Input Data: Lv

Jaime Díez González-Pardo

Pablo Lavín Pellón

Inés Sánchez de Movellán Sáiz

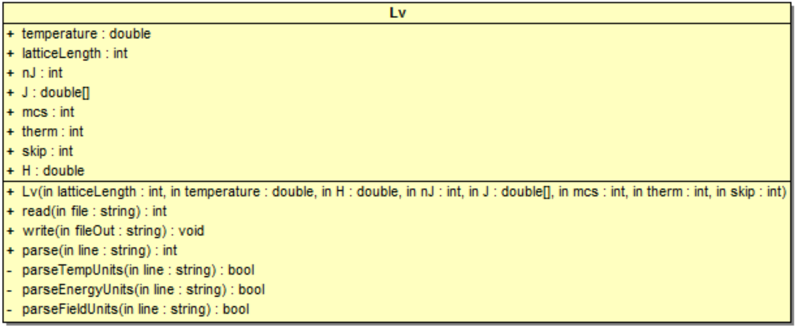
Advanced Computation

January 11, 2019

1. Introduction
2. Code: Lv.java
   - Parse
   - Parse Units
   - Read
   - Write
3. Tests

Introduction

Read the data → parse the information → build an object → write an output file



```
Lv
+ temperature : double
+ latticeLength : int
+ nJ : int
+ J : double[]
+ mcs : int
+ therm : int
+ skip : int
+ H : double
+ Lv(in latticeLength : int, in temperature : double, in H : double, in nJ : int, in J : double[], in mcs : int, in therm : int, in skip : int)
+ read(in file : string) : int
+ write(in fileOut : string) : void
+ parse(in line : string) : int
- parseTempUnits(in line : string) : bool
- parseEnergyUnits(in line : string) : bool
- parseFieldUnits(in line : string) : bool
```

Figure: BOUML of the Lv.class

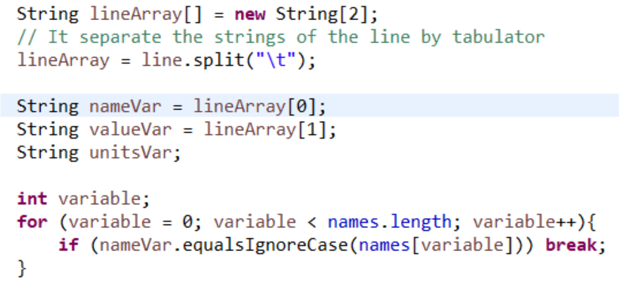
Parse

This method check the variables that program reads and assign to the atributtes.

The Parse received a parameter (String line) which is the line of the file to check, and divided it by the name of variable, the value and units (if the variable has).

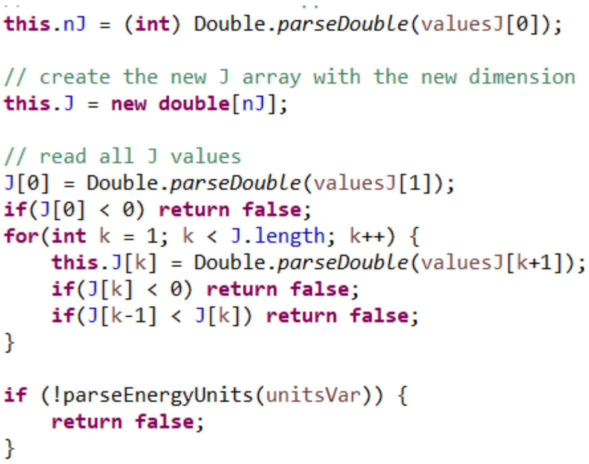
With a swicht-case loop, it assigns the value to the right attributes checking some possible mistakes.

Parse



```java
String lineArray[] = new String[2];
// It separate the strings of the line by tabulator
lineArray = line.split("\t");

String nameVar = lineArray[0];
String valueVar = lineArray[1];
String unitsVar;

int variable;
for (variable = 0; variable < names.length; variable++){
    if (nameVar.equalsIgnoreCase(names[variable])) break;
}
```

Parse



```java
this.nJ = (int) Double.parseDouble(valuesJ[0]);

// create the new J array with the new dimension
this.J = new double[nJ];

// read all J values
J[0] = Double.parseDouble(valuesJ[1]);
if(J[0] < 0) return false;
for(int k = 1; k < J.length; k++) {
    this.J[k] = Double.parseDouble(valuesJ[k+1]);
    if(J[k] < 0) return false;
    if(J[k-1] < J[k]) return false;
}

if (!parseEnergyUnits(unitsVar)) {
    return false;
}
```

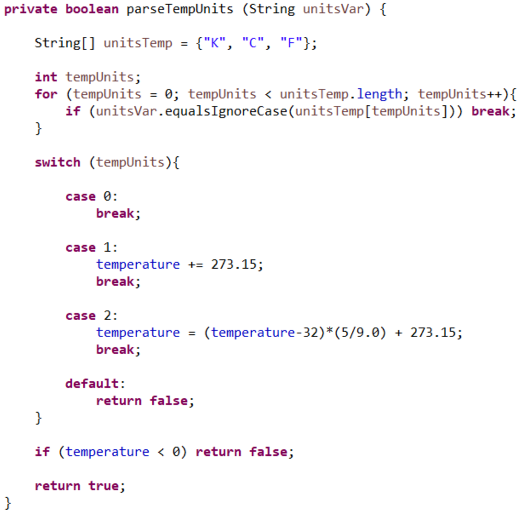
Parse Units

Three methods similar to parse but to modifier the units of the temperature, energy and magnetic field.

With a switch-case loop modified the variables to convert units.

We have consider a few units per variables modifying all to SI.

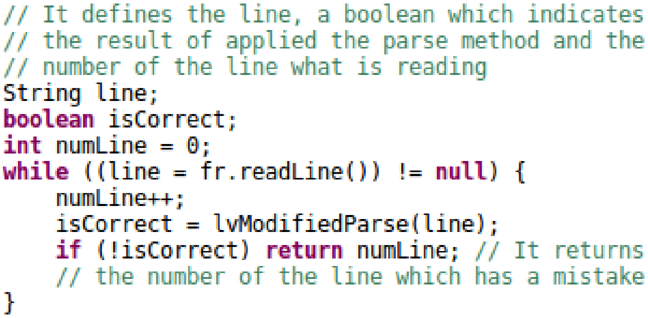
Parse Units



```
private boolean parseTempUnits (String unitsVar) {

    String[] unitsTemp = {"K", "C", "F"};

    int tempUnits;
    for (tempUnits = 0; tempUnits < unitsTemp.length; tempUnits++){
        if (unitsVar.equalsIgnoreCase(unitsTemp[tempUnits])) break;
    }

    switch (tempUnits){

        case 0:
            break;

        case 1:
            temperature += 273.15;
            break;

        case 2:
            temperature = (temperature-32)*(5/9.0) + 273.15;
            break;

        default:
            return false;
    }

    if (temperature < 0) return false;

    return true;
}
```

Read

It fills the atributtes of the InputData object with those read from the file given as argument.

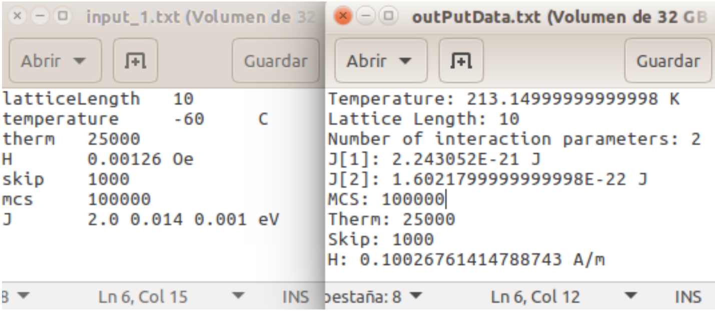


```
// It defines the line, a boolean which indicates
// the result of applied the parse method and the
// number of the line what is reading
String line;
boolean isCorrect;
int numLine = 0;
while ((line = fr.readLine()) != null) {
    numLine++;
    isCorrect = lvModifiedParse(line);
    if (!isCorrect) return numLine; // It returns
    // the number of the line which has a mistake
}
```

Figure: Section of the code used to read a file

Write

It writes the values stored in the InputData object into a file which is given as argument.



```
latticeLength   10
temperature     -60     C
therm   25000
H       0.00126 Oe
skip    1000
mcs     100000
J       2.0 0.014 0.001 eV
```

```
Temperature: 213.14999999999998 K
Lattice Length: 10
Number of interaction parameters: 2
J[1]: 2.243052E-21 J
J[2]: 1.6021799999999998E-22 J
MCS: 100000
Therm: 25000
Skip: 1000
H: 0.10026761414788743 A/m
```

Figure: Left: input file. Right: output file