

INFORMATICS INSTITUTE OF TECHNOLOGY

IN COLLABORATION WITH

UNIVERSITY OF WESTMINSTER (UOW)

BENG (HONS) SOFTWARE ENGINEERING

**MODULE: 5CCGD006C.1 Applied Maths and
Physics for Games**

MODULE LEADER : MR. RANDIL PUSHPANANDA

COURSE WORK – 01

REPORT

SUBMISSION DATE : 18TH JANUARY 2018

USERNAME: w1654191

UOW ID: w16541915/1

STUDENT ID: 2016323

GROUP : B

STUDENT FIRST NAME: KAJENDRAN

STUDENT SURNAME: CHANDRESWARAN

Code and Explanations

Addition Method

```
/* This method is to add two vectors and two 2D vector parameters are passed  
* to this method and it will return the addition of both vectors as a 2D vector object  
*/
```

```
public static Vector2D add2DVector(Vector2D vector1 , Vector2D vector2){  
    Vector2D add2DVector = new Vector2D();  
    add2DVector.setX(vector1.getX() + vector2.getX());  
    add2DVector.setY(vector1.getY() + vector2.getY());  
    return add2DVector;  
}
```

- Every 2D vector objects which are passed as a parameter will have the x and y components, this method will add those with the corresponding components with each other and return the addition as a 2D vector object.

```
Press [1] to 2D vector Additions.  
Press [2] to 2D vector Subtraction.  
Press [3] to 2D vector Dot Product.  
Press [4] to 2D vector Unit Vector.  
Press [5] to 2D vector Magnitude.  
Press [6] to 2D vector Multiplication.  
Press [7] to 2D vector Rotation.  
Press [8] to 2D vector Velocity vector.  
Press [9] to 2D vector Scalar Multiplication
```

```
1
```

```
Enter X1 : 3.3  
Enter Y1 : 6.2  
Enter X2 : 4.2  
Enter Y2 : 12  
The addition of vectors : 7.5,18.2
```

```
Press [0] to Go to main menu.  
Press [e] to Go to End.
```

[Type here]

Subtraction Method

```
Vector2D add2DVector = new Vector2D();
add2DVector.setX(vector1.getX() - vector2.getX());
add2DVector.setY(vector1.getY() - vector2.getY());
return add2DVector;
}
```

Every 2D vector objects which are passed as a parameter will have the x and y components, this method will subtract those with the corresponding components with each other and return the subtract values as a 2D vector object.

```
Press [2] to 2D vector Subtraction.
Press [3] to 2D vector Dot Product.
Press [4] to 2D vector Unit Vector.
Press [5] to 2D vector Magnitude.
Press [6] to 2D vector Multiplication.
Press [7] to 2D vector Rotation.
Press [8] to 2D vector Velocity vector.
Press [9] to 2D vector Scalar Multiplication
2
Enter X1 : 3.3
Enter Y1 : 6.2
Enter X2 : 4.2
Enter Y2 : 12.7
The subtraction of vectors : -0.89999986,-6.5

Press [0] to Go to main menu.
Press [e] to Go to End.
```

[Type here]

Dot Product Method

/* This method is to find the dot product of two vectors and two 2D vector parameters are passed
* to this method and it will return the dot product of both vectors as a float variable.
*/

```
public static float dotProduct(Vector2D vector1 , Vector2D vector2){  
    Vector2D add2DVector = new Vector2D();  
    add2DVector.setX(vector1.getX() * vector2.getX());  
    add2DVector.setY(vector1.getY() * vector2.getY());  
    float result = add2DVector.getX() + add2DVector.getY();  
    return result;  
}
```

- Every 2D vector objects which are passed as a parameter will have the x and y components, this method will multiply those with the corresponding components with each other and return the multiplied values as a 2D vector object.

```
Press [3] to 2D vector Dot Product.  
Press [4] to 2D vector Unit Vector.  
Press [5] to 2D vector Magnitude.  
Press [6] to 2D vector Multiplication.  
Press [7] to 2D vector Rotaion.  
Press [8] to 2D vector Velocity vector.  
Press [9] to 2D vector Scalar Multiplication  
3  
Enter X1 : 7.5  
Enter Y1 : 7.5  
Enter X2 : 12.0  
Enter Y2 : 0.0  
The Dot product of vectors : 90.0  
  
Press [0] to Go to main menu.  
Press [e] to Go to End.
```

[Type here]

Unit Vector Method

/* This method is to find the unit vector of two vectors and two 2D vector parameters are passed
* to this method and it will return the unit vector of both vectors as a float variable.

*/

```
public static double unitVector(Vector2D vector1 , Vector2D vector2){  
    Vector2D add2DVector = new Vector2D();  
    add2DVector.setX(vector1.getX() * vector2.getX());  
    add2DVector.setY(vector1.getY() * vector2.getY());  
    float result = add2DVector.getX() + add2DVector.getY();  
  
    double x = Math.sqrt((vector1.getX() * vector1.getX()) + (vector1.getY() *  
vector1.getY()));  
    double y = Math.sqrt((vector2.getX() * vector2.getX()) + (vector2.getY() *  
vector2.getY()));  
  
    double ans = Math.cos(result / (x*y));  
    return ans;  
}
```

Every 2D vector objects which are passed as a parameter will have the x and y components, this method will multiply those with the corresponding components with each other , the multiplication of both vectors X and Y will be added as a one unit(float result) then double x and y variables will store the sqrt value X^2 and Y^2 after that added to this equation $\text{Math.cos}(\text{result} / (x*y))$ it will return the multiplied values as a 2D vector object.

[Type here]

```

Press [4] to 2D vector Unit Vector.
Press [5] to 2D vector Magnitude.
Press [6] to 2D vector Multiplication.
Press [7] to 2D vector Rotaion.
Press [8] to 2D vector Velocity vector.
Press [9] to 2D vector Scalar Multiplication
4
Enter X1 : 7.5
Enter Y1 : 7.5
Enter X2 : 12.0
Enter Y2 : 0.0
The Unit Vector of vectors : 0.7602445970756302

Press [0] to Go to main menu.
Press [e] to Go to End.

```

Magnitude Method

```

/* This method is to find the magnitute two vectors and two 2D vector parameters
are * * to this method and it will return the magnitude of both vectors as a
float variable.
*/

```

```

public static double findMagnitute(Vector2D vector1){
    double result = Math.sqrt((vector1.getX() * vector1.getX() +
(vector1.getY() * vector1.getY()));
    return result;
}

```

- Every 2D vector objects which are passed as a parameter will have the x and y components first we need to find the addition of both square of X and Y.it will return as a double variable.

[Type here]

```
Press [5] to 2D vector Magnitude.  
Press [6] to 2D vector Multiplication.  
Press [7] to 2D vector Rotaion.  
Press [8] to 2D vector Velocity vector.  
Press [9] to 2D vector Scalar Multiplication  
5  
Enter X1 : 7.5  
Enter Y1 : 7.5  
The Magnitude of vectors : 10.606601717798213  
  
Press [0] to Go to main menu.  
Press [e] to Go to End.
```

Multiplication Method

```
/* This method is to find the multiple of two vectors and two 2D vector parameters  
are  
* to this method and it will return the multiple of both vectors as a 2D vector  
object.  
*/
```

```
public static Vector2D vectorMultiple(Vector2D vector1, Vector2D vector2){  
    Vector2D multiple2DVector = new Vector2D();  
    multiple2DVector.setX(vector1.getX() * vector2.getX());  
    multiple2DVector.setY(vector1.getY() * vector2.getY());  
    return multiple2DVector;  
}
```

- Every 2D vector objects which are passed as a parameter will have the x and y components, this method will multiply those with the corresponding components with each other and return the multiplication as a 2D vector object.

[Type here]

```
Press [8] to 2D vector Velocity vector.  
Press [9] to 2D vector Scalar Multiplication  
8  
Enter a speed : 20  
Enter a angle : 45  
The Velocity vector is : 14.142136,14.142136  
Press [0] to Go to main menu.  
Press [e] to Go to End.
```

Rotation Method

/ This method is to find the rotation of 2D vector and two 2D vector parameter and angle of the vector are passed to this method and it will return the vector rotation of both vectors as a 2D vector object.*

**/*

```
public static Vector2D vectorRotation(Vector2D vector1 , double angle){  
    Vector2D rotation2DVector = new Vector2D();  
    rotation2DVector.setX((float)((vector1.getX() * Math.cos(Math.toRadians(angle)))  
- vector1.getY() * Math.sin(Math.toRadians(angle))));  
    rotation2DVector.setY((float)((vector1.getX() * Math.cos(Math.toRadians(angle)))  
+ vector1.getY() * Math.sin(Math.toRadians(angle))));  
    return rotation2DVector;  
}
```

- Every 2D vector objects which are passed as a parameter will have the x and y components, first finding the cos value of angle and multiply by X component then find the sin value of angle and multiply by Y component then subtract both. after that again do the same and add this time, Both these values will refer as X , Y components.

[Type here]


```

Press [7] to 2D vector Rotaion.
Press [8] to 2D vector Velocity vector.
Press [9] to 2D vector Scalar Multiplication
7
Enter X1 : 3
Enter Y1 : 4
Enter a angle : 60
The vector rotation is : -1.9641017,4.964102
Press [0] to Go to main menu.
Press [e] to Go to End.

```

Rotation Method

/* This method is to find the Velocity of the vector and speed and angle of the vector are passed

* to this method and it will return the velocity vetor of both vectors as a 2D vector object.
*/

```

public static Vector2D findVelocityVector(double speed , double angle){
    Vector2D VelocityVector = new Vector2D();
    VelocityVector.setX((float)(speed * Math.cos(Math.toRadians(angle))));
    VelocityVector.setY((float)(speed * Math.sin(Math.toRadians(angle))));
    return VelocityVector;
}

```

- Every 2D vector objects which are passed as a parameter will have the x and y components, To the X component multiply the speed by the cos value of angle and , to the Y component multiply the speed by the sin value of angle. It will return as a 2D vector object.

[Type here]

ScalarMultiplication Method

```
/* This method is to find multiplication of vector and two 2D vector parameter and  
scalar are passed  
* to this method and it will return the multiplication of vector as a 2D vector  
object.  
*/
```

```
public static Vector2D scalarMultiple(Vector2D vector1, double scalar){  
    Vector2D ScalarMultipleVector = new Vector2D();  
    ScalarMultipleVector.setX((float)(vector1.getX() * scalar));  
    ScalarMultipleVector.setY((float)(vector1.getY() * scalar));  
    return ScalarMultipleVector;  
}
```

- Every 2D vector objects which are passed as a parameter will have the x and y components, these components will multiply by a scalar and return as a 2D vector object.

[Type here]

Press [9] to 2D vector Scalar Multiplication

9

Enter X1 : 5

Enter Y1 : 3

Enter a Scalar : 9

The Multiplication of vector is : 45.0,27.0

Press [0] to Go to main menu.

Press [e] to Go to End.

Addition Method

/* This method is to find the addition of two 3D vectors and two 3D vector parameters are passed

* to this method and it will return the addition of both vectors as a 3D vector object

*/

```
public static Vector3D add3DVector(Vector3D vector1, Vector3D vector2){  
    Vector3D add3DVector = new Vector3D();  
    add3DVector.setX(vector1.getX() + vector2.getX());  
    add3DVector.setY(vector1.getY() + vector2.getY());  
    add3DVector.setZ(vector1.getZ() + vector2.getZ());  
    return add3DVector;  
}
```

[Type here]

- Every 3D vector objects which are passed as a parameter will have the X , Y and Z components, this method will add those with the corresponding components with each other and return the addition as a 3D vector object.

```

Press [1] to 3D vector Additions.
Press [2] to 3D vector Subtraction.
Press [3] to 3D vector Dot Product.
Press [4] to 3D vector Unit Vector.
Press [5] to 3D vector Magnitude.
Press [6] to 3D vector Multiplication.
Press [7] to 3D vector Scalar Multiplication
1
Enter X1 : 1
Enter Y1 : 4.5
Enter Z1 : 8.3
Enter X2 : 12.6
Enter Y2 : -4.5
Enter Z2 : 6.7
The addition of vectors : 13.6,0.0,15.0

Press [0] to Go to main menu.
Press [e] to Go to End.

```

Subtraction Method

```

/* This method is to find the subtraction of two 3D vectors and two 3D vector parameters are
passed
* to this method and it will return the subtraction of both vectors as a 3D vector object
*/

```

```

public static Vector3D sub3DVector(Vector3D vector1, Vector3D vector2){
    Vector3D sub3DVector = new Vector3D();
    sub3DVector.setX(vector1.getX() - vector2.getX());
    sub3DVector.setY(vector1.getY() - vector2.getY());
    sub3DVector.setZ(vector1.getZ() - vector2.getZ());
    return sub3DVector;
}

```

[Type here]

- Every 3D vector objects which are passed as a parameter will have the X , Y and Z components, this method will subtract those with the corresponding components with each other and return the subtraction as a 3D vector object.

```

Press [2] to 3D vector Subtraction.
Press [3] to 3D vector Dot Product.
Press [4] to 3D vector Unit Vector.
Press [5] to 3D vector Magnitude.
Press [6] to 3D vector Multiplication.
Press [7] to 3D vector Scalar Multiplication
2
Enter X1 : 1
Enter Y1 : 4.5
Enter Z1 : 8.3
Enter X2 : 12.6
Enter Y2 : -4.5
Enter Z2 : 6.7
The subtraction of vectors : -11.6,9.0,1.6000004

Press [0] to Go to main menu.
Press [e] to Go to End.

```

Dot Product Method

/* This method is to find the dot product two vectors and two 3D vector parameters are passed
 * to this method and it will return the dot product of both vectors as a 3D vector object
 */

```

public static float dotProduct3DVector(Vector3D vector1, Vector3D vector2){
    Vector3D dotProduct = new Vector3D();
    dotProduct.setX(vector1.getX() * vector2.getX());
    dotProduct.setY(vector1.getY() * vector2.getY());
    dotProduct.setZ(vector1.getZ() * vector2.getZ());
}

```

[Type here]

```

        float result = dotProduct.getX() + dotProduct.getY() + dotProduct.getZ();
        return result;
    }

```

- Every 3D vector objects which are passed as a parameter will have the x , y and z components, this method will multiply those with the corresponding components with each other and return the multiplied values as a 3D vector object.

```

Press [3] to 3D vector Dot Product.
Press [4] to 3D vector Unit Vector.
Press [5] to 3D vector Magnitude.
Press [6] to 3D vector Multiplication.
Press [7] to 3D vector Scalar Multiplication
3
Enter X1 : 1
Enter Y1 : 4.5
Enter Z1 : 8.3
Enter X2 : 12.6
Enter Y2 : 0
Enter Z2 : 6.3
The Dot product of vectors : 64.89

Press [0] to Go to main menu.
Press [e] to Go to End.

```

Unit Vector Method

```

/* This method is to find unit vector of two vectors and two 3D vector parameters are passed
 * to this method and it will return the unit vector of both vectors as a float variable.
 */

```

```

public static double findUnitVector(Vector3D vector1, Vector3D vector2){
    Vector3D unitVector = new Vector3D();
    unitVector.setX(vector1.getX() * vector2.getX());
    unitVector.setY(vector1.getY() * vector2.getY());
    unitVector.setZ(vector1.getZ() * vector2.getZ());
    float result = unitVector.getX() + unitVector.getY() + unitVector.getZ();
}

```

[Type here]

```

        double x = Math.sqrt((vector1.getX() * vector1.getX()) + (vector1.getY() * vector1.getY())
+ (vector1.getZ() * vector1.getZ()));
        double y = Math.sqrt((vector2.getX() * vector2.getX()) + (vector2.getY() * vector2.getY())
+ (vector2.getZ() * vector2.getZ()));

        double ans = Math.cos(result / (x*y));
        return ans;
    }

```

- Every 3D vector objects which are passed as a parameter will have the x , y and z components, this method will multiply those with the corresponding components with each other , the multiplication of both vectors X and Y will be added as a one unit(float result) then double x , y and z variables will store the sqrt value X^2 and Y^2 after that added to this equation $\text{Math.cos}(\text{result} / (x*y))$ it will return the multiplied values as a 3D vector object.

```

Press [4] to 3D vector Unit Vector.
Press [5] to 3D vector Magnitude.
Press [6] to 3D vector Multiplication.
Press [7] to 3D vector Scalar Multiplication
4
Enter X1 : 1.0
Enter Y1 : 4.5
Enter Z1 : 8.3
Enter X2 : 12.6
Enter Y2 : 0
Enter Z2 : 6.3
The unitVector of vectors : 0.8845959461732227

Press [0] to Go to main menu.
Press [e] to Go to End.

```

Magnitude Method

/* This method is to find the magnitude of two vectors and 3D vector parameter is passed
 * to this method and it will return the magnitude value of vector as a float variable.
 */

```

public static double findMagnitude(Vector3D vector1){
    double result = Math.sqrt((vector1.getX() * vector1.getX()) +
(vector1.getY() * vector1.getY()) + (vector1.getZ() * vector1.getZ()));
    return result;
}

```

[Type here]

}

- Every 3D vector objects which are passed as a parameter will have the x , y and z components first we need to find the addition of both square of X , Y and z. it will return as a double variable.

```
Press [4] to 3D vector Unit Vector.
Press [5] to 3D vector Magnitude.
Press [6] to 3D vector Multiplication.
Press [7] to 3D vector Scalar Multiplication
4
Enter X1 : 1.0
Enter Y1 : 4.5
Enter Z1 : 8.3
Enter X2 : 12.6
Enter Y2 : 0
Enter Z2 : 6.3
The unitVector of vectors : 0.8845959461732227

Press [0] to Go to main menu.
Press [e] to Go to End.
```

Multiplication Method

```
/* This method is to find the multiplication of two vectors and two 2D
vector parameters are passed
* to this method and it will return the multiplication of both vectors as
a 3D vector object.
*/
```

[Type here]


```

    public static Vector3D multipleVector(Vector3D vector1,Vector3D
vector2){
        Vector3D multipleVector = new Vector3D();
        multipleVector.setX(vector1.getX() * vector2.getX());
        multipleVector.setY(vector1.getY() * vector2.getY());
        multipleVector.setZ(vector1.getZ() * vector2.getZ());
        return multipleVector;
    }

```

- Every 3D vector objects which are passed as a parameter will have the x , y and z components, this method will multiply those with the corresponding components with each other and return the multiplication as a 3D vector object.

Scalar Multiplication Method

```

    /* This method is to find the multiplication of vectors by scalar and 2D vector parameter ,
    scalar are passed
    * to this method and it will return the multiplication of vectors as a 3D vector object.
    */
    public static Vector3D multipleScalarVector(Vector3D vector1,double scalar){

```

[Type here]

```

        Vector3D multipleScalarVector = new Vector3D();
        multipleScalarVector.setX((float)(vector1.getX() * scalar));
        multipleScalarVector.setY((float)(vector1.getY() * scalar));
        multipleScalarVector.setZ((float)(vector1.getZ() * scalar));
        return multipleScalarVector;
    }

```

- Every 3D vector objects which are passed as a parameter will have the x , y and z components,these components will multiply by a scalar and return as a 3D vector object.

```

Press [7] to 3D vector Scalar Multiplication
7
Enter X1 : 6
Enter Y1 : 3
Enter Z1 : 7
Enter Scalar : 4
The unitVector of vectors : 24.0,12.0,28.0

Press [0] to Go to main menu.
Press [e] to Go to End.

```

QuestionA Method

```
public static void questionA(double velocity,double angle){
```

[Type here]

```
ArrayList<Float> vector2Dlist = new ArrayList<Float>(); // creating the arraylists
to store the position and velocity vector components.
```

```
ArrayList<Float> ComponentXList = new ArrayList<Float>();
```

```
ArrayList<Float> ComponentYList = new ArrayList<Float>();
```

```
double time = (velocity * Math.sin(Math.toRadians(angle)) * 2)/9.82; //  $t = v * (\sin(q) * 2)/9.82$ 
```

```
double Xcomponent = velocity * Math.cos(Math.toRadians(angle)); //  $X = v * \cos(Q)$ 
calculating the velocity vector
```

```
double Ycomponent = velocity * Math.sin(Math.toRadians(angle)); //  $Y = v * \sin(Q)$ 
```

```
double positionX = 0; // keep tracking of x
```

```
double positionY = 0; // keep tracking of y
```

```
vector2Dlist.add((float) Xcomponent); // adding the calculated x and y velocity vectors to
the arraylist.
```

```
vector2Dlist.add((float) Ycomponent);
```

```
ComponentXList.add((float) positionX); // adding the calculated x and y position vectors to
the arraylist.
```

```
ComponentYList.add((float) positionY);
```

```
System.out.println("Velocity Vector: " + vector2Dlist.get(0) + "," + vector2Dlist.get(1));
```

```
System.out.println("Position Vector: " + ComponentXList.get(0) + "," +
ComponentYList.get(0) + "\n");
```

```
double x = 0;
```

```
while (x <= (time - 0.01)){
```

```
double velociyY = vector2Dlist.get(1) - 9.82 * 0.02; // velocity vector =  $x - (9.82 * t)$ 
```

```
double posY = ComponentYList.get(0) + ((vector2Dlist.get(1)+ velociyY)/2) * 0.02; //
position vector =  $x * ((y + velocity\_vector.y)/2) * t$ 
```

```
vector2Dlist.remove(1);
```

```
vector2Dlist.add((float) velociyY); // adding the velocity vector y to the list.
```

```
double x1 = ComponentXList.get(0) + ((Xcomponent + Xcomponent)/2) * 0.02; //  $x = x + ((x+x)/2) * t$ 
```

```
ComponentYList.remove(0); // removing the old position vector
```

```
ComponentYList.add((float) posY); // adding the new position vector after the move
```

[Type here]

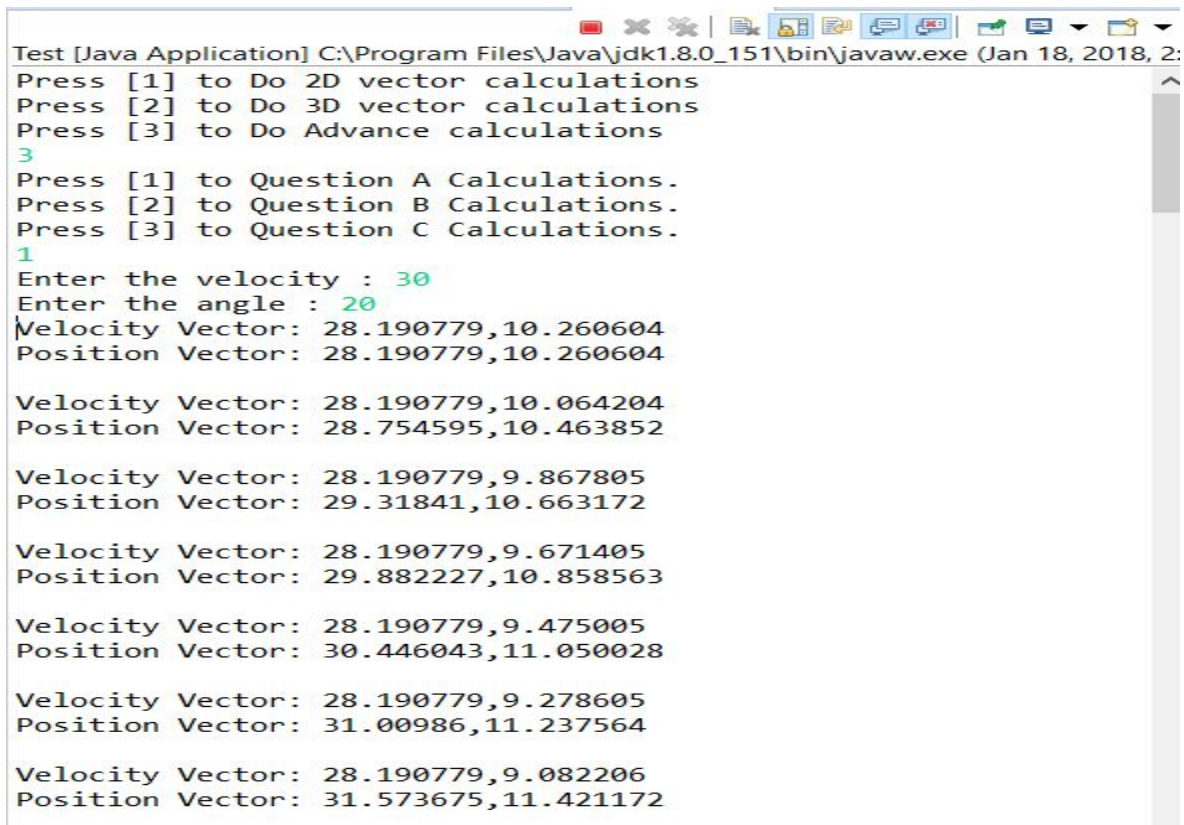
```

ComponentXList.remove(0);
ComponentXList.add((float) x1);

System.out.println("Velocity Vector: " + vector2Dlist.get(0) + "," + vector2Dlist.get(1));
System.out.println("Position Vector: " + ComponentXList.get(0) + "," +
ComponentYList.get(0) + "\n");

    x += 0.02;
}
}

```



```

Test [Java Application] C:\Program Files\Java\jdk1.8.0_151\bin\javaw.exe (Jan 18, 2018, 2:
Press [1] to Do 2D vector calculations
Press [2] to Do 3D vector calculations
Press [3] to Do Advance calculations
3
Press [1] to Question A Calculations.
Press [2] to Question B Calculations.
Press [3] to Question C Calculations.
1
Enter the velocity : 30
Enter the angle : 20
Velocity Vector: 28.190779,10.260604
Position Vector: 28.190779,10.260604

Velocity Vector: 28.190779,10.064204
Position Vector: 28.754595,10.463852

Velocity Vector: 28.190779,9.867805
Position Vector: 29.31841,10.663172

Velocity Vector: 28.190779,9.671405
Position Vector: 29.882227,10.858563

Velocity Vector: 28.190779,9.475005
Position Vector: 30.446043,11.050028

Velocity Vector: 28.190779,9.278605
Position Vector: 31.00986,11.237564

Velocity Vector: 28.190779,9.082206
Position Vector: 31.573675,11.421172

```

QuestionB Method

```

public static void questionB(double time,Vector3D vector3D , double mass){
    ArrayList <Float> vector3Dlist = new ArrayList<Float>();

```

[Type here]

```

        ArrayList <Float> componentXList = new ArrayList<Float>(); // stores the values
of X
        ArrayList <Float> componentYList = new ArrayList<Float>(); // stores the values
of Y
        ArrayList <Float> componentZList = new ArrayList<Float>(); // stores the values
of Z
        double posX = 0;
        double posY = 0;
        double posZ = 0;
        double tyme = (((time * vector3D.getY())/mass)*2)/9.82;
        double veloX = ((time * vector3D.getX())/mass);
        double veloY = ((time * vector3D.getY())/mass);
        double veloZ = ((time * vector3D.getZ())/mass);
        componentXList.add((float) posX);//save in xPosition ArrayList
        componentYList.add((float) posY);
        componentZList.add((float) posZ);
        vector3Dlist.add((float) veloX);//save in vector ArrayList
        vector3Dlist.add((float) veloZ);
        vector3Dlist.add((float) veloY);
        System.out.println("Position Vector: " + componentXList.get(0) + "," +
componentYList.get(0) + "," + componentZList.get(0));
        System.out.println("Velocity Vector: " + vector3Dlist.get(0) + "," + vector3Dlist.get(2) + "," +
vector3Dlist.get(1) + "\n");

        double i = 0;
        while (i <= (tyme - 0.01)){
            double vY = vector3Dlist.get(2) + (-9.82) * 0.02;
            double py = componentYList.get(0) + ((vector3Dlist.get(2) + vY)/2)*0.02; // calculating
the new postion vector of Y
            vector3Dlist.remove(2);
            vector3Dlist.add((float) vY);
            componentXList.remove(0); // removing the old position of X
            componentXList.add((float) pX); // adding the new position of Y
            i += 0.02;

            System.out.println("Velocity Vector: " + vector3Dlist.get(0) + "," + vector3Dlist.get(2) + ","
+ vector3Dlist.get(1) + "\n");
            System.out.println("Position Vector: " + componentXList.get(0) + "," +
componentYList.get(0) + "," + componentZList.get(0));
        }

```

[Type here]

questionC Method

```
public static void questionC(Vector2D vector2D,double distance,double mass,double
force,double angle,double radius){
    ArrayList <Float> vector2Dlist = new ArrayList<Float>(); // arraylists to store the
value of x,y before and after the move.
    ArrayList <Float> x1Component = new ArrayList<Float>();
    ArrayList <Float> x2Component = new ArrayList<Float>();
    ArrayList <Float> y1Component = new ArrayList<Float>();
    ArrayList <Float> y2Component = new ArrayList<Float>();
    ArrayList <Float> vector2DXComponent = new ArrayList<Float>();
    ArrayList <Float> vector2DYComponent = new ArrayList<Float>();
    double a = force/mass; // f = ma calculating the accelaration.

    double v = Math.sqrt((Math.pow(vector2D.getY(),2)) - (2 * a * distance));

    double Vb = v * Math.sin(Math.toRadians(angle)); // velocity vector = v * sin(Q)
    double tyme = ( vector2D.getY() - v)/a; // calculate the time
    double t2 = Vb * Math.sin(Math.toRadians(angle))/a;

    double position1X = Vb; // POSITION VECTOR BEFORE THE MOVE
    double position1Y = t2;

    vector2Dlist.add(vector2D.getX()); // passing the x and y component to the vector2Dlist
    ArrayList.
    vector2Dlist.add( vector2D.getY());

    x1Component.add((float) position1X); // passing the postion vector before the move.

    [Type here]
```

```

y1Component.add((float) position1Y);

System.out.println("Motion before the collision of Cue ball" + "\n");
System.out.println("Position Vector: " + x1Component.get(0) + "," + y1Component.get(0));
System.out.println("Velocity Vector: " + vector2Dlist.get(0) + "," + vector2Dlist.get(1) + "\n");

double i = 0;
while (i <= (tyme - 0.01)){

    vector2Dlist.remove(1); // updating the position and velocity vector in arraylist
    vector2Dlist.add((float) Vay);

    y1Component.remove(0);
    y1Component.add((float) pY);

    System.out.println("Position Vector: " + x1Component.get(0) + "," +
y1Component.get(0));
    System.out.println("Velocity Vector: " + vector2Dlist.get(0) + "," + vector2Dlist.get(1) +
"\n");

    i += 0.02;
}
vector2DXComponent.add((float)(Vb * Math.cos(Math.toRadians(angle)))); //calculating the
velocity vector

//System.out.println(t2+" "+Vb);
System.out.println("Motion of the Object ball after the collision" + "\n");
System.out.println("Position Vector: " + x2Component.get(0) + "," + y2Component.get(0));
System.out.println("Velocity Vector: " + vector2DXComponent.get(0) + "," +
vector2DYComponent.get(0) + "\n");

while (i <= (t2 - 0.01)){

    vector2DXComponent.remove(0); // removing the starting velocity of vector
    vector2DYComponent.remove(0);

    vector2DXComponent.add((float)Vbx); // adding the final velocity of the vector.
    vector2DYComponent.add((float)Vby);

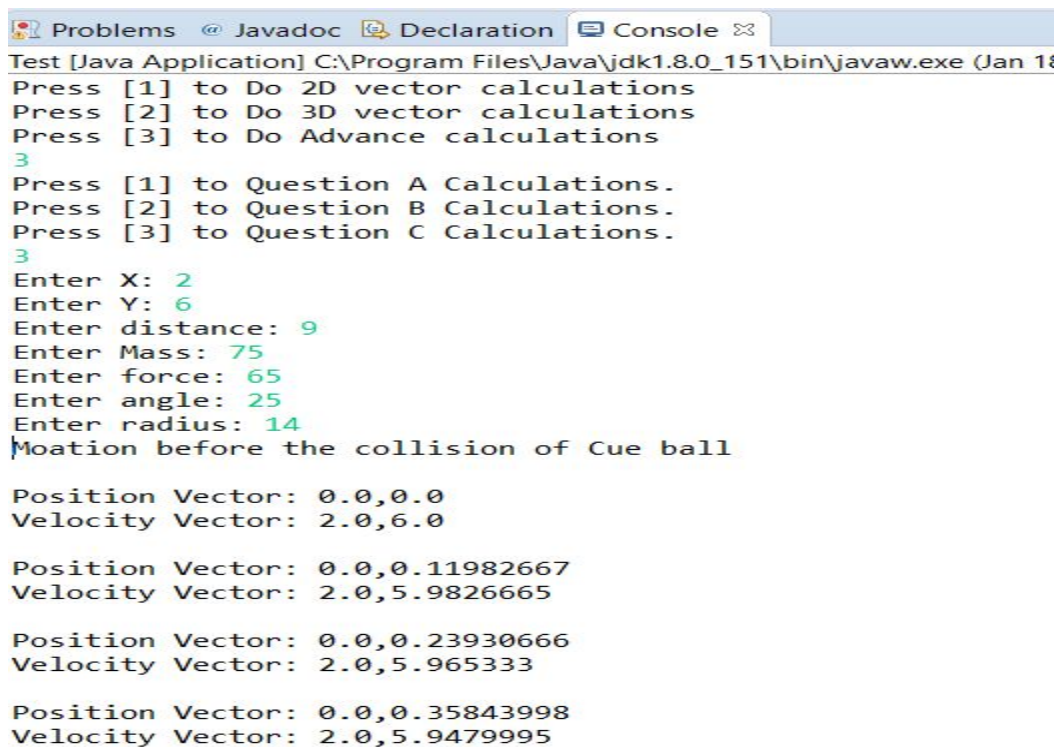
    x2Component.remove(0); // removing the starting position of vector
    y2Component.remove(0);

```

[Type here]

```
x2Component.add((float)pX2); // adding the final velocity of vector
y2Component.add((float)pY2);
```

```
System.out.println("Position Vector: " + x2Component.get(0) + "," +
y2Component.get(0));
System.out.println("Velocity Vector: " + vector2DXComponent.get(0) + "," +
vector2DYComponent.get(0) + "\n");
i += 0.02;
}
}
```



```
Problems @ Javadoc Declaration Console
Test [Java Application] C:\Program Files\Java\jdk1.8.0_151\bin\javaw.exe (Jan 18
Press [1] to Do 2D vector calculations
Press [2] to Do 3D vector calculations
Press [3] to Do Advance calculations
3
Press [1] to Question A Calculations.
Press [2] to Question B Calculations.
Press [3] to Question C Calculations.
3
Enter X: 2
Enter Y: 6
Enter distance: 9
Enter Mass: 75
Enter force: 65
Enter angle: 25
Enter radius: 14
Motion before the collision of Cue ball

Position Vector: 0.0,0.0
Velocity Vector: 2.0,6.0

Position Vector: 0.0,0.11982667
Velocity Vector: 2.0,5.9826665

Position Vector: 0.0,0.23930666
Velocity Vector: 2.0,5.965333

Position Vector: 0.0,0.35843998
Velocity Vector: 2.0,5.9479995
```

[Type here]