Kristopher Sewell
CE2336.002
kjs170430

Method: Main
Return: Void
Arguments: Null
Depends: java.io.File java.util.Scanner java.io.PrintWriter Number Complex
Description:

- if Input file exists and can be opened. Open Scanner While hasNext
- Getnextline, call trim and call split, store in parts array.
- If parts array is greater then 3 continue;
- is part[0] a Number? Yes – Create Number Object
- is part[0] a Complex? Yes – Create Complex Object
- If no to both – Continue;
- is part[2] a Number? Yes – Create Number Object
- is part[2] a Complex? Yes – Create Complex Object
- If no to both – Continue;
- is part[1] a valid operator? Yes – Do calculation and toString Object1 + " " + part[1] + " " + Object2 + " = " + calculation into file.
- Close files, exit.

Method: Number.isValid
Return: Bool
Arguments: String
Depends: java.util.regex
Description:

- pattern is equal "[0-9]+([.][0-9]*){0,1}" 0 to 9 at least one time with '.' 0 to 1 time with [0-9]*
- return string.matches(pattern)

Method Complex.isValid
Return: Bool
Arguments: String
Depends: java.util.regex Number.isValid
Description:

- pattern is equal ValidNumber[+-]{1,1}ValidNumber'i'
- return string.matches(pattern)

Method: checkOp
Return: Bool
Arguments: String
Depends:
Description:

- if string.size == 2 check that its /=
- char array {+,-,*,/,<,>,=} for each check == string[0] return true
- return false by default

Method: calculate
Return: Object
Arguments: Object, Object, String
Depends:
Description:

- 2 * 2 * 8 = 32 ways to calculate
- if number number (1 instanceof Number && 2 instanceof Number)
  - switch case operators
  - return new Number(1.getnum <operator> 2.getnum)
  - **check 2 != 0 for divides overflow/underflow checks
- if number complex
  - switch case operators
  - return new Complex(1.getnum <operator> 2.getnum, 2.getcomplex)
  - will need to calculate magnitude for most operations
- if complex number
  - switch case operators
  - call number complex with modified operators(if needed)
- if complex complex
  - if not (+ or -) calculate magnitude for both
  - switch case operators
- If to return true make Number.number = infinity **include toString case for this
- If to return false make Number.number =  NaN **include toString case for this