Design Patterns for Overflow

Summary

The goal for this paper is to describe in concrete terms what classes will need to be implemented as well as their general structure to handle cases of integer overflow while performing operations on our mathematical objects. The diagram describes a process whereby the user will input information to be processed, request an operation to be performed on said data, how the program will track the state of the calculation, assess its output, and then output its solution. This paper is divided into two sections describing in detail how each will accomplish its task.

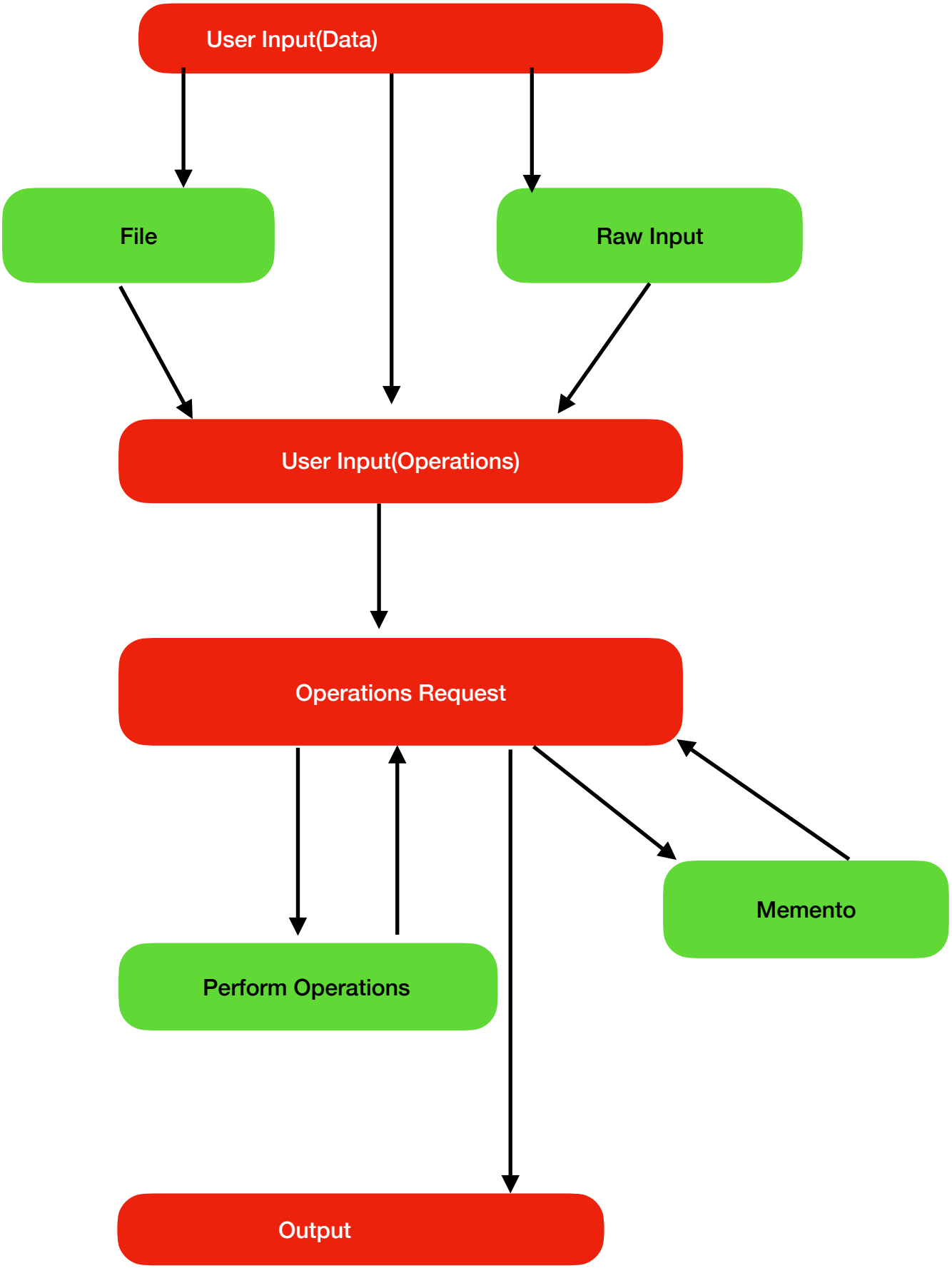
User Input

For the first part of the program the user will either input data to be turned into a matrix. Input is divided into two separate cases. The first is a case where the user provides a file to read from, ideally a spreadsheet. This process should be abstracted, that is, allowing for many different forms of reading from file. For instance, an excel document or a Libre Office one turning each into a matrix. It would be nice to also handle an input case whereby the numbers present in the sheet are already too large for raw data types and must be created initially as a StringMatrix. The program will then take further input from the user whereby the user will select from different options for operations to be performed on the matrix.

Operations Request

Following the choice of an operation an operations request object will be used to instantiate a memento or observer object. The operations request will also have the job of choosing the correct operation to be performed, monitoring the result of the operation for a case of overflow, and requesting a prior state of the matrix when such a case occurs restarting the calculation with the matrix as a different subtype of of itself. When the calculation is complete the operations request will then send the result to output.

If the requested operation resulted in a case of overflow the calculation will return null. In this case the operations request class will then request a memento object, recreate the Matrix object as a StringMatrix and restart the operation. To save time in performing our calculations it would be an excellent feature to update the state of the memento object such that the object is in prior to overflow. While this feature would not be a problem for cases where we are solving systems of equations, it would be one for cases of performing other arithmetic operations and tracking the state of execution in the operation would be required.


User Input(Data)
File
Raw Input
User Input(Operations)
Operations Request
Memento
Perform Operations
Output