**CSCI 310 Project**

**Group Assignment**

The purpose of this project is to increase your knowledge of the Bankers Algorithm,

Threads and synchronization by solving a problem using Java

**Members:**

Jared Rickert

Toua Vang

Sarbesh Banskota

**Theory:**

**Introduction**

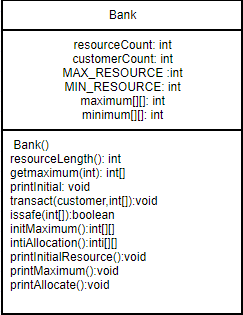
The Banker's algorithm, sometimes referred to as the detection algorithm, is a [resource allocation](https://en.wikipedia.org/wiki/Resource_allocation) and [deadlock](https://en.wikipedia.org/wiki/Deadlock) avoidance [algorithm](https://en.wikipedia.org/wiki/Algorithm) developed by [Edsger Dijkstra](https://en.wikipedia.org/wiki/Edsger_Dijkstra" \o "Edsger Dijkstra) that tests for safety by simulating the allocation of predetermined maximum possible amounts of all [resources](https://en.wikipedia.org/wiki/Resource_(computer_science)), and then makes an "s-state" check to test for possible deadlock conditions for all other pending activities, before deciding whether allocation should be allowed to continue. [Wikipedia]

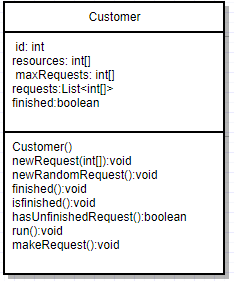
**Usage**

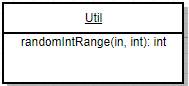
The Project uses multiple threads of execution that simulates the Banker Algorithm. Bank has a set number of resources to be provided to the customers and the Banker Algorithm is used to illustrate the thread management of that system. If a customer request is granted the customer then holds the resources until all of their needs have been met, then the customer returns all returns all resources to the bank. The possibility of Thread lock occurs if resources are granted in such a manner which prevents any customer from fulfilling the needs as the remaining other customers are holding resources that are needed. The algorithm is run by the operating system when a process request resources. It actually prevents deadlock if the accepting request puts the system in an unsafe state.

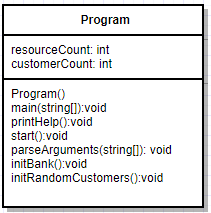
**Data Structures and Organization:**

* Bank Class
* Customer Class
* Util Class
* Program Class









**Functions:**

**Banker Class**

* Resource Length(): Some Information
* getMaximum(): Some Info
* printInitial(): some info
* transact(customer, int[]): some info
* isSafe(int[]): some info
* initMaximum(): some info
* initAllocation():some info
* printInitialResources():some info
* printMaximum(): Some Info
* print Allocated():Some info

**Customer Class**

* newRequest():some info
* newRandomRequest():some info
* finished():some info
* isfinished():some info
* hasUnfinishedRequest(): some info
* run(): some info
* makeRequest(): some info

**Program Class**

* main(): some info
* printHelp(): some info
* start(): some info
* parseArguments(string[]): some info
* initBank(): some info
* initRandomCustomers(): some info

**User Manual:**

The name of the programs are Bank.java, Customer.java, Program.java, util.java. The address of the project is

* Some Address

To compile this program, simply enter:

* Command to compile

The program is now compiled. To run the program enter this command:

* Command to run.

This command will execute the program. A sample of executed programs and tests are shown below:

**Source Code:**

**Test:**