## **Collision Simulator- Design Document**

Michael Zhou, Yash Shah, Tony Li

**Product Description**

Collision Simulator is a program that can reproduce three types of 1-dimensional collisions: inelastic, completely inelastic and elastic. Given the required information, the program will display a simulation of the collision and calculate initial or final velocities of both colliding objects.

**Purpose**

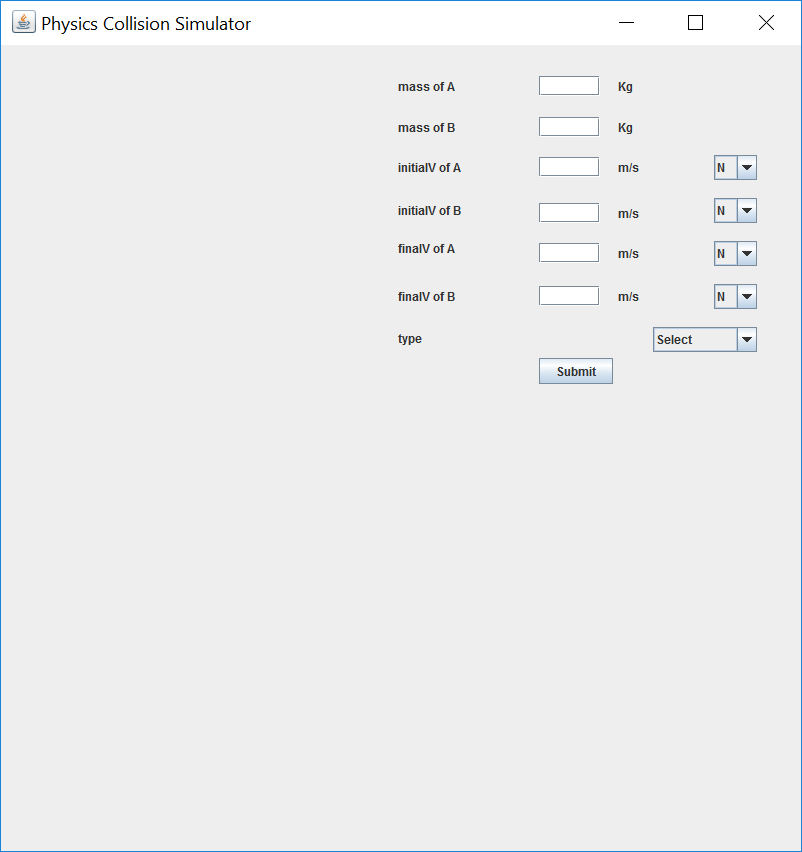
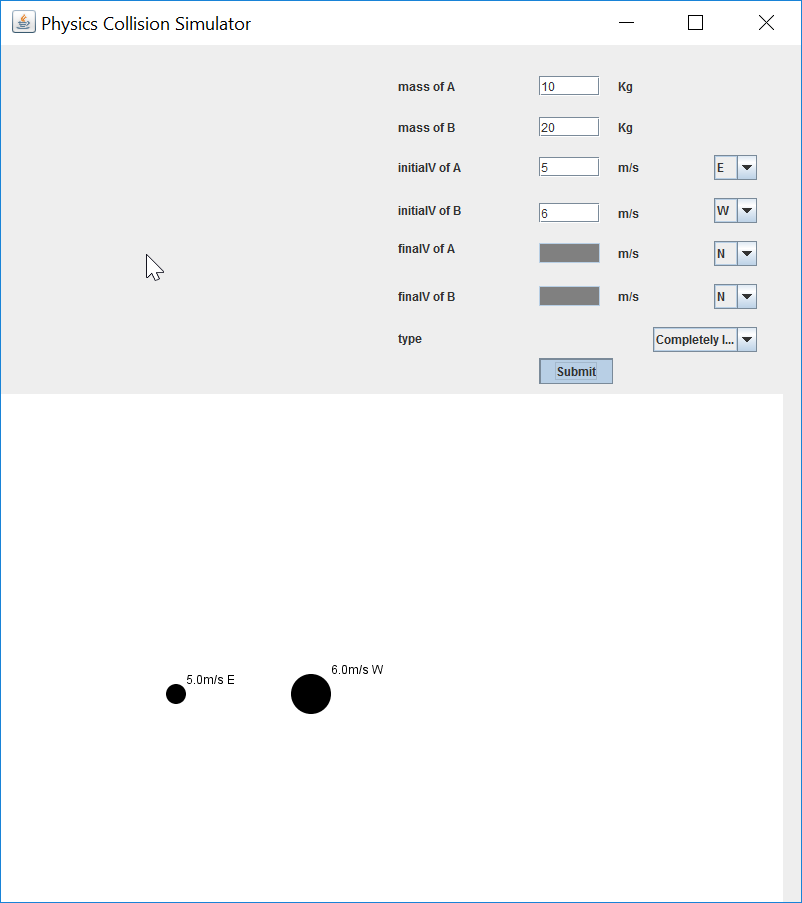
As high school students, our goal was to create an app that was useful for school purposes. In physics, we saw that students often found it hard to imagine how objects were colliding, since there were 3 different types of collisions, each with their own requirements. The Collision Simulator gives these students an opportunity to understand how things collide using a GUI, and also helps them check their homework answers (and potentially study for a test) with the final velocities that are displayed on the screen.

**Features of Collision Simulator**

Users are first presented with a GUI in which they can input information about the collision:

* Masses of both objects
* Initial velocities of both objects
* Final velocities of both objects
* Type of collision

Depending on the type of collision, the user’s input will be restricted to the variables needed to perform calculations for the collision. Each collision is explained in the user manual. The program then will calculate the missing variables, with different calculations for different types of collisions. Additionally, it will animate a simulation of the collision on screen.

As a result, since the completely inelastic collision only uses two variables, the other two text fields are greyed out

In this example, the user chose the completely inelastic collision

This screen shows what the user can potentially enter based on the collision

More about the GUI and the errors that can show up is explained in the User Manual.

**Issues**

Although the project seemed simple to us at first, there were many issues we had while coding it. Since the 3 types of collisions require math that involves rearranging equations, substituting variables and expanding binomials, we took the majority of the time trying to plan out the math behind the project instead of starting to program it immediately.

Also, since the Java in built JFrame design code is very rigid to work with, we were very frustrated in trying to edit the textfields, buttons etc. using our own code. Painting the animation on the screen was also a problem in itself. All these issues were resolved, and did not pose as a problem afterwards.

**Bug Checking**

It is a common saying in the field of software testing—no product is ever bug free. All bugs cannot be fixed in a program due to limited time and resources. However, we tried our best to test and solve any bugs that came our way. A few include not allowing negative mass input, greying out input boxes that are not needed, showing error messages when required velocities are not entered etc.

**Completion Plan**

We divided the entire project into modules that should have been finished at a certain date. The plan with each group member’s contribution is shown below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Task** | **Team members responsible** | **Target completion date** | **Status** |
| **Plan overall design** | **All** | **4/13** | **Done** |
| **Design document/documentation** | **Tony, Yash** | **4/27** | **Done** |
| Product description | Tony | 4/17 | Done |
| Feature description | Tony | 4/17 | Done |
| User manual | Tony | 4/27 | Done |
| Completion plan | Tony | 4/27 | Done |
| Issues, Bugs, Screenshot annotations | Yash | 4/28 | Done |
| **Module 1: GUI** | **Yash, Tony** | **4/17** | **Done** |
| Labels | Yash | 4/17 | Done |
| Text fields (input restricted to positive numbers) | Yash, Tony | 4/27 | Done |
| Combo boxes | Yash | 4/17 | Done |
| Button | Yash | 4/17 | Done |
| **Module 2: Calculations for colliding objects** | **Michael, Yash** | **4/25** | **Done** |
| Inelastic collisions | Michael, Yash | 4/25 | Done |
| Completely inelastic collisions | Michael | 4/25 | Done |
| Elastic collisions | Yash | 4/25 | Done |
| **Module 3: Displaying objects colliding** | **Michael** | **4/17** | **Done** |
| Move objects in the specified direction with the specified velocity | Michael | 4/17 | Done |
| Collision animation | Michael | 4/17 | Done |
| Display results on the screen | Michael | 4/17 | Done |