot Product of Two Vectors
Do you wish to calculate with...
1. a.b = (x1 * x2) + (y1 * y2)...
2. a.b = |a| * |b| * cos(theta)
Please enter the number of your choice (Any other number to exit): 2
Please enter Vector A's Length
9.84
Please enter Vector B's Length
7
Please enter the Angle
90
The Dot Product of these Two Vectors is: -20.63
```

Figure 8: Attempt at calculating Dot Product using $\cos(90)$

When the user selects the second option, they will be required to input the length of their respective vectors and the angle between them. Using $\cos(\theta)$, the program would then send back the correct dot product of these vectors. However this is not the case, as using $\cos(90)$ should give back zero as the result. Instead it is to be believed that it is taking the angle inputted as radians and spitting out a false answer. Attempts to correct this hasn't worked out and now remains in the hope of eventually finding the correct solution.

4.4 Calculating the Cross Product of Two Vectors

For calculating the cross product of two vectors, a formula of the like will be used:

$$A = A_i + B_j + C_k$$

$$B = D_i + E_j + F_k$$

However, for this operation it will need to be converted into a matrix style formula, of which becomes:

$$a \times b = \begin{vmatrix} i & j & k \\ A & B & C \\ D & E & F \end{vmatrix}$$

Finally, the following formula below is how the function will calculate the cross product, with the lower most showing how it will read it back to the user:

$$\begin{aligned} a \times b &= (BF - EC)i - (AF - DC)j + (AE - DB)k \\ &= xi - yj + zk \end{aligned}$$

With this information in mind, the user must be shown the matrix in question and to enter their corresponding number into the letter of the matrix. It will then calculate i, j, and k separately and output them in a final sentence stating their values. However, j must have a negative modifier in front as it must be subtracted from the equation.

```
Calculating the Cross Product of Two Vectors

This program works out the answer using the formula of:
| i j k |
| A B C | = a x b
| D E F |

Please enter A:2
Please enter B:-1
Please enter C:3
Please enter D:5
Please enter E:7
Please enter F:-4
The Cross Product of these Two Vectors is = -17i 23j 19k
```

Figure 9: The matrix operation display, input and result

The user upon entry of the function will be greeted with how the matrix looks like, and afterwards will input their integers into the following 'character'. The result shown in this image is correct after working it out manually, as without the negative modifier in front of the j value it would display as: '-23j'.

5 Conclusion & Appendices

5.1 Conclusion

In conclusion, the maths library that has been attempted is bare and minimalistic. The issue with calculating $\cos(\theta)$ was incredibly frustrating and if more time were at hand, would have likely been fixed as well as an attempt to fulfil the rest of the criteria as outlined. Some research could've have gone a long way towards achieving these goals also. Perhaps also making the validation a function would've have been a far more reasonable choice as it appears to clutter the code more than it should have to be.

Below in the appendices is the code of the attempted library.

5.2 Appendix

```
#include <string>
#include <iostream>
#include <sstream>
#include <windows.h>
#include <iomanip> //Precise Values
#define PI 3.141592653589793238463 //To convert theta from radians to degrees work with cos()

using namespace std;

//Seperate Functions to keep main clearer
void VectorLengthNormal(void);
void Distance2Point(void);
void DotProduct(void);
void CrossProduct(void);

int main(int argc, char** argv)
{
    //Menu Function

    int choice = 0; //Initialized to hopefully avoid errors
    string input;
    stringstream stream;

    while (choice != 9) //User comes back to main after function choice is completed unless 9 is
chosen
    {
        cout << "Please choose from the list below" << endl << endl;

        cout << "1. Calculate The Length of a Vector & Normalize" << endl;
```

Application of Maths for Software Development

```
cout << "2. Find the Distance Between Two Points" << endl;
cout << "3. Calculate Dot Product of Two Vectors" << endl;
cout << "4. Calculate Cross Product of Two Vectors" << endl << endl;
cout << "Please enter the number of your choice (9 to exit): ";
cin >> choice;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> choice;
    }
    else
    {
        break;
    }
}

system("cls");

switch (choice)
{
    case 1: //Vector Length & Normalization
        cout << "Calculating Vector Length & Normalization" << endl << endl;
        VectorLengthNormal();
        choice = 0; //Had an error where switch would go straight back into exercise
        //after completion, inefficient but works
        break;
    case 2: // Distance Between 2 Points
```

Application of Maths for Software Development

```
        cout << "The Distance Between Two Points" << endl << endl;

        Distance2Point();

        choice = 0;

        break;

    case 3: // Dot Product

        cout << "Calculating the Dot Product of Two Vectors" << endl << endl;

        DotProduct();

        choice = 0;

        break;

    case 4: // Cross Product

        cout << "Calculating the Cross Product of Two Vectors" << endl << endl;

        CrossProduct();

        choice = 0;

        break;

    default:

        break;

    }

}

return 0;

}

void VectorLengthNormal()
{
    int x, y, z; //Coordinates
    int bigX, bigY, bigZ; //Coordinates Squared
    int BIGXYZ; //Coords added together
    double Length; //Vector Length
    int dimechoice = 0; //2D or 3D Vector choice

    cout << "Is your Vector..." << endl << endl;

    cout << "1. 2D (Two Dimensional)" << endl;
```

```
cout << "2. 3D (Three Dimensional)" << endl << endl;
cout << "Please enter the number of your choice (Any other number to exit): ";
cin >> dimechoice;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> dimechoice;
    }
    else
    {
        break;
    }
}
cout << endl;

switch (dimechoice)
{
case 1:
    cout << "Please enter your X coordinate: ";
    cin >> x;
    while (1)
    {
        if (cin.fail()) //Detects character input when number required
        {
            cin.clear(); //Prevents looping display
            cin.ignore(); //Skips to next line
            cout << "Incorrect Input detected, try again" << endl;
```

Application of Maths for Software Development

```
        cin >> x;
    }
    else
    {
        break;
    }
}
cout << "Please enter your Y coordinate: ";
cin >> y;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> y;
    }
    else
    {
        break;
    }
}
cout << endl;

bigX = x * x, bigY = y * y;
BIGXYZ = bigX + bigY;
Length = sqrt(BIGXYZ);

cout << "The Length of Your Vector is: " << setprecision(2) << fixed << Length <<
endl; //Set to display 2 decimal points
```

```
        cout << "And When Normalized is: " << x / Length << ", " << y / Length << endl <<
endl; //Hopefully don't have to apply precision here since it comes out similar here too

        break;

    case 2:

        cout << "Please enter your X coordinate: ";

        cin >> x;

        while (1)
        {

            if (cin.fail()) //Detects character input when number required
            {

                cin.clear(); //Prevents looping display
                cin.ignore(); //Skips to next line
                cout << "Incorrect Input detected, try again" << endl;
                cin >> x;

            }

            else
            {

                break;

            }

        }

        cout << "Please enter your Y coordinate: ";

        cin >> y;

        while (1)
        {

            if (cin.fail()) //Detects character input when number required
            {

                cin.clear(); //Prevents looping display
                cin.ignore(); //Skips to next line
                cout << "Incorrect Input detected, try again" << endl;
                cin >> y;

            }

        }
```

Application of Maths for Software Development

```
        else
        {
            break;
        }
    }
    cout << "Please enter your Z coordinate: ";
    cin >> z;
    while (1)
    {
        if (cin.fail()) //Detects character input when number required
        {
            cin.clear(); //Prevents looping display
            cin.ignore(); //Skips to next line
            cout << "Incorrect Input detected, try again" << endl;
            cin >> z;
        }
        else
        {
            break;
        }
    }
    cout << endl;

    bigX = x * x, bigY = y * y, bigZ = z * z;
    BIGXYZ = bigX + bigY + bigZ;
    Length = sqrt(BIGXYZ);

    cout << "The Length of Your Vector is: " << setprecision(2) << fixed << Length <<
endl;

    cout << "And When Normalized is: " << x / Length << ", " << y / Length << ", " << z /
Length << endl << endl;

    break;
```

```
default:
    cout << "Thanks Anyway" << endl << endl;
}

}

void Distance2Point()
{
    int x1, x2, y1, y2;
    int Xcom, Ycom; //Each respective coordinate Combined
    double SqXcom, SqYcom; //Each coordinate squared
    double Com; //Combined coords
    double Distance;

    cout << "Please enter your first set of coordinates" << endl;
    cout << "Please enter X1:";
    cin >> x1;
    while (1)
    {
        if (cin.fail()) //Detects character input when number required
        {
            cin.clear(); //Prevents looping display
            cin.ignore(); //Skips to next line
            cout << "Incorrect Input detected, try again" << endl;
            cin >> x1;
        }
        else
        {
            break;
        }
    }
}
```

```
cout << "Please enter Y1:";
cin >> y1;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> y1;
    }
    else
    {
        break;
    }
}
cout << endl;
cout << "Please enter your second set of coordinates" << endl;
cout << "Please enter X2:";
cin >> x2;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> x2;
    }
    else
    {

```

```
        break;
    }
}
cout << "Please enter Y2:";
cin >> y2;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> y2;
    }
    else
    {
        break;
    }
}

Xcom = x2 - x1, Ycom = y2 - y1;
SqXcom = Xcom * Xcom, SqYcom = Ycom * Ycom;
Com = SqXcom + SqYcom;
Distance = sqrt(Com);

cout << "The Distance Between the inputted points is: " << setprecision(2) << fixed <<
Distance << endl << endl;
}

void DotProduct()
{
```

```
int x1, y1, z1, x2, y2, z2; //Coordinates
double ab; //Dot product
double VecA, VecB;
int VecCalc = 0;
int vectornumber = 0;
double degrees; //Radian Bad
double theta = 0;

cout << "Do you wish to calculate with..." << endl << endl;
cout << "1. a.b = (x1 * x2) + (y1 * y2)..." << endl;
cout << "2. a.b = |a| * |b| * cos(theta)" << endl;
cout << "Please enter the number of your choice (Any other number to exit): ";
cin >> VecCalc;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> VecCalc;
    }
    else
    {
        break;
    }
}
cout << endl;

switch (VecCalc)
{
```

case 1:

```
cout << "The number of Vectors is?..." << endl << endl;
cout << "1. 2 Vectors" << endl;
cout << "2. 3 Vectors" << endl << endl;
cout << "Please enter the number of your choice (Any other number to exit): ";
cin >> vectornumber;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> vectornumber;
    }
    else
    {
        break;
    }
}
cout << endl;

switch (vectornumber)
{
    case 1:
        cout << "Please enter your first set of coordinates" << endl;
        cout << "Please enter X1:";
        cin >> x1;
        while (1)
        {
            if (cin.fail()) //Detects character input when number required
```

Application of Maths for Software Development

```
{
    cin.clear(); //Prevents looping display
    cin.ignore(); //Skips to next line
    cout << "Incorrect Input detected, try again" << endl;
    cin >> x1;
}
else
{
    break;
}
}
cout << "Please enter Y1:";
cin >> y1;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> y1;
    }
    else
    {
        break;
    }
}
cout << endl;
cout << "Please enter your second set of coordinates" << endl;
cout << "Please enter X2:";
cin >> x2;
```

Application of Maths for Software Development

```
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> x2;
    }
    else
    {
        break;
    }
}
cout << "Please enter Y2:";
cin >> y2;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> y2;
    }
    else
    {
        break;
    }
}
```

Application of Maths for Software Development

```
ab = (x1 * x2) + (y1 * y2);
cout << "The Dot Product of these Two Vectors is: " << setprecision(2) <<
fixed << ab << endl << endl;
break;
case 2:
cout << "Please enter your first set of coordinates" << endl;
cout << "Please enter X1:";
cin >> x1;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> x1;
    }
    else
    {
        break;
    }
}
cout << "Please enter Y1:";
cin >> y1;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
```

Application of Maths for Software Development

```
        cin >> y1;
    }
    else
    {
        break;
    }
}
cout << "Please enter Z1:";
cin >> z1;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> z1;
    }
    else
    {
        break;
    }
}
cout << endl;
cout << "Please enter your second set of coordinates" << endl;
cout << "Please enter X2:";
cin >> x2;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
```

Application of Maths for Software Development

```
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> x2;
    }
    else
    {
        break;
    }
}
cout << "Please enter Y2:";
cin >> y2;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> y2;
    }
    else
    {
        break;
    }
}
cout << "Please enter Z2:";
cin >> z2;
while (1)
{
    if (cin.fail()) //Detects character input when number required
```

Application of Maths for Software Development

```
        {
            cin.clear(); //Prevents looping display
            cin.ignore(); //Skips to next line
            cout << "Incorrect Input detected, try again" << endl;
            cin >> z2;
        }
        else
        {
            break;
        }
    }

    ab = (x1 * x2) + (y1 * y2) + (z1 * z2);

    cout << "The Dot Product of these Two Vectors is: " << setprecision(2) <<
fixed << ab << endl << endl;

    break;

default:
    break;

}

break;

case 2:

    cout << "Please enter Vector A's Length" << endl;
    cin >> VecA;
    while (1)
    {
        if (cin.fail()) //Detects character input when number required
        {
            cin.clear(); //Prevents looping display
            cin.ignore(); //Skips to next line
            cout << "Incorrect Input detected, try again" << endl;
```

Application of Maths for Software Development

```
        cin >> VecA;
    }
    else
    {
        break;
    }
}
cout << "Please enter Vector B's Length" << endl;
cin >> VecB;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> VecB;
    }
    else
    {
        break;
    }
}
cout << "Please enter the Angle" << endl;
cin >> theta;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
```

Application of Maths for Software Development

```
        cout << "Incorrect Input detected, try again" << endl;
        cin >> theta;
    }
    else
    {
        break;
    }
}

degrees = theta * (180 / PI); //Tried converting it into degrees, still gives me the
wrong number

ab = VecA * VecB * cos(degrees); //I don't get it, doesn't work. Is it measuring in
radians or what?

cout << "The Dot Product of these Two Vectors is: " << setprecision(2) << fixed << ab
<< endl << endl;

    break;

    default:

        break;

    }
}

void CrossProduct()
{
    int a, b, c, d, e, f; //constants
    int i, j, k; //unit vectors

    cout << "This program works out the answer using the formula of: " << endl;
    cout << "| i j k |" << endl;
    cout << "| A B C | = a x b" << endl;
    cout << "| D E F |" << endl << endl;
```

```
cout << "Please enter A:";
cin >> a;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> a;
    }
    else
    {
        break;
    }
}
cout << "Please enter B:";
cin >> b;
while (1)
{
    if (cin.fail()) //Detects character input when number required
    {
        cin.clear(); //Prevents looping display
        cin.ignore(); //Skips to next line
        cout << "Incorrect Input detected, try again" << endl;
        cin >> b;
    }
    else
    {
        break;
    }
}
```

```
}  
  
cout << "Please enter C:";  
  
cin >> c;  
  
while (1)  
{  
  
    if (cin.fail()) //Detects character input when number required  
    {  
  
        cin.clear(); //Prevents looping display  
        cin.ignore(); //Skips to next line  
        cout << "Incorrect Input detected, try again" << endl;  
        cin >> c;  
  
    }  
    else  
    {  
  
        break;  
  
    }  
}  
  
cout << "Please enter D:";  
  
cin >> d;  
  
while (1)  
{  
  
    if (cin.fail()) //Detects character input when number required  
    {  
  
        cin.clear(); //Prevents looping display  
        cin.ignore(); //Skips to next line  
        cout << "Incorrect Input detected, try again" << endl;  
        cin >> d;  
  
    }  
    else  
    {  
  
        break;  
  
    }  
}
```

```
    }  
}  
cout << "Please enter E:";  
cin >> e;  
while (1)  
{  
    if (cin.fail()) //Detects character input when number required  
    {  
        cin.clear(); //Prevents looping display  
        cin.ignore(); //Skips to next line  
        cout << "Incorrect Input detected, try again" << endl;  
        cin >> e;  
    }  
    else  
    {  
        break;  
    }  
}  
cout << "Please enter F:";  
cin >> f;  
while (1)  
{  
    if (cin.fail()) //Detects character input when number required  
    {  
        cin.clear(); //Prevents looping display  
        cin.ignore(); //Skips to next line  
        cout << "Incorrect Input detected, try again" << endl;  
        cin >> f;  
    }  
    else  
    {  
        break;  
    }  
}
```

Application of Maths for Software Development

```
        break;
    }
}

i = (b*f - e * c);
j = (a*f - d * c);
k = (a*e - d * b);

cout << "The Cross Product of these Two Vectors is = " << i << "i " << -j << "j " << k << "k" <<
endl; //a minus symbol is in front of j as it is required to be negated
}
```

FINAL GRADE

GENERAL COMMENTS

Instructor

40/100

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14

PAGE 15

PAGE 16

PAGE 17

PAGE 18

PAGE 19

PAGE 20

PAGE 21

PAGE 22

PAGE 23

PAGE 24

PAGE 25

PAGE 26

PAGE 27

PAGE 28

PAGE 29

PAGE 30

PAGE 31

PAGE 32

PAGE 33

PAGE 34

PAGE 35

GRADING FORM: SMU GENERIC_GJ

**KAI BENTLEY -
JENNINGS**

40

RESEARCH-INFORMED LITERATURE



Please use refs. No bibliography or references. Get used to including refs now.

1

KNOWLEDGE AND UNDERSTANDING



No need to use cos for the dot products. The sum of products is fine. Several sections missing.

8

ANALYSIS



Cals correct. Code design appropriate and testing included. Sections missing.

8.5

PRACTICAL APPLICATION AND DEP.



Maths ok. Code working - some sections missing.

20.5

SKILLS FOR PROFESSIONAL PRAC.



Structured report. good. No refs. Sections missing.

2