For this brief document, I will provide a short class description for each of the classes contained in this project.

Scheduler- Interface
This class provides the basis on how each of the algorithms work, and to which this class is implemented from. Contains the getNextJob methods and returnJobList methods which are an integral part of each algorithm.

SchedulerException
Prints out exceptions given from algorithm classes based oon the exception. Methods take in the exception annd then show this to the user.

AbstractScheduler
Acts as a template-style algorithm scheduler class, and can be extended by schedulers to perform functions already within this classs. GetNextJob method is final in this class to prevent overriding it in a class extending it.

HighestPriority
Implements the scheduler interface and contains the algorith to sort processes by their priority. Returns to the head of the queue the one which has the highest priority.

ShortestJobFirst
Impliments the scheduler interface again and will keep checkinf for the highest priority task in the job queue and returning it to the head of the queue.

FirstComeFirstServed
A first come, first served scheduling algorithm. It will keep re-presenting the same job each time getNextjob is called, until that job is removed from the queue, either because it has finished, or it gets blocked for I/O.

Lottery
Implements thescheduler class again and contains the algorithm within the getNextJob method that randomly picks a process from the queue after each call of this class.

Job
Class that acts as a job object to store key information about each process such as name, length and prioirty. Conatins the getters and setters for some information.

Simulator
Sets up some information about each process such as the process queue. It contains methods such as getting the amount of cpu ticks and getting a list of finished, incomplete and waiting processes. Also allows for a scheduler to be set for use in the program and also gets the scheduler in use. Can clear the queue of processes ready to run.

SimulatorLoader
Reads in files such as the job files containing lists of processes. It gets each job from the file and loads this into the job queue in an array of job objects. Throws a file not found and IO exception when nessecary.

Simulator Window
The window to be loaded up when starting the program. This is a placeholder for the cpu panel, queue panel and util class items.. Incorperates the menu, simulator and window listeners for actions

performed by the user.

CpuPanel
This class creates the CPU panel displaying information about the descriptions of what scheduler is being used, what file is in use and the tick count of the cpu. Updates every time a job is returned.

QueuePanel
Displays the list of processes in the queue or currently being processes on. Gets the process queue and then lists this in the simulator window. Also is updated every time a job is returned (every tick)

Util
Sets up the grid bag constraints settings for the window, allowing the set up method inside to create a grid style layout to the requirred size as passed in by parameters.

SimulatorApplication
Class containing the main method of the program. Sets up the simulator window, and adds the queue information into it. Also prints out the reuslts summary after each run using the makeResultsSummary() method. Loads in the schedulers, and also sets up any action listeners.