**IDS Project 2**

For the ease of operation, the input is automatically set for every run in eclipse by following the steps below:

run-> run configurations->arguments-> give the required values.

**CountMin**

Here we take k number of counter arrays where each array has w number of counters.

We read each flow ID and then hash it to each of the counters to increment value, in the end we check all the arrays and the present the least value from all the k counters for a given array.

We have the following functions:

* Countmin constructor
* Record
* Random number generator
* hashXOR
* query
* getAvgError
* main
  + This method takes in the input and gives the values to the required variable names. We invoke the constructor by creating the object here and then we true to read the output of this from eclipse IDE console to a new file in the system.

**CounterSketch**

This is almost similar to the count min, the only difference is we assign a flag as + or minus based on the first bit of the hashed value and using this we either increment or decrement.

This reduces the relative error by reducing the number of times we increment counter array.

* Countmin constructor
* Record
* Random number generator
* hashXOR
* query
* getAvgError
* main
  + This method takes in the input and gives the values to the required variable names. We invoke the constructor by creating the object here and then we true to read the output of this from eclipse IDE console to a new file in the system.

**ActiveCounter**

For large range of numbers or flow IDS, there is a way to design small number of counters. Active counters mainly aim to reduce the number of counters

* main
  + This method takes in the input and gives the values to the required variable names. We invoke the constructor by creating the object here and then we true to read the output of this from eclipse IDE console to a new file in the system.