

Rapport

Jean-emmanuel Chouinard (20246807), Timothe Payette (20239892)

10 juillet 2023

1 Auto-évaluation

Notre programme fonctionne comme prévu. Il respecte les consignes et le format des sorties demandé. Le seul exemple où notre programme ne concorde pas à 100% est l'exemple 6. Nous jugeons cela normal puisque le stock donné et la date courante permet d'utiliser ces médicaments. Voici des images de ces différences

95 Medicament8 23 11 COPPADE	95 Medicament8 23 11 COPPADE
96 Medicament26 4 COPPADE	96 Medicament26 4 COPPADE
97	97
98 PRESCRIPTION 18	98 PRESCRIPTION 18
99 Medicament18 9 OK	99 Medicament18 9 OK
100 Medicament47 38 12 COPPADE	100 Medicament47 38 12 COPPADE
101 Medicament25 38 4 COPPADE	101 Medicament25 38 4 COPPADE
102	102
103 PRESCRIPTION 19	103 PRESCRIPTION 19
104 Medicament11 23 2 COPPADE	104 Medicament11 23 2 COPPADE
105	105
106 APPROV OK	106 APPROV OK
107 STOCK 2016-04-01	107 STOCK 2016-04-01
108 Medicament18 5 2019-10-07	108 Medicament18 5 2019-10-07
109 Medicament11 91 2019-07-01	109 Medicament11 91 2019-07-01
110 Medicament13 18 2019-11-14	110 Medicament13 18 2019-11-14
111 Medicament18 98 2017-10-01	111 Medicament18 98 2017-10-01
112 Medicament2 41 2016-10-05	112 Medicament2 41 2016-10-05
113 Medicament2 4 2016-05-01	113 Medicament2 4 2016-05-01
114 Medicament28 99 2019-02-01	114 Medicament28 99 2019-02-01
115 Medicament13 53 2017-07-01	115 Medicament13 53 2017-07-01
116 Medicament22 57 2016-07-01	116 Medicament22 57 2016-07-01
117 Medicament24 117 2016-07-11	117 Medicament24 117 2016-07-11
118 Medicament24 57 2019-01-01	118 Medicament24 57 2019-01-01
119 Medicament25 4 2017-05-26	119 Medicament25 4 2017-05-26
120 Medicament28 44 2017-05-31	120 Medicament28 44 2017-05-31
121 Medicament28 47 2018-04-01	121 Medicament28 47 2018-04-01
122 Medicament28 88 2016-05-15	122 Medicament28 88 2016-05-15
123 Medicament11 75 2019-08-05	123 Medicament11 75 2019-08-05
124 Medicament32 85 2019-10-12	124 Medicament32 85 2019-10-12
125 Medicament35 81 2017-01-01	125 Medicament35 81 2017-01-01
126 Medicament36 88 2019-06-30	126 Medicament36 88 2019-06-30
127 Medicament37 36 2019-08-01	127 Medicament37 36 2019-08-01
128 Medicament39 95 2018-09-01	128 Medicament39 95 2018-09-01
129 Medicament41 53 2018-05-11	129 Medicament41 53 2018-05-11
130 Medicament48 120 2017-02-28	130 Medicament48 120 2017-02-28
131 Medicament49 95 2017-04-12	131 Medicament49 95 2017-04-12
132 Medicament5 138 2019-01-21	132 Medicament5 138 2019-01-21
133 Medicament7 16 2018-07-17	133 Medicament7 16 2018-07-17

145 Medicament25 304	145 Medicament25 304
146 Medicament27 99	146 Medicament27 99
147 Medicament28 104	147 Medicament28 104
148 Medicament3 48	147 Medicament3 48
149 Medicament38 124	148 Medicament38 124
150 Medicament11 232	149 Medicament11 232
151 Medicament32 118	150 Medicament32 118
152 Medicament33 475	151 Medicament33 475
153 Medicament34 63	152 Medicament34 63
154 Medicament37 208	153 Medicament37 208
155 Medicament39 48	154 Medicament39 48
156 Medicament41 272	155 Medicament41 272
157 Medicament42 96	156 Medicament42 96
158 Medicament46 28	157 Medicament46 28
159 Medicament47 368	158 Medicament47 368
160 Medicament6 66	159 Medicament6 66
161 Medicament7 56	160 Medicament7 56
162 Medicament8 253	161 Medicament8 253
163	162
164 STOCK 2015-04-21	163 STOCK 2015-04-21
165 Medicament18 5 2019-10-07	164 Medicament18 5 2019-10-07
166 Medicament11 91 2019-07-01	165 Medicament11 91 2019-07-01
167 Medicament13 18 2019-11-14	166 Medicament13 18 2019-11-14
168 Medicament18 98 2017-10-01	167 Medicament18 98 2017-10-01
169 Medicament2 41 2016-10-05	168 Medicament2 41 2016-10-05
170 Medicament2 4 2016-05-01	169 Medicament2 4 2016-05-01
171 Medicament28 99 2019-02-01	170 Medicament28 99 2019-02-01
172 Medicament13 53 2017-07-01	171 Medicament13 53 2017-07-01
173 Medicament22 57 2016-07-01	172 Medicament22 57 2016-07-01
174 Medicament24 117 2016-07-11	173 Medicament24 117 2016-07-11
175 Medicament24 57 2019-01-01	174 Medicament24 57 2019-01-01
176 Medicament25 4 2017-05-26	175 Medicament25 4 2017-05-26
177 Medicament28 44 2017-05-31	176 Medicament28 44 2017-05-31
178 Medicament28 47 2018-04-01	177 Medicament28 47 2018-04-01
179 Medicament28 88 2016-05-15	178 Medicament28 88 2016-05-15
180 Medicament11 75 2019-08-05	179 Medicament11 75 2019-08-05
181 Medicament32 85 2019-10-12	180 Medicament32 85 2019-10-12
182 Medicament35 81 2017-01-01	181 Medicament35 81 2017-01-01

2 Analyse de la complexité temporelle (pire cas) théorique en notation grand O

2.1 Légende

Voici la légende que nous allons utiliser tout au long de l'analyse:

- n - indique le nombre de types de médicaments différents en stock
- m - indique le nombre d'items sur la prescription
- k - indique le nombre d'items sur la liste de commande
- l - indique le nombre de médicaments à ajouter au stock

2.2 Prescription

Voici notre algorithme pour la fonction "Prescription"

```
for (String partition : partitions) {
    boolean inStock = false;
    boolean inCommand = false;
    Medicament theMedInStock = new Medicament( nom: "none", nbrPerCycle: 0, nbrDeCycle: 0, currentDate);
    Medicament theMedInCommand = new Medicament( nom: "none", nbrPerCycle: 0, nbrDeCycle: 0, currentDate);
    if (partition.startsWith("PRESCRIPTION")) {
        continue;
    }

    String[] med = partition.split( regex: "\\s+");
    Medicament prescription = new Medicament( med[0], Integer.parseInt(med[1]), Integer.parseInt(med[2]), currentDate);

    for (Map.Entry<String, Medicament> entry : stockMed.entrySet()) {
        if (prescription.getNom().equals(entry.getValue().getNom()) &&
            prescription.getNbrDeMed() <= entry.getValue().getNbrDeMed()) {
            inStock = true;
            theMedInStock = entry.getValue();
            break;
        }
    }

    for (Map.Entry<String, Medicament> entry : commandeMed.entrySet()) {
        if (entry.getValue().getNom().equals(prescription.getNom())) {
            inCommand = true;
            theMedInCommand = entry.getValue();
            break;
        }
    }
}
```

```
if (inStock &&
    prescription.getNbrDeCycle() <= calculateDaysBetweenDates(currentDate, theMedInStock.getDateStr())) {
    theMedInStock.addNbrDeMed(-prescription.getNbrDeMed());
    if (theMedInStock.getNbrDeMed() <= 0) {
        stockMed.remove(theMedInStock.getKey());
    }
    outputText.append(prescription.getNom()).append(" ").append(prescription.getNbrPerCycle()).append(" ").append(prescription.getNbrDeCycle()).append(" ");
} else if (inCommand) {
    theMedInCommand.addNbrDeMed(prescription.getNbrDeMed());
    outputText.append(prescription.getNom()).append(" ").append(prescription.getNbrPerCycle()).append(" ").append(prescription.getNbrDeCycle()).append(" ");
} else {
    commandeMed.put(prescription.getKey(), prescription);
    outputText.append(prescription.getNom()).append(" ").append(prescription.getNbrPerCycle()).append(" ").append(prescription.getNbrDeCycle()).append(" ");
}
}
writeToOutputFile( content: outputText + "\n");
```

On voit que pour chaque item de la prescription ($O(m)$), deux boucles sont faites: une sur les médicaments en stock ($O(n)$) et une sur les médicaments déjà dans la commande ($O(k)$). Notre fonction se réalise donc en $O(m * (n + k))$.

2.3 Approv

Voici notre algorithme pour la fonction "Approv"

```
public void approv(String parts) {
    String modifiedInput = parts.replaceFirst( regex: "\n", replacement: " ");
    String[] partitions = modifiedInput.split( regex: "\n");

    for (String partition: partitions){
        if (partition.startsWith("APPROV :")) {
            continue;
        }
        String[] med = partition.split( regex: "[\t]");
        Medicament medicament = new Medicament(med[0], Integer.parseInt(med[1]),med[2]);

        // Nom + date est déjà dans stockMed
        if (stockMed.containsKey(medicament.getKey())) {
            stockMed.get(medicament.getKey()).addNbrDeMed(medicament.getNbrDeMed());
            continue;
        }
        stockMed.put(medicament.getKey(), medicament);
    }
    writeToOutputFile( content: "APPROV OK\n");
}
```

On voit qu'une boucle passe à travers la liste d'ajout ($O(l)$). De plus, la fonction "contains" de l'arbre est utilisé pour trouver des éléments dans la liste de médicaments en stock ($O(\log n)$). Notre fonction se réalise donc en $O(l * \log(n))$.

2.4 Date

Voici notre algorithme pour la fonction "Date"

```
public void date(String parts){
    String modifiedInput = parts.replace( target: "\n", replacement: "");
    String[] partitions = modifiedInput.split( regex: " ");

    currentDate = partitions[1];

    if (commandeMed.isEmpty()) {
        writeToOutputFile( content: currentDate + " OK\n\n");
    } else {
        writeToOutputFile( content: currentDate + " COMMANDES :\n");
        for (Map.Entry<String, Medicament> entry : commandeMed.entrySet()) {
            Medicament entryValue = entry.getValue();

            writeToOutputFile( content: entryValue.getNom() + " " + entryValue.getNbrDeMed() + "\n");
        }
        commandeMed.clear();
        writeToOutputFile( content: "\n");
    }

    ArrayList<Medicament> toRemove = new ArrayList<>();
    for (Map.Entry<String, Medicament> entry : stockMed.entrySet()) {
        Medicament medicament = entry.getValue();
        if (medicament.getDate().isBefore(LocalDate.parse(currentDate))) {
            toRemove.add(medicament);
        }
    }
    toRemove.forEach(medicament -> stockMed.remove(medicament.getKey()));
}
```

On voit qu'une boucle permet d'imprimer toutes les médicaments en commande ($O(k)$). Ensuite, on vérifie si des médicaments ont expirés et on les supprime de l'arbre ($O(n^2)$). Notre fonction se réalise donc en $O(k * n^2)$.