Welcome to my Thermodynamics Project!

There's lot to show you in this merry land! So without further ado, let's proceed into showing you around this workspace as well as showing how to use it.

`Martin He, CID: 02039120, Imperial College London, Department of Physics`

Dependencies, all of which are found on the Anaconda Environment in the Blackett Computing labs, as per the request by the Head of Computing, Professor Richards. These include the following:

`numpy scipy matplotlib.pyplot pylab pandas copy.deepcopy seaborn time sys os itertools heapdict`

If there is a package used, but which is not listed, it is part of the default Python packages, though these are avoided where possible in favour of using the numpy equivalent, for instance with `np.ndenumerate` or `np.array`.

Finally, for some statistics, because STEM people like numbers:

```
GitHub Commits: 30  Date of Project Completion: December 6, 2022
Lines of Code: 255 (ball.py) + 1230 (simulation.py) +
Hours of Life Spent Debugging: 20+ hours
Hours Spent Debugging Other People's Code: 5+ hours  Number of Hot Chocolates Made: 35
Number of Albums Listened: 85 (according to Spotify, mostly on repeat)
Hours Spent Choosing Axes Fonts: 3
```

Python Files

First of all, let's introduce you to the Python files required.

`ball.py [Required]`

This aptly-named Python file, unsurprisingly, houses the Ball class, where the ball-related methods can be found for acting on singular spheres at a time. Additionally, some people around me in the computing lab told me that the Container class is supposed to inherit from the Ball class, so I listened, and then realised that this wasn't actually a requirement, so I'm guess I'm also stuck in the container being inside the Ball.py file.

`optional.py [Optional]`

This Python file, also unsurprisingly, is optional. It contains a few extra simulations I tried to complete just for the sake of completion, as well as attempting to look better than I actually am (i.e. putting make-up on a pig).

`simulation.py [Required]`

Simulation.py houses the Simulation class, which controls the running of the simulation. It is split neatly into nine hopefully easy-to-read sections, which can be described as the following:

- **Ball Initialisation:** contains the `__init__` method.
- **Ball Information:** contains `__repr__` , `__str__` , and `glossary` .
- **Ball Property:** contains corresponding functions for returning information.
- **Ball Movement:** contains the `next_collision` function, essentially.
- **Ball Randomisation:** contains randomisation functions.
- **Ball Recording:** contains the prefix `record_` , and records.
- **Ball Miscellaneous:** contains functions that don't fit elsewhere.
- **Ball Run:** contains the `run` method.
- **Ball Plots:** produces optional plots of the simulations.

With these functions the following tests can be performed:

- **Task 5:** Rigid Spherical Collisions in a Container
- **Task 6:** Conservation Laws
- **Task 9:** Absolute and Relative Distance of Balls
- **Task 10:** Ideal Gas Relations
- **Task 11:** Further Pressure Calculations
- **Task 12:** The Ideal Gas Law
- **Task 13:** The Maxwell-Boltzmann Distribution
- **Task 14:** The Van der Waals Law
- **3D Plots [Optional]:** 3D continuums of various properties, for several variables.
- **Brownian [Optional]:** Recreates Robert Brown's investigation with Pollen.

Documentation

Getting Started

Environment

The recommended environment for running this simulation is on `Ubuntu 22.04 LTS` with `Visual Studio Code` installed, as well as `Python 3.10` and all the Python packages which will be listed below.

Anaconda is also a possible operating environment, though is not recommended. Any Python packages which for some reason are not included with Anaconda can be obtained by inputting the following command:

```
conda install [$PACKAGENAME$]
```

Python Packages

Having installed as required for your environment, perform the following command:

```
pip3 install numpy scipy matplotlib.pyplot pylab pandas copy.deepcopy seaborn
time sys os itertools heapdict
```

Then after installation, check that all your modules are installed by performing:

```
pip3 list
```

If successful, all the listed packages above should be installed, and you are ready!

Installation

Now, we move onto installing the program so that we can use it.

i. Installation on Windows

If you are on Windows, like the vast majority of Planet Earth, you are in luck! I have custom-made a script just for Windows to copy the files to a custom install directory, for you to use. This can be specified in the terminal program. Additionally, I will try to create a Start Menu shortcut. This will provide a link to the testing file for you to use.

Installation on Linux

If you are on Linux, well done. You're on your own, kid.

Installation on Mac

If you're on Mac, I have tried to make a set-up script, though I do not own a Mac.

ii. FAQ - Frequently Asked Questions

Q. My balls are sticking together. Why? A. Assuming you're talking about the simulation, I have absolutely no idea. Sounds like an error in the Ball.py. Although this error has never occurred for me, so it's probably not my fault.
Q. My balls are escaping the container. Why? A. Again, this is highly improbable, but if this occurs, this didn't happen the last time I tested.
Q. Why are there so many package dependencies? Why not just stick to the Holy Trinity of np, plt and sp? A. This might be the most glaring of questions, and to that, I provide a thorough explanation. The use of numpy allows for data processing far faster than built in Python methods. Similarly, this applies to scipy. It is simply impossible for me to plot graphs without matplotlib, and pylab. The use of seaborn is to make my graphs prettier. Seaborn is also required for my 3D plots, since as far as I am aware, matplotlib does not have this capability. Itertools is always a godsend and makes the calculation of pair combinations for collisions far simpler. Deepcopy allow us to perform operations with the same ball object, without pointing to the same object. Another godsend. System and OS are essentially for loading and reading filepaths. Time allows me to record the global time"zone" of the simulation.

Q. Why do Ball, Simulation or Optional modules not load?
A. This is likely due to your operating environment. I would suggest you switch to VSCode and make the folder the directory with all these files directly in the chosen folder.

Q. Why did you make so many hot chocolates?
A. I like to run a lot (>10K), so the addition of calories is no worry.

Any additional questions can be fielded to `mah121@ic.ac.uk` .

Features

There are many features to this program, which have been listed before.

- **Task 5:** Rigid Spherical Collisions in a Container
- **Task 6:** Conservation Laws
- **Task 9:** Absolute and Relative Distance of Balls
- **Task 10:** Ideal Gas Relations
- **Task 11:** Further Pressure Calculations
- **Task 12:** The Ideal Gas Law
- **Task 13:** The Maxwell-Boltzmann Distribution
- **Task 14:** The Van der Waals Law
- **3D Plots [Optional]:** 3D continuums of various properties, for several variables.
- **Brownian [Optional]:** Recreates Robert Brown's investigation with Pollen.

Simulations

The simulations as shown above can be run quite simply. Below is a preview for each of the expected outputs of the simulations as shown graphically, and sample data can also be obtained in the `sample_data` folder.

Testing

The above simulations also offer testing capabilities. You can test for the conservation laws, as well as the ideal gas laws. Additionally, you can see if the absolute and relative distances are as expected. You can see if the obtained plots are in tune with the laws of Physics through a qualitative or quantitative comparison which is possible.

Plots

Plots are obtained, which are also highly customisable. For the plots to look the best, it is recommended that the following Typefaces are installed to be used in the axes plots:

```
Inter  CMU Serif  CMU Sans Serif  SF Mono  Times New Roman
```

If not, you can change which typefaces at the top of the `simulation.py` file.

3D Plots

This is a special feature. It is also highly time-consuming and memory-intensive. Using this feature, you can obtain a 3D plot for various features across a variety of variables, in order to gauge how adjusting increments of these variables affect the results.

Exporting Datasets

Another special feature, if you wish to use other programs to analyse the data.

Customisation

You can customise your fonts are specified in plots.
Other customisation options are included in the installer program.

License

Acknowledgements

There are many people to thank for the completion of this project. Most of all is my local Wetherspoons, the Coronation Hall, which provide fantastic food at an affordable price. I would also like to thank many people at Imperial College for helping, including but not limited to the demonstrators would always pointed me in the right direction, without revealing anything explicitly, in particular Contaldi who is a certified G.O.A.T, as well as Logan Filipovich who is rather hilarious. I

would also like to thank Wikipedia for helping with optimising various algorithms use, and the approach I do this coding.

I do *not* thank the sheer number of docstrings used during this code, which compose almost half of all the code that is written. I hope I have managed to strike the right balance between the use of docstrings and commenting being excessive and sparse.