Group 53
2020A7PS0971P - Deepam Desai
2020A7PS0123P - Aryan Desai

Drive Link for project video:-
https://drive.google.com/file/d/1UUtiweqDcSQmWx9gPzKWsGZc_PuLjhHz/view?usp=sharing

Spring, Friction and Forces

Brief Description of Project
To realize a time sliced simulation output of various objects present in the system of classical mechanics with respect to time. The user selects various properties like the type of object to simulate, friction coefficient, spring constant, angle of inclination of plane, initial velocities and orientations according to which a simulation is produced which depicts the position of the selected object in the plane with respect to time.

Input source - System input using the command line.
Output source - A table with data of time stamps, final location in x axis and final location in y axis stored in a **.csv file.** The **output is also displayed in the command line.**

Input and Output Methodology
Variables common among all the objects are taken as input variables firstly. For example, mass of object, time Slice, Number Of Snapshots, Initial Location X and Initial Location Y are taken as input.
Afterwards,
The input of type of object is taken as an object ID, where:
Object ID 1 - Box
Object ID 2 - Ball
Object ID 3 - Cylinder
Object ID 4 - Ring

If the **object ID selected is 1**, that is the object selected is Box; we now have 2 cases:-

1) If the user wants a spring to be attached with the box, we take the assumption that the object is on ground with no initial velocity in either x-axis or y-axis. The object at t=0 is on the specified position as given in input with oscillations only in the x-axis and about its mean position at the starting.
   The amplitude and spring constant are taken as an input and the object moves according to simple harmonic motion.

2) If the user doesn't watch a spring, then we have 2-D motion of the box. We now take the friction coefficient and initial velocity in x-axis and y-axis as input. We further have 2 cases in the motion of box:-

    A. If the initial velocity in y-axis is zero, we further have 2 sub-cases:-
        a) If the angle of inclination is zero, the object moves on the ground, executing basic linear motion with sliding friction acting on it.
        b) If the angle of inclination is not-zero, then we need to simulate the box moving on an infinite plane undergoing sliding friction.

    B. If the initial velocity in y-axis is non - zero, we have 2 sub-parts of this condition:-
        a) If the initial location of an object in the y-axis is zero, that object will firstly complete a projectile motion and after reaching the ground again, will go on to move on the plane surface with sliding friction. During the projectile motion, no frictional force would be applied as the object is not in contact with the ground only.
        The assumption taken is that after projectile motion, the object will fall on the horizontal plane only, not on an inclined plane.

        b) If the initial location of an object in the y-axis is non - zero, than the object first will reach the ground by doing a projectile motion from its initial position in the y-axis until it reaches the ground. After reaching the ground, the box will again move on the horizontal plane with resistive force as the sliding friction.
        The assumption taken again is that after projectile motion, the object will fall on the horizontal plane only, not on an inclined plane.

For the **object IDs 2,3 and 4**; rolling objects are taken. The objects are assumed to be on an infinitely long **inclined plane** with angle of **theta** (assumed between 0 to 90 degrees) with the x-axis. The motion of the objects is stopped either by friction or time. Velocity along the inclined plane and the radius of the object is taken as an additional input. The velocities along the x and y axis are initialised according to the velocity down the inclined plane.

If the **object ID selected is 2**, that means the object selected is Ball; we now calculate the radius of gyration of the ball using radius. Then a new object of the ball class is created and all the necessary variables are passed as arguments in the constructor. The roll function is executed which returns the final location of the object as a 2D array. Each row of the 2D array denotes each snapshot of the simulation and the 2 columns denote the final location in the x and y axis.

The 2D array is passed in the simulate function as an argument and the simulate function stores all the data in a .csv file.

The **process remains the same for object ID 3 and 4**, which stands for cylinder and ring respectively.

Test cases :-

For Object ID 1 (Box):-

1) **For implementing spring and box** (k != 0) :-

   Input:

   Give the mass of the object in kg:
   10
   Give the time slice required for output and the number of Snapshots to be taken:
   1
   10
   Give the initial location of the Object in X and Y coordinates respectively:
   5
   0
   Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
   1
   Give spring constant(input '0' if no spring is present):
   5
   Enter value of amplitude:
   10

   Output:

   The location of object at t = 0 : 5.0 0.0
   The location of object at t = 1 : 8.20701742283476 0.0
   The location of object at t = 2 : -1.0752470152250018 0.0
   The location of object at t = 3 : 13.3016903357205 0.0
   The location of object at t = 4 : -4.651146946435382 0.0
   The location of object at t = 5 : 14.981060878023216 0.0
   The location of object at t = 6 : -4.256580176041773 0.0
   The location of object at t = 7 : 12.554238645226754 0.0
   The location of object at t = 8 : -0.05387085392004831 0.0
   The location of object at t = 9 : 7.019614493828174 0.0

2) For implementing Box moving on a horizontal surface (InitialVelocityY = 0 and AngleOfInclination = 0) :-

Input:

Give the mass of the object in kg:
10
Give the timeslice required for output and the number of Snapshots to be taken:
1
5
Give the initial location of the Object in X and Y coordinates respectively:
5
0
Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
1
Give spring constant(input '0' if no spring is present):
0
Give the initial velocity of the Object in X and Y coordinates respectively:
10
0
Give coefficient of Friction:
0.5
Give the angle of inclination of ground:
0

Output:-

Location of object at time 0 is 5.0 0.0
Location of object at time 1 is 12.55 0.0
Location of object at time 2 is 15.2 0.0
Location of object at time 3 is 15.20408163265306 0.0
Location of object at time 4 is 15.20408163265306 0.0

3) **For implementing Box moving on an inclined surface** (InitialVelocityY = 0 and AngleOfInclination != 0) :-

Input:-

Give the mass of the object in kg:
10
Give the timeslice required for output and the number of Snapshots to be taken:
1
5
Give the initial location of the Object in X and Y coordinates respectively:
5
5
Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
1

Give spring constant(input '0' if no spring is present):
0
Give the initial velocity of the Object in X and Y coordinates respectively:
10
0
Give coefficient of Friction:
0.5
Give the angle of inclination of ground:
45

Output:-
Location of object at time 0 is 5.0 5.0
Location of object at time 1 is 13.267588386092958 1.8324116139070412
Location of object at time 2 is 18.070353544371834 -7.670353544371835
Location of object at time 3 is 19.430750636460154 -21.38557589415209
Location of object at time 4 is 19.430750636460154 -21.38557589415209

4) **For implementing a box undergoing a projectile motion and later on sliding friction** (InitialVelocityY != 0 and InitialLocatonY = 0):-
Input:-
Give the mass of the object in kg:
10
Give the time slice required for output and the number of Snapshots to be taken:
1
7
Give the initial location of the Object in X and Y coordinates respectively:
5
0
Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
1
Give spring constant(input '0' if no spring is present):
0
Give the initial velocity of the Object in X and Y coordinates respectively:
5
5
Give coefficient of Friction:
0.5

Output:-
Location of object at time 0 is 5.0 0.0
Location of object at time 1 is 10.0 0.09999999999999964
Location of object at time 2 is 12.653061224489795 0.0
Location of object at time 3 is 17.653061224489797 0.0

Location of object at time 4 is 17.653061224489797 0.0
Location of object at time 5 is 17.653061224489797 0.0
Location of object at time 6 is 17.653061224489797 0.0

5) **For implementing a box undergoing a projectile motion and later on sliding friction** (InitialVelocityY != 0 and InitialLocatonY != 0):-

Input:-

Give the mass of the object in kg:
10
Give the time slice required for output and the number of Snapshots to be taken:
1
7
Give the initial location of the Object in X and Y coordinates respectively:
5
5
Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
1
Give spring constant(input '0' if no spring is present):
0
Give the initial velocity of the Object in X and Y coordinates respectively:
5
5
Give coefficient of Friction:
0.5

Output:-

Location of object at time 0 is 5.0 5.0
Location of object at time 1 is 10.0 5.1
Location of object at time 2 is 15.760477809392556 0.0
Location of object at time 3 is 16.706769516941165 0.0
Location of object at time 4 is 16.706769516941165 0.0
Location of object at time 5 is 16.706769516941165 0.0
Location of object at time 6 is 16.706769516941165 0.0

For Object ID 2 (Ball) :-

6) **For implementing Ball rolling on a inclined plane**

Input:-

Give the mass of the object in kg:
5
Give the timeslice required for output and the number of Snapshots to be taken:

1
5
Give the initial location of the Object in X and Y coordinates respectively:
25
25
Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
2
Give the radius of the object in metre:
1
Give angle of inclination to the ground:
15
Give the velocity along the direction of incline:
12

Output:-
Location of object at time 0 is 25.0 25.0
Location of object at time 1 is 36.59110991546882 21.89417145876975
Location of object at time 2 is 48.18221983093764 18.788342917539502
Location of object at time 3 is 59.77332974640646 15.682514376309253
Location of object at time 4 is 71.36443966187528 12.576685835079005

For Object ID 3 (Cylinder) :-

7) **For implementation of Cylinder rolling on a inclined plane**

Input:-
Give the mass of the object in kg:
6
Give the timeslice required for output and the number of Snapshots to be taken:
1
5
Give the initial location of the Object in X and Y coordinates respectively:
10
15
Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
3
Give the radius of the object in metre:
1.5
Give angle of inclination to the ground:
30
Give the velocity along the direction of incline:
10

Output:-

Location of object at time 0 is 10.0 15.0
Location of object at time 1 is 18.66025403784439 10.0
Location of object at time 2 is 27.320508075688775 5.000000000000002
Location of object at time 3 is 35.98076211353316 1.7763568394002505E-15
Location of object at time 4 is 44.64101615137755 -4.9999999999999964

For object ID 4 (Ring) :-

8) **For implementation of Ring rolling on a inclined plane**

Input:-

Give the mass of the object in kg:
7
Give the timeslice required for output and the number of Snapshots to be taken:
1
5
Give the initial location of the Object in X and Y coordinates respectively:
15
10
Give the object ID: (Press: 1 for BOX, 2 for BALL, 3 for CYLINDER, 4 for RING)
4
Give the radius of the object in metre:
2.3
Give angle of inclination to the ground:
35
Give the velocity along the direction of incline:
5

Output:-

Location of object at time 0 is 15.0 10.0
Location of object at time 1 is 19.09576022144496 7.13211781824477
Location of object at time 2 is 23.191520442889917 4.2642356364895395
Location of object at time 3 is 27.28728066433488 1.3963534547343102
Location of object at time 4 is 31.383040885779835 -1.471528727020921

The **6 principles of OOP** and their usage in our project:-

1) **Encapsulation** : It is defined as wrapping of data under a single unit so as to prevent its access from other classes. It can be achieved by declaring all the variables in a class as private and the getter and setter methods as public.
   In our project, the variables spring Constant(k) and amplitude are set as private methods as they are not required anywhere outside the spring class, thus forming an encapsulation in the Spring class.

2) **Favour Composition over inheritance** : Composition is when an object contains another object and the another object cannot exist without the existence of the original object, it is known as composition.
   In our project, both sliding objects and rolling objects inherit from the Objects class and they cannot exist without the Objects class, thus encompassing composition.

3) **Program to an interface not implementation :** An interface in java is a blueprint of a class. It cannot have a method body; only abstract methods and variables. In our project, we currently don't have interfaces implemented.

4) **Strive for loose coupling between objects that interact :** Coupling refers to the amount of knowledge one object has about the other object it interacts with. It reduces the chances of a change made in one element leading to unexpected changes in the other element with which it is interacting. In our project, the object box interacts with the object spring and neither objects have any properties which can cause unexpected changes to other object; thus leading to loose coupling.

5) **Classes should be open for extension and closed for modification** : It refers to classes which can allow its behaviour to be extended to other classes without modifying its source code. In our project, we currently don't have interfaces implemented.

6) **Depend on abstraction, not on concrete classes** : Abstract class is a class declared with an abstract keyword, and may have both abstract and non-abstract methods. It is used for hiding the implementation details and just showing the functionality. In our project, we haven't used any abstract class right now but we could have used it in places like the "Objects" class. The "Objects" class could have been made an abstract class instead of a concrete class.

Strategy Design Pattern could have been used in our project. In this design pattern, we use an interface to apply an algorithm, which is then used for implementation in various other algorithms. In our project also, "Objects" class could have been made interface which would

then further be implemented to all the objects (sliding objects, rolling objects etc) for applying their algorithms.