SYNOPSIS

The project is based on the reflection and refraction by different types of spherical mirrors and lenses respectively.

It uses a graphical representation of the image formation taking into account the necessary values like focal length, object distance and the object height to accurately calculate and plot the image.

The program uses an interactive interface allowing the user to:

* Feed values from a text window which opens a graphics window with required data
* Manipulate previous values in graphics window which gets refreshed
* Save the current data (in a data file).
* Open previously saved data.

REASONS FOR PROJECT

The project provides a graphical approach to the topic which is required for a better understanding. Many cases can be handled more accurately with the tool.

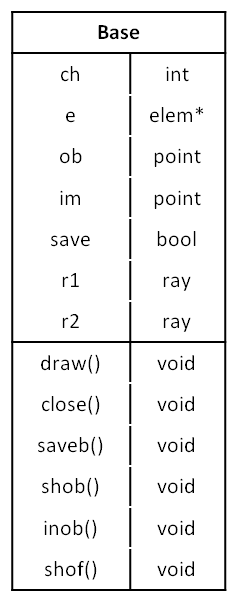
SYSTEM REQUIREMENTS

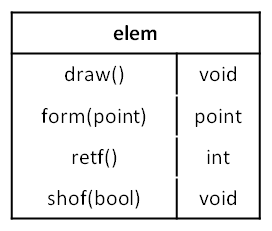
Requirements for program execution:

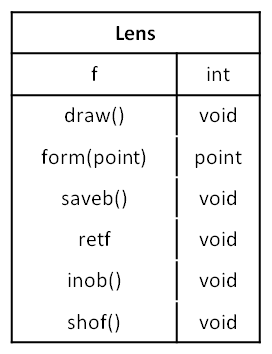
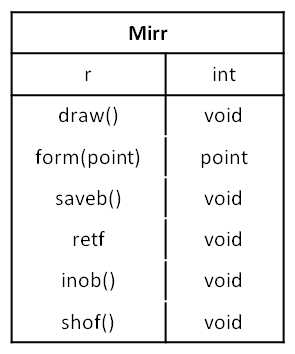
* + OS – Windows XP/ Appil or higher
  + RAM – 1GB or higher

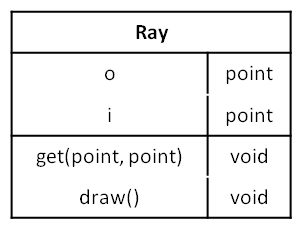
Requirements for Compilation and debugging:

* + OS – Windows XP/ Appil or higher
  + RAM – 1GB or higher
  + C++ compiler with BGI graphics support (e.g. Bloodshed Dev C++, Visual Studio)

CLASS DIAGRAM







STANDARD HEADER FILES

iostream

fstream

math.h

FLOWCHART

SOURCE CODE

SCREENSHOTS

SCOPE FOR IMPROVEMENT

The project is a basic version of the intended result and requires a lot of upgrades, patches and debugging to include more cases, add more features and improve its efficiency.

We find the following to be major areas which require attention:

Known Bugs and errors:

* Certain cases (e.g. image at infinity) cannot be displayed appropriately.
* Numeric details (distance, height, focal length) are not displayed in the graphics window correctly (feature removed).

Areas of Improvement:

* Upgrade to include comparison of two or more cases simultaneously.
* Improve the save function to remove duplicates and include a delete option. This can be done by upgrading to databases from flat files.
* Make the interface more user-friendly, e.g. by making the graphics more dynamic, adding more features like control of curvature, number of rays, etc.

cover,certificate,acknowledgements,index,synopsis(done),reason for project(done),system requirements(done), class diagrams with function description(func desc left), standard header files(done), flow chart(done), source code(back to back), screenshots, scope for improvement(done), bibliography, webliography.