Name- Ritik Upmanyu 2019B3A70517P, Tejas Deshpande 2019B3A70562P

Documentation

Github link of final project- https://github.com/RitikUpmanyu/OOP_spring_and_forces

Video link- https://drive.google.com/drive/folders/12D_yiiSaCx7hLFVZqZXlSFurc1CM1HoP?usp=sharing

//dependencies

Java – libraries=

Processing

OpenCSV

//How to run-

The entry point of the code in the main function of the driver class

In driver class we need to set the input and output paths explicitly in the variable INPUTPATH and OUTPATH, if not set then they are default set to input.txt and output.csv in the main project folder.

Alternatively, run the jar executable file. There needs to be input.txt, images in the same folder as jar file.

NOTE- jar file is present in Final-project folder with the name final-jar.jar

Code is present in spring_and_forces folder

//What to write in input.txt

Define Global Vars: in order

acceleration due to gravity

coefficient of friction

spring constant(if not applicable put 0)

wind(positive means right, negative means left)

Enter angle of slope: (put 0 in case of ground)

Enter number of objects:

Please enter N lines each containing information for an object in following format:

Arguments of each object should be given in specified format as mentioned below.

IDs can be any random unique number, typeid describes what kind of Object it is.

Note that arguments marked * are required!

Following are the valid Object Ids and formats of arguments for the available objects:

For Object: Ball;

ID:00 TypeID: 1, x_coordinate, x_velocity, y_velocity, radius, mass

For Object: Box;

Id:01 TypeID: 2, x_coordinate, x_velocity, y_velocity, length, width, mass

For Object: Ring;

Id:02 TypeID: 3, x_coordinate, x_velocity, y_velocity, radius, mass

For Object: Cylinder;

Id:03 TypeID: 4, x_coordinate, x_velocity, y_velocity, radius, mass

For Object: Spring;

Id:04 TypeID: 5, Obj id of anchor, Obj id of bob, restlength, radius, 0/1(make anchor locked?)

Note: x and y positions can vary from 0 to 1000;

Example input->

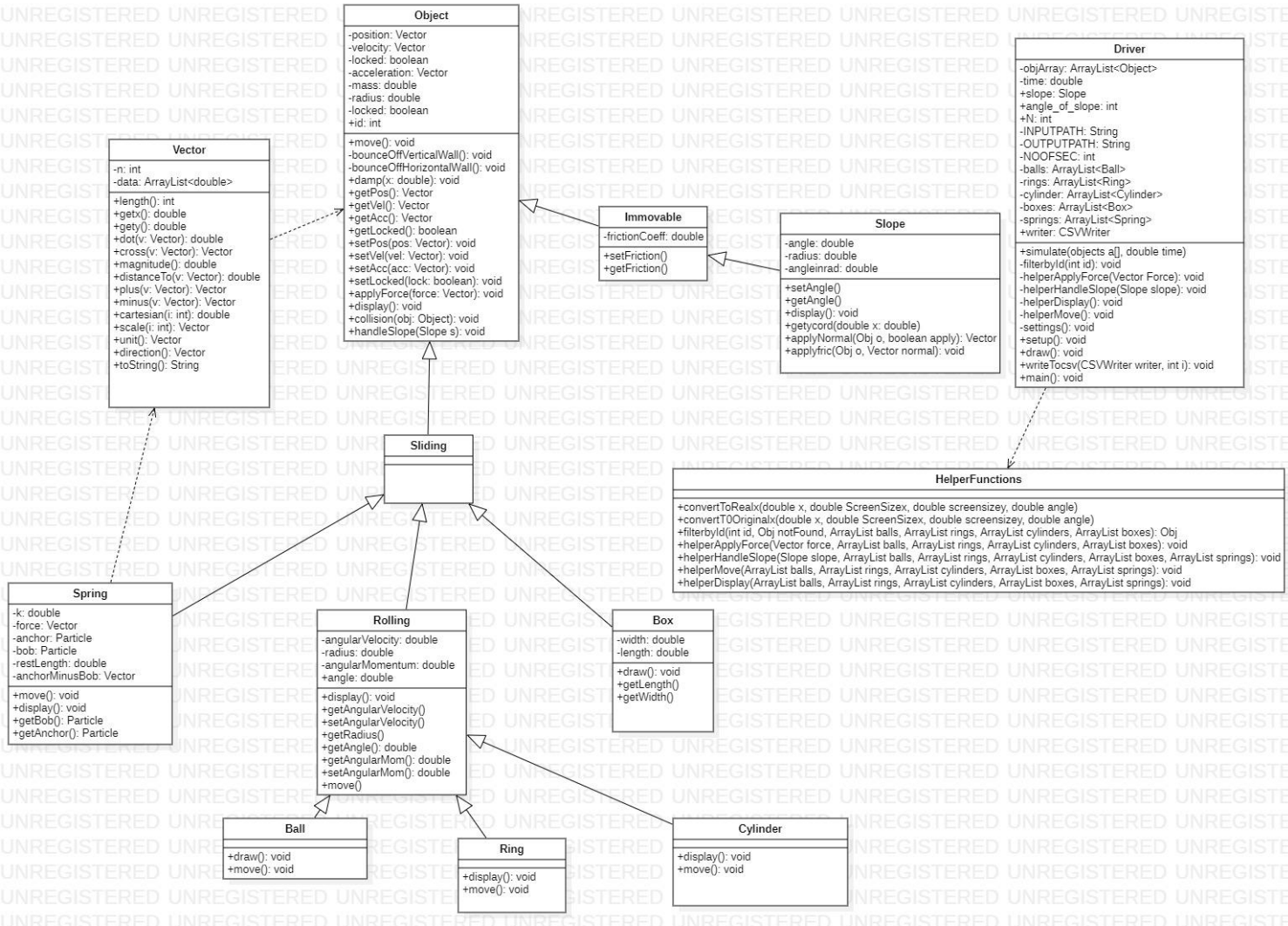
//input.txt

```
9.8
0.0
0.02
0
0
5
0 1 90 10 0 50 10
1 2 70 1 0 50 100 10
2 3 90 2 0 50 10
3 4 8 0 0 50 10
4 5 1 2 100 50 0
```

In output.csv we get position and angle data for all objects for first NOOFSEC seconds.

(NOOFSEC can be defined in driver class, default is 50).

UML



Description of classes-

//Vector.java

Not made by us, this was made by Princeton university and was available in the resources provided for this project.

This class is used for making n-dimensional vectors and performing several operations on it like dot product, cross product, addition, subtractions, magnitude, direction and unit vector.

//obj.java

This is the base object class which every object extends.

Attributes->

Position, velocity, acceleration, mass (defaulted to be 1, can be changed), radius, locked(if true, it can't move), and id

Methods->

Move() this method updates the position, velocity and acceleration from the previous frame to the next one

handleSlope(Slope s) this method is used to apply the normal force, friction force and not let objects fall into the slope.

damp() for creating damping effect

applyForce(Vector force) whatever force is passed into this function will be applied to the current object (acceleration will be updated)

display()- for displaying in gui

//immovable

Extends Obj

Attributes- friction coeff.

//Slope

Extends Immovable

Attributes- angle, if angle=0, it will act as ground (that's why no ground class)

Methods- applyNormal(Obj o) – applies normal force according to angle,

Applyfric(Obj o) – applies friction in the opposite direction of motion of object, along the angle of slope.

//Sliding Obj

Just a skeleton class with no special attributes or methods, (just to be organised)

//Box

Extends sliding Obj

Attributes- angle, width, length

//Rolling Obj

Extends Sliding Obj

Attributes- Angular velocity, angular momentum, angle

//Ball

Extends Rolling Obj

//Cylinder

Extends Rolling Obj

//Ring

Extends Rolling Obj

//Spring

Extends Obj

Attributes- spring coeff., force, Obj anchor, Obj bob, restLength

//helperFunctions

Several helperfunctions for the driver class like displaying, moving each object in all object type arraylists, filtering objects by id

All classes have move and display methods according to their type

Assumptions-

1. Mass=1 (can be changed easily)
2. Pure rolling in every case.
3. Initial y coordinate is 0 for all objects i.e. they are all falling from sky (no collision has been modelled so it wouldn't make a difference)

Known limitations/bugs-

1. One small bug is associated with friction, when friction is applied and body stops, there a little motion still remaining after some time
2. If you increase angle of slope such that initial condition coordinates get inside of slope-> undefined behaviour-> objects disappear
3. Objects on top of each other has not been modelled.
4. Rolling with sliding hasn't been modelled. Very simple rolling motion is modelled according to velocity.
5. Collision hasn't been modelled, except with walls (which has also hot been used in the final submission)

Future work that is possible along the same lines as the project

1. Perfect modelling of rolling with sliding physics
2. Objects stacking and collision with each other
3. We can also take input by mouse, set the initial conditions and then simulate
4. A full-fledged 2d game engine can be developed using this model.