

Project 2: - Spring, Action and Forces

Made by: - Deep Pandya (2020A7PS0148P) and Harsh Priyadarshi (2020A7PS0110P)

The project is based on simulation of physics problems based on classical mechanics which includes Sliding motion in plane, projectile, inclined surface, rolling motion on plane and inclined surface, two block pulley system, Spring problems on rolling objects as well as sliding objects based on lagrangian Physics, collision of two different objects and Checks if an object Topples or not under specific conditions.

It includes Following Classes:-

1)Main

It introduces Main menu and navigates between different choices based on the input by the user. It also includes receiving the input and instantiating the objects of all the other classes as need be.

2)Coordinates

It includes the x coordinate and y coordinate with getters and setters for all objects of Object class , SlidingObjects Class and RollingObjects Class.

3)Ground

It includes the basic properties of the surrounding on which formulas depend like coefficient of friction, coefficient of restitution and acceleration due to gravity with getters and setters and constructors.

4)Object extends Ground

It includes the basic properties of all objects irrespective of SlidingObjects or RollingObjects which are mass, velocity, weight, acceleration, coordinates and objectID.

It includes the constructor that initializes different values based on fillObject function in main.

5)SlidingObjects extends Object implements Driver

SlidingObjects Class has the dimensions of length and height as required to check for toppling, in other cases it works the same as Object class and gets instantiated through constructor by fillSO method in main. It implements driver class and has simulate function which includes simulating the problems by calling individual functions. It includes methods simulate which includes polymorphism , projectile to simulate projectile motion, planeGround to simulate object moving on ground and planeSpring for object attached to Spring and inclinedPlane to simulate object moving down the incline and topple to see if an object topples or not.

6) RollingObjects extends Object implements Driver

RollingObjects Class has the required radius, radius of gyration and omega to calculate the coordinates of an object at any time t under specific simulations. These values will be taken from the user using fillRO method in main and initialized using the constructor defined. It includes methods simulate which it gets from driver class , RollingSpring to simulate rolling of an object attached with a spring and plane_roll for rolling on plane surface and slope_roll for rolling on inclined plane.

Interface used:-

Driver

It has Simulate method required in both SlidingObjects and RollingObjects which takes in the input from the user, decides which simulation to perform, asks for timeslice and time and calls the required function accordingly.

Project Navigation and Test cases

Main Menu

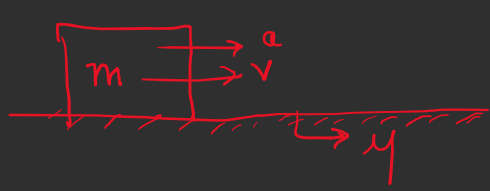
```
SPRING, ACTION AND FORCES SIMULATOR
what would you like to do?
1.No Roll(Sliding and projectile)
2.Roll
3.Collide
4.Spring
5.String
6.Toppling from external Force
7.Exit
Enter Your Choice:
```

Case1) No Roll(Sliding and projectile) (executed by method **simulate**)

```
please choose a simulation to perform
1.Object sliding on plane surface
2.object doing projectile motion
3.object on slope
```

1.1) Sliding on Plane surface (executed by method **planeGround**)

```
Please enter the values of Coefficient of friction
0.25
please enter initial coordinates X and Y
5
10
Please enter initial velocity:-
30
Please enter initial acceleration:-
2
Enter the timeslice and total time respectively :
5
25
coordinates of the object after 0.0 seconds is (5.0,10.0).
coordinates of the object after 5.0 seconds is (149.375,10.0).
coordinates of the object after 10.0 seconds is (282.5,10.0).
coordinates of the object after 15.0 seconds is (404.375,10.0).
coordinates of the object after 20.0 seconds is (515.0,10.0).
Object comes to rest after 66.66666666666664 seconds and at coordinates (1004.9999999999997,10.0).
the final position after time 25.0 is (614.375,10.0).
Press enter to proceed to Menu
```

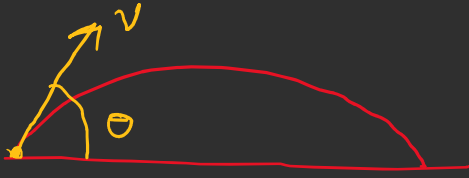


Walkthrough:- Takes the above values and checks if $\text{acceleration} < \text{frictional retardation}$ and makes two cases accordingly, 1) object stops in which case coordinates don't change after stopping time 2) object does not stop, and acceleration keeps increasing the velocity.

1.2) Projectile Motion (executed by method **projectile**)

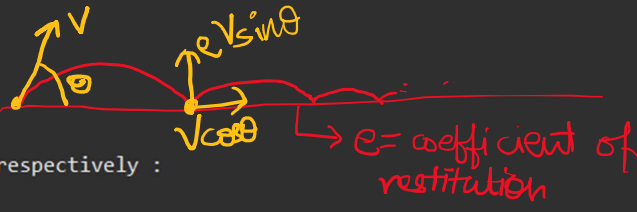
Output 1

```
please enter the values of angle of Projection (in Degrees) and Coefficient Of Restitution
30
0.5
please enter initial coordinates X
5
Please enter initial velocity:-
30
Enter the timeslice and total time respectively :
0.1
1.5
coordinates of the object after 0.0 seconds is (0.0,0.0).
coordinates of the object after 0.1 seconds is (2.598076211353316,1.451).
coordinates of the object after 0.2 seconds is (5.196152422706632,2.804).
coordinates of the object after 0.3000000000000004 seconds is (7.794228634059949,4.058999999999999).
coordinates of the object after 0.4 seconds is (10.392304845413264,5.215999999999999).
coordinates of the object after 0.5 seconds is (12.99038105676658,6.275).
coordinates of the object after 0.6 seconds is (15.588457268119896,7.236).
coordinates of the object after 0.7 seconds is (18.186533479473212,8.098999999999998).
coordinates of the object after 0.7999999999999999 seconds is (20.784609690826528,8.863999999999999).
coordinates of the object after 0.8999999999999999 seconds is (23.38268590217984,9.530999999999999).
coordinates of the object after 0.9999999999999999 seconds is (25.980762113533157,10.1).
coordinates of the object after 1.0999999999999999 seconds is (28.578838324886473,10.570999999999996).
coordinates of the object after 1.2 seconds is (31.176914536239792,10.943999999999997).
coordinates of the object after 1.3 seconds is (33.77499074759311,11.218999999999998).
coordinates of the object after 1.4000000000000001 seconds is (36.373066958946424,11.395999999999997).
Press enter to proceed to Menu
```



Output -2 – give the points where object keeps hitting the ground.

```
please enter the values of angle of Projection (in Degrees) and Coefficient Of Restitution
30
0.5
please enter initial coordinates X
5
Please enter initial velocity:-
30
Enter the timeslice and total time respectively :
2
10
coordinates of the object after 0.0 seconds is (0.0,0.0).
coordinates of the object after 2.0 seconds is (51.96152422706632,10.399999999999993).
Object hits the ground at 3.0612244897959178 seconds at coordinates (79.53294524550965,0).
Object hits the ground at 4.591836734693876 seconds at coordinates (119.29941786826447,0).
Object hits the ground at 5.357142857142856 seconds at coordinates (139.1826541796419,0).
Time Slice too large in comparison to current time of flight for x and y to be calculated
Press enter to proceed to Menu
```



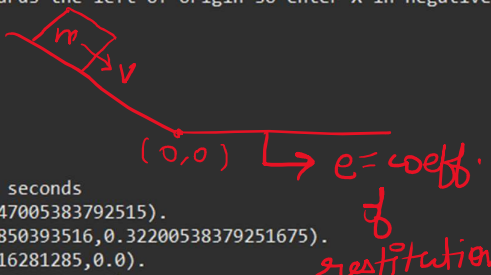
Assumptions:- impact of friction is negligible when the object strikes the ground.

Walkthrough- takes angle of projection and initial velocity, then keeps giving the output as long as the time slice is greater than the time of flight of nth collision. After every collision function is recalled via recursion with vertical velocity = $e \cdot v_y$ to give the point at which it hits the ground if total time is more than initial time of flight.

Limitations- Cannot give time wise position if time of flight of nth jump is less than time slice.

1.3)1.3) Object on Slope (executed by method `inclinedPlane`)

```
please enter the value of slope of the inclined plane and coefficient Of restitution
30
0.25
please enter initial coordinates X, the slope starts towards the left of origin so enter X in negative
-20
Please enter initial velocity:-
20
Enter the timeslice and total time respectively :
1
5
Initial coordinates are (-20.0,11.547005383792515)
Object leaves the inclined surface at 1.0257983996743743 seconds
coordinates of the object after 0 seconds is (-20.0,11.547005383792515).
coordinates of the object after 1 seconds is (-0.5577296850393516,0.32200538379251675).
coordinates of the object after 2 seconds is (4.8710080016281285,0.0).
coordinates of the object after 3 seconds is (9.871008001628129,0.0).
coordinates of the object after 4 seconds is (14.871008001628127,0.0).
Press enter to proceed to Menu
```



Assumptions:- frictionless surface and slope starts towards left (mentioned to the user for easy usage).

Walkthrough:- takes in the slope and x-coordinate, gives time at which it leaves the inclined surface and coordinates at all time slices.

Limitations:- requires frictionless surface.

Case2) Rolling (executed by method `simulate`)

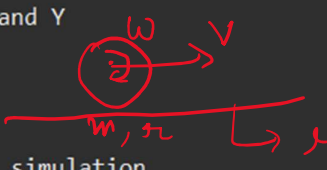
```
rolling
please choose a simulation to perform
1.Object Rolling on plane surface
2.object rolling on inclined surface
```

2.1) Object Rolling on plane surface (executed by methods `plane_roll`)

```

rolling
please choose a simulation to perform
1.Object Rolling on plane surface
2.object rolling on inclined surface
1
Please enter the values of Coefficient of friction
0.25
please enter initial coordinates X and Y
5
10
Please enter initial velocity:-
30
Please enter the type of object for simulation
1. Cylinder 2. Sphere
3. Ring
2
Please enter initial angular velocity:-
20
Enter radius
0.5
Enter the timeslice and total time respectively :
1
5
coordinates of the object after 0.0 seconds is (5.0,10.0).
coordinates of the object after 1.0 seconds is (35.0,10.0).
coordinates of the object after 2.0 seconds is (65.0,10.0).
coordinates of the object after 3.0 seconds is (95.0,10.0).
coordinates of the object after 4.0 seconds is (125.0,10.0).
Press enter to proceed to Menu

```



Walkthrough:- takes in the input of angular velocity, velocity of COM, radius, calculates the rolling motion velocity and gives the output w.r.t. time and time slices.

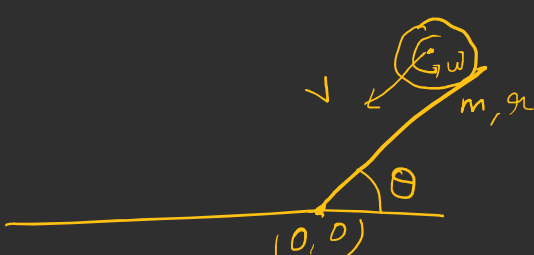
Future Work:- can also be extended to 3 dimensional geometry.

2.2) Object Rolling on inclined surface (executed by methods `slope_roll`)

```

please enter the value of slope of the inclined plane
30
please enter initial coordinates X, the slope starts towards the right of origin so enter X in positive
30
Please enter initial velocity:-
20
Please enter the type of object for simulation
1. Cylinder 2. Sphere
3. Ring
1
Please enter initial angular velocity:-
20
Enter radius
0.5
Enter the timeslice and total time respectively :
1
5
coordinates of the object after 0.0 seconds is (30.0,17.32050807568877).
coordinates of the object after 1.0 seconds is (7.550000000000001,4.358994532381675).
coordinates of the object after 2.0 seconds is (-19.799999999999997,0.0).
coordinates of the object after 3.0 seconds is (-52.05,0.0).
coordinates of the object after 4.0 seconds is (-89.19999999999999,0.0).
Press enter to proceed to Menu

```



Walkthrough:- takes in the input of angular velocity, velocity of COM, radius, calculates the rolling motion velocity and gives the output w.r.t. time and time slices.

Limitations:- initial angular acceleration = 0

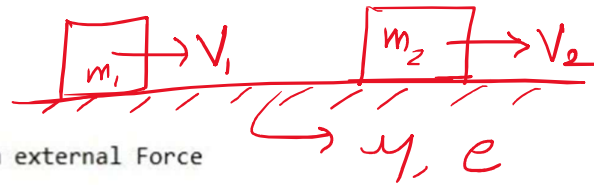
Future Work:- can also include initial angular acceleration.

Case3) Collide (executed by methods **simulate**)

```

SPRING, ACTION AND FORCES SIMULATOR
what would you like to do?
1.Slide
2.Roll
3.Collide
4.Pendulum
5.Spring
6.String
7.Toppling from external Force
8.Exit
Enter Your Choice:
3
Please enter object on the left first
Please enter the values of Coefficient of friction
0.1
Please enter the values of Coefficient of restitution
1
please enter initial coordinates X and Y respectively
0
0
Please enter initial velocity:-
20
Please enter mass of object
1
please enter initial coordinates X and Y respectively
19.5
0
Please enter initial velocity:-
0
Please enter mass of object
1
Enter the timeslice and total time respectively :
0.2
2
coordinates of the object1 after 0.0 seconds is (0.0,0.0).
coordinates of the object2 after 0.0 seconds is (19.5,0.0).
coordinates of the object1 after 0.2 seconds is (3.9804,0.0).
coordinates of the object2 after 0.2 seconds is (19.4804,0.0).
coordinates of the object1 after 0.4 seconds is (7.9216,0.0).
coordinates of the object2 after 0.4 seconds is (19.4216,0.0).
coordinates of the object1 after 0.6000000000000001 seconds is (11.823600000000003,0.0).
coordinates of the object2 after 0.6000000000000001 seconds is (19.3236,0.0).
coordinates of the object1 after 0.8 seconds is (15.686399999999999,0.0).
coordinates of the object2 after 0.8 seconds is (19.1864,0.0).
Objects will collide at 0.975seconds
coordinates of the object1 after 0.975 seconds is (18.10258125,0.0).
coordinates of the object2 after 0.975 seconds is (18.10258125,0.0).
coordinates of the object1 after 1.175 seconds is (17.89188125,0.0).
coordinates of the object2 after 1.175 seconds is (21.89188125,0.0).
coordinates of the object1 after 1.375 seconds is (17.64198125,0.0).
coordinates of the object2 after 1.375 seconds is (25.64198125,0.0).
coordinates of the object1 after 1.575 seconds is (17.35288125,0.0).
coordinates of the object2 after 1.575 seconds is (29.352881249999996,0.0).
coordinates of the object1 after 1.775 seconds is (17.024581249999997,0.0).
coordinates of the object2 after 1.775 seconds is (33.02458125,0.0).
coordinates of the object1 after 1.9749999999999999 seconds is (16.65708125,0.0).
coordinates of the object2 after 1.9749999999999999 seconds is (36.65708125,0.0).
Press enter to proceed to Menu

```



Walkthrough :- initializes two SlidingObjects and then passes it to simulate to get the time slice and time for each object and then gives the coordinates of both the objects back.

Future Work :- can also be extended to oblique collisions and 3D.

Case4) Spring (executed by methods **simulate**)

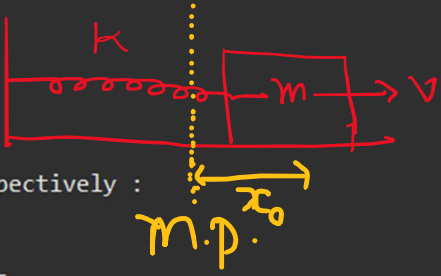
```
SPRING, ACTION AND FORCES SIMULATOR
what would you like to do?
1.No Roll(Sliding and projectile)
2.Roll
3.Collide
4.Spring
5.String
6.Toppling from external Force
7.Exit
Enter Your Choice:
4
spring
Please choose a category
1.Without rolling
2.With Rolling
```

Walkthrough of 4.1 and 4.2 :- Takes the input and uses langrangian method to solve and obtain the equation of oscillatory motion.

Future Work :- Damping oscillations can be introduced and langrangian method itself is applicable to wide range of physical setups increasing the cases where a programme can be coded.

4.1) Without Rolling (executed by **planeSpring**)


```
please enter initial coordinates X and Y
5
10
Please enter initial velocity:-
30
Please enter mass:-
5
Enter the timeslice and total time respectively :
2
10
please enter the spring coefficient : -
1000
Please enter initial displacement from mean position x = 5.0 at t=0 seconds
2
coordinates of the object after 0.0 seconds is (7.0,10.0).
coordinates of the object after 2.0 seconds is (2.993072087776124,10.0).
coordinates of the object after 4.0 seconds is (7.013657639486455,10.0).
coordinates of the object after 6.0 seconds is (2.9798114827756232,10.0).
coordinates of the object after 8.0 seconds is (7.026519900510776,10.0).
coordinates of the object after 10.0 seconds is (2.967348835881064,10.0).
Press enter to proceed to Menu
```



4.2) With Rolling (executed by method `rollingSpring`)

```
spring
Please choose a category
1.Without rolling
2.With Rolling
2
please enter initial coordinates X and Y
5
10
Please enter initial velocity:-
30
Please enter mass:-
5

Choose an object
1.Solid Sphere
2.HollowSphere
3.Disc or solid cylinder
4.Custom radius of gyration
1
Please enter Radius in centimeters:-
50
Enter the timeslice and total time respectively :
2
10
please enter the spring coefficient : -
1000
Please enter initial displacement from mean position  $x = 5.0$  at  $t=0$  seconds
1
coordinates of the object after 0.0 seconds is (6.0,10.0).
coordinates of the object after 2.0 seconds is (3.1676031664939477,10.0).
coordinates of the object after 4.0 seconds is (2.7687655728439458,10.0).
coordinates of the object after 6.0 seconds is (5.33317319835382,10.0).
coordinates of the object after 8.0 seconds is (7.455102043885144,10.0).
coordinates of the object after 10.0 seconds is (6.3164728299187995,10.0).
Press enter to proceed to Menu
```

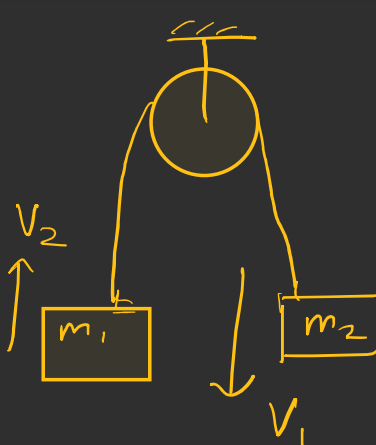


The diagram illustrates a physics experiment setup. A horizontal spring is attached to a vertical wall on the left. A sphere is attached to the other end of the spring. A yellow arrow labeled v points to the right from the center of the sphere, representing its linear velocity. A yellow curved arrow labeled ω points clockwise around the center of the sphere, representing its angular velocity. The sphere is shown in contact with a horizontal surface, indicating it is rolling without slipping.

case5) String (executed by methods `simulate`, `class String`)

Walkthrough: - Two Block pulley System that takes the x and y coordinates of two objects, starts the motion till the string are taught and then gives the values as the heavier object falls down and lighter object goes up and flips over the pulley and performs free fall.

```
Enter Your Choice:
5
string
For Object 1
please enter initial coordinates X and Y
5
10
Please enter mass of object
30
For Object 2
please enter initial coordinates X and Y
5
20
Please enter mass of object
20
Enter the timeslice and total time respectively :
2
10
coordinates of the object1 after 0.0 seconds is (5.0,13.92).
coordinates of the object2 after 0.0 seconds is (5.0,16.08).
coordinates of the object1 after 2.0 seconds is (5.0,25.68).
coordinates of the object2 after 2.0 seconds is (5.0,2.16).
coordinates of the object1 after 4.0 seconds is (5.0,37.44).
coordinates of the object2 after 4.0 seconds is (5.0,13.92).
coordinates of the object1 after 6.0 seconds is (5.0,49.2).
coordinates of the object2 after 6.0 seconds is (5.0,25.68).
coordinates of the object1 after 8.0 seconds is (5.0,60.960000000000001).
coordinates of the object2 after 8.0 seconds is (5.0,37.44).
Press enter to proceed to Menu
```

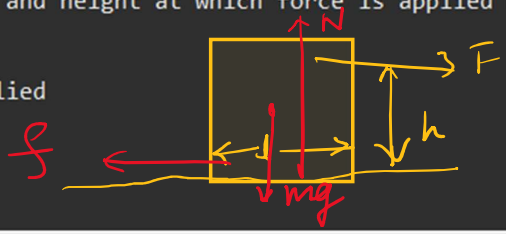


case6) Toppling (executed by methods `simulate`, `topple`)

walkthrough: - Takes input of the blocks dimensions, friction and force at a height h and checks if the force applied causes sliding, toppling, both or none of these.

Future Work: - can also be extended to pyramids and different shapes and inclined surfaces.

```
toppling from external force
Please enter the values of Coefficient of friction
0.25
Please enter mass:-
5
Please enter length along which and height at which force is applied respectively
10
20
Please enter External Force applied
2000
Object topples
Press enter to proceed to Menu
```



Case7) Exit

Video Explanation Link

<https://drive.google.com/file/d/1TDsi3xIVW0D9gpkwM1rKWp3JIJlcQzsd/view?usp=sharing>

Github repository link

https://github.com/itsharsh334/OOPS_Project2