Class PeriodicTable

Method

classifieds:

count (int)

elementArray (Element[][])

constructed (boolean)

default constructor

ASSERTION ALGORITHM

count = 0

elementArray = new Element[11][11]

constructed = false

copy constructor

IMPORT inPeriodicTable (inCount, inArray, isConstructed)

ASSERTION ALGORITHM:

count = inCount

elementArray = inElementArray

constructed = isConstructed

Getters

getCount

Export count

ALGORITHM:

Export count

getArray

Export namel

ALGORITHM:

Export elementArray

isConstructed

Export constructed

ALGORITHM:

Export constructed

Setters

setCount

IMPORT inCount (int)

ASSERTION ALGORITHM:

countl = inCount

setArray

IMPORT inArray(Element 2dArray)

ASSERTION ALGORITHM:

elementArray = inArray

isConstructed

IMPORT isConstructed (boolean)

constructed = inConstructed

setMass

IMPORT inMass double

ASSERTION ALGORITHM:

IF mass > 0.00

mass = inMass

Equals

IMPORT inPeriodicTable

ASSERTION ALGORITHM:

If count == getCount

If elementArray == getArray

If constructed == isConstructed isEqual = true

Else

isEqual = false

toString

IMPORT nothing

EXPORT returnString

ASSERTION ALGORITHM:

returnString = elementArray..getSymbol

ENDMETHOD

SUBMODULE addElement

METHOD

PRINT "==============================================” NEWLINE

"Is the New Element a Metal “Yes” "

IF userInput = Yes

isMetal = true

ELSEIF

isMetal = false

ENDIF

PRINT "==============================================” NEWLINE

"Please Enter the Symbol For the Element"

inSymbol = userInput

IF inSymbol length > 2

inSymbol = inSymbol REDUCED to 2

ENDIF

PRINT “==============================================” NEWLINE

"Please Enter the Name for the Element"

inName = userInput

WHILE NOT 0<inAtomic && inAtomic<101

PRINT “=====================================” NEWLINE

"Please Enter the Atomic number for the Element"

TRY

Parse userInput to double

inAtomic = userInput

CATCH

PRINT “==============================” NEWLINE

“Please Enter a Valid Value

ENDTRYCATCH

ENDWHILE

IF isMetal = true

WHILE NOT inConductivity > 0.00

PRINT “=====================================” NEWLINE

“Please Enter the Conductivity for the Element”

TRY

Parse userInput to double

inConductivity = userInput

CATCH

PRINT “==============================” NEWLINE

“Please Enter a Valid Value”

ENDWHILE

ELSEIF

WHILENOT userInput = G || S|| L

PRINT “=====================================” NEWLINE

“Please Enter the state for the Element”

inState = userInput

ENDWHILE  
ENDIF

WHILE NOT userInput > 0.00

PRINT “=====================================” NEWLINE

“Please Enter the Mass of the Element”

TRY

Parse userInput to double

inMass = userInput

CATCH

PRINT “==============================” NEWLINE

“Please Enter a Valid Value”

inMass = -1

ENDWHILE

IF isMetal == true

Element element = new metal(symbol, name, atomicInt, massDouble,conductivityDouble)

elementArray[getCount()/10%0][getCount()] = element

ELSEIF

Element element = new metal(symbol, name, atomicInt, massDouble,stateChar)

elementArray[getCount()/10%0][getCount()] = element

ENDIF

ENDMETHOD

SUBMODULE readFileToArray

Import lines from ReadFile

Export elementArray

METHOD

TRY

TRY

WHILE NOT EOL

processLine (“<”) && (“/>”)

processLine(“,”)

TRY

Send line to Metal

CATCH

Send line to nonMetal

ENDWHILE

CATCH

System out “Failed”

END TRY CATCH

Return elementArray

END METHOD

private void FileOutput()

{

FileOutputStream fileStrm = null;

PrintWriter pw;

fileStrm = new FileOutputStream(File);

pw = new PrintWriter(fileStrm);

for(int i; i < 10; i++)

{

for(int j; j <10; j++)

{

pw.println("<" + elementArray[i][j].getSymbol() + "," + elementArray[i][j].getName() + "," + elementArray[i][j].getAtomic() + "," + elementArray.getMass() + "/>");

}

i++;

}

}