Travaux pratiques: implémentation d'une plateforme de système répartie sur une infrastructure distribuée.

Etape 1: faire un programme séquentiel non parallélisé qui compte le nombre d'occurrences des mots dans un fichier.

Prérequis pour les questions suivantes:

- **programmer en Java**
  https://www.infres.telecom-paristech.fr/people/bellot/java/
- **manipuler l'environnement de développement Eclipse**
  https://www.infres.telecom-paristech.fr/people/bellot/java/tps/index.html
- **lire et écrire dans un fichier en java**
  https://www.infres.telecom-paristech.fr/~bellot/CoursJava/JavaIO.html
-

1 Premier comptage en séquentiel pur

Implémentez un logiciel en java qui compte le nombre d'occurrences des mots d'un fichier d'entrée de manière non parallélisée (monothread), en utilisant un seul processeur.
Quelle structure de donnée est la plus pertinente pour stocker les résultats: List, Hash ou Set ?

Testez votre programme avec un fichier d'entrée *input.txt* avec comme contenu:

```
Deer Beer River
Car Car River
Deer Car Beer
```

Résultat:

```
Deer 2
Beer 2
River 2
Car 3
```

2 Premier tri en séquentiel pur

Modifiez votre programme pour trier par nombre d'occurrences:

Résultat:

```
Car 3
Deer 2
Beer 2
River 2
```

3 Deuxième tri alphabétique en séquentiel pur
Modifiez le programme pour trier alphabétiquement pour les mots à égalité du nombre d'occurrences:

Résultat:
Car 3
Beer 2
Deer 2
River 2

4 Test du programme séquentiel sur le code forestier de Mayotte
Testez ensuite votre programme avec le code forestier de Mayotte disponible sur github *forestier_mayotte.txt* :
https://github.com/legifrance/Les-codes-en-vigueur
Votre programme a-t-il fonctionné du premier coup ?
Vérifiez en ouvrant le fichier texte qu'il contient bien du texte et non du code HTML.

5 Réparez les erreurs en séquentiel
Réparez toutes les erreurs possibles et ajoutez une étape de "nettoyage/filtrage" du fichier INPUT si nécessaire pour les lignes vides et les caractères spéciaux. Faites attention de bien séparer cette étape de filtrage pour ne l'appliquer qu'avant les calculs. Ceci permet d'enlever facilement cette étape a posteriori: en effet, nous enlèverons cette étape de filtrage/nettoyage plus tard. Veillez donc à respecter au maximum le découplage entre le calcul et le filtrage/nettoyage. De plus, ne passez pas beaucoup de temps pour cette étape car nous l'enlèverons par la suite assez rapidement.

6 Les 50 mots du code forestier de Mayotte
Répondez à la question:
- Quels sont les 50 mots les plus utilisés dans le code forestier de Mayotte ? Les pronoms et mots "bizarres" sont compris - c'est à dire les mots qui ne sont pas présents dans le dictionnaire.

7 On filtre les pronoms et conjonctions
Filtrez les mots tels que "le" "la" "les" "on": les pronoms, conjonctions, liste à compléter vous même ici: http://www.scalpa.info/fr_gram_nature_pronom.php
De même que pour cette question, ne passez pas beaucoup de temps pour le filtrage ici. En effet, cette étape sera supprimée par la suite. Ne vous focalisez pas trop sur le filtrage et le nettoyage qui seront des étapes que nous supprimerons rapidement pour raison d'optimisation.

8 On filtre les mots bizarres
Filtrez les mots bizarres, c'est à dire ceux avec des caractères spéciaux accolés et ceux qui vous paraissent invraisemblables. Encore une fois, ne passez pas trop de temps pour le filtrage/nettoyage qui sera supprimé plus tard.

De nouveau, quels sont les 50 mots les plus utilisés dans le code forestier de Mayotte après filtrage des pronoms et mots bizarres ?

9 Les 50 mots du code de la déontologie de la police nationale

Testez votre programme avec le code de déontologie de la police nationale disponible sur github *deontologie_police_nationale.txt* : https://github.com/legifrance/Les-codes-en-vigueur .

De même que précédemment, filtrez les pronoms, mots bizarres: quels sont les 50 mots les plus utilisés dans le code de déontologie de la police nationale ?

10 Les 50 mots du code du domaine public fluvial

Testez votre programme avec le code du domaine public fluvial *domaine_public_fluvial.txt*.
Filtrez les pronoms et mots bizarres. Quels sont les 50 mots les plus utilisés dans le code du domaine public fluvial ?

11 Les 50 mots du code de la santé publique

Testez votre programme avec le code de la santé publique *sante_publique.txt*.
Filtrez les pronoms et mots bizarres. Quels sont les 50 mots les plus utilisés dans le code de la santé publique ?

12 Chronométrage du programme séquentiel

Chronométrer votre programme sur le code de la santé publique.
Chronométrage possible avec:

```
long startTime = System.currentTimeMillis();
...
long endTime   = System.currentTimeMillis();
long totalTime = endTime - startTime;
```

Combien de temps faut-il pour chacune des étapes:

- Nettoyer/filtrer le fichier d'entrée (cette étape doit être découplée de l'étape suivante)
- Compter le nombre d'occurrences
- Tri (par nombre d'occurrences et alphabétique)
- Temps total (somme des étapes précédentes)

13 Travailler sur des plus gros fichiers

Testez votre programme sur un cas réel: un extrait de toutes les pages internet transformées au format texte brut (format WET). Toutes les pages sur internet au format texte sont disponibles sur http://commoncrawl.org/the-data/get-started/ : chaque mois, environ 3 milliards de pages web, soit 250 To de données sont stockées. Ces données sont disponibles par tranche de moins d'1Go environ, vous travaillerez sur une tranche de 380Mo.
J'ai choisi une tranche en particulier pour avoir une comparaison entre nous (vous pouvez tester sur d'autres tranches si vous voulez). Téléchargez cette tranche ici:
https://commoncrawl.s3.amazonaws.com/crawl-data/CC-MAIN-2017-13/segments/1490218189495.77/wet/CC-MAIN-20170322212949-00140-ip-10-233-31-227.ec2.internal.warc.wet.gz

Décompressez et obtenez le fichier
`CC-MAIN-20170322212949-00140-ip-10-233-31-227.ec2.internal.warc.wet`
Il s'agit d'une tranche contenant un ensemble de sites internet au format texte brut (WET).
Testez votre programme avec ce fichier en entrée. Chronométrez-le.
Constatez que le filtrage ne sert plus car la majorité des mots sont des pronoms en anglais.
Pour optimiser la vitesse, enlevez l'étape de filtrage/nettoyage. Chronométrez de nouveau.

Etape 2: faire un programme parallélisé (avec plusieurs threads) qui compte le nombre d'occurrences des mots dans un fichier en utilisant plusieurs processeurs et plusieurs coeurs de calcul sur une seule machine.

Prérequis et documentation pour les questions suivantes :

- **connaître les architectures matérielles multiprocesseur, multicoeurs**
  https://fr.wikipedia.org/wiki/Microprocesseur_multi-c%C5%93ur
- **savoir différencier un processus lourd et un processus léger**
  https://fr.wikipedia.org/wiki/Thread_(informatique)
  https://fr.wiktionary.org/wiki/processus_l%C3%A9ger
  https://fr.wikipedia.org/wiki/Processus_(informatique)
  https://fr.wiktionary.org/wiki/processus_lourd
- **paralléliser un code en Java en utilisant les Thread:**
  https://docs.oracle.com/javase/tutorial/essential/concurrency/runthread.html

14 Nombre de processeurs.
Combien votre ordinateur a-t-il de processeurs ? Quelle commande faut-il utiliser pour connaître ce nombre de processeurs ?

15 Nombre de coeurs de calcul.
Combien votre ordinateur a-t-il de coeurs de calculs ? Quelle commande faut-il utiliser pour connaître ce nombre de coeurs de calculs ?

16 Parallélisation du calcul.
Modifiez votre programme pour qu'il utilise tous les coeurs de calcul de votre ordinateur. Il faut pour cela créer **autant de processus léger(threads) que de coeurs de calcul**. Il faut également découper le calcul en autant de calculs parallèles que de threads.
Quelle structure de donnée partagée entre tous les threads et résistante à la modification en parallèle de tous ces threads est la plus pertinente pour stocker les résultats: HashTable, HashMap, TreeMap, ConcurrentHashMap, SynchronizedMap ?
Attention de bien vérifier le bon fonctionnement de votre programme en comparant très minutieusement les sorties avec les versions de vos programmes en séquentiel non parallèle.

17 Chronométrage du programme parallèle
Chronométrer votre programme parallèle sur le code de la santé publique.
Votre programme parallèle est-il plus rapide que la version séquentielle non parallèle ?
Même question pour le fichier WET de 380Mo.

Etape 3: travailler avec plusieurs ordinateurs en réseau.

Prérequis et documentation pour les questions suivantes :

- **avoir un compte et pouvoir se connecter aux machines de l'école (contacter la DSI de l'école dans le cas contraire)**
- **lancer un interpréteur de commande (console linux , shell) pour taper des commandes:**
  https://fr.wikipedia.org/wiki/Interpréteur_de_commandes
  https://fr.wikipedia.org/wiki/Shell_Unix
- **connaître quelques commandes de base sous Linux**
  https://fr.wikipedia.org/wiki/Commandes_Unix
- **savoir qu'un ordinateur a un nom d'hôte (hostname) et plusieurs adresses IP:**
  https://en.wikipedia.org/wiki/Hostname
  https://fr.wikipedia.org/wiki/Adresse_IP
- **savoir qu'un ordinateur peut faire partie d'un domaine (comme à l'école, le domaine enst.fr ou le domaine telecom-paristech.fr)**
  https://fr.wikipedia.org/wiki/Nom_de_domaine
- **savoir qu'un nom peut être transformé en adresse IP (et inversement) par un serveur qui gère le système du nom de domaine (DNS, Domain Name System)**
  https://fr.wikipedia.org/wiki/Domain_Name_System
- **se connecter à distance à un ordinateur avec SSH en ligne de commande**
  https://fr.wikipedia.org/wiki/Secure_Shell
  http://www.commentcamarche.net/faq/74-se-connecter-a-distance-avec-ssh-linux

18 Nom court, nom long
quel est le nom COURT de votre ordinateur (le nom simple sans le domaine) ? quel est le nom LONG de votre ordinateur (le nom avec le domaine) ? Comment les connaître en ligne de commande ? Sur les ordinateurs de l'école, est-il possible d'obtenir ces noms autrement qu'en ligne de commande ?

19 Adresse ip
Comment connaître les adresses (plusieurs) IP de votre ordinateur en ligne de commande ? Autrement (en passant par un site internet par exemple) ?

20 Du nom vers l'IP
Comment à partir du nom d'un ordinateur, obtenir les adresses IP en ligne de commande ?

21 De l'IP vers le nom
Comment, à partir d'une adresse IP, obtenir les noms associés en ligne de commande ?

22 Ping pong à l'intérieur!
Testez la communication avec d'autres ordinateurs (pas le vôtre) depuis le réseau de l'école en utilisant la commande ping (pour arrêter le ping faire CTRL + C). suivi du nom court, du nom long, de l'IP. Les trois méthodes fonctionnent-elles ?

23 Ping pong à l'extérieur
Si vous effectuez le ping depuis un réseau différent, il est possible que celui ne fonctionne pas (filtrage des accès vers le réseau de l'école depuis un réseau extérieur), contactez la DSI pour mettre en place une connection VPN / OpenVPN afin d'être sur le même réseau que les machines en salle de TP.

24 Calculer en ligne de commande sur l'ordinateur local
Comment lancer un calcul en ligne de commande sur votre ordinateur (par exemple 2 + 3) ? Parmi les multiples réponses possibles, lesquelles permettent de lancer le calcul et d'obtenir le résultat en appuyant une seule fois sur la touche <Entrée> ?

25 Calculer en ligne de commande sur un ordinateur distant
Comment lancer un calcul (par exemple 2 + 3) en ligne de commande sur un autre ordinateur (à distance) ? Il faudra certainement vous authentifier avec un mot de passe. Comment obtenir le résultat du calcul immédiatement après avoir tapé son mot de passe ?

26 Calculer à distance sans mot de passe
Comment lancer un calcul à distance en utilisant SSH sans taper le mot de passe et en une seule ligne de commande (c'est à dire qu'on appuie sur <Entrée> et on a le résultat directement)?

Etape 4: travailler avec des fichiers locaux ou sur un serveur NFS.

Prérequis et documentation pour les questions suivantes :

- **pouvoir transférer un fichier d'un ordinateur à un autre en utilisant la commande SCP (Secure Copy): https://fr.wikipedia.org/wiki/Secure_copy**
- **connaître l'architecture d'un système de fichier NFS (Network File System) : https://fr.wikipedia.org/wiki/Network_File_System**

27 Chemin absolu
Quel est le chemin absolu de votre répertoire personnel, votre *home directory* ?
(commandes "`cd`" puis "`pwd`")

28 Un fichier dans le répertoire personnel
Créez un fichier fperso.txt contenant le texte "bonjour" dans votre répertoire personnel (sur un ordinateur de l'école).
Vérifiez le contenu du fichier avec cette commande exactement:
`cat ~/fperso.txt`

29 Ou se trouve le fichier dans le répertoire personnel
Ce fichier est-il sur le disque dur de l'ordinateur ou autre part ? Comment savoir où est stocké physiquement ce fichier, à l'aide de quelle commande ?

30 Un dossier et un fichier dans le répertoire temporaire
Créez un dossier /tmp/<votre nom d'utilisateur> en remplaçant <votre nom d'utilisateur>.
Créez un fichier ftemp.txt dans le répertoire /tmp/<votre nom d'utilisateur> .
Vérifiez le contenu du fichier avec cette commande exactement:
`cat /tmp/<votre nom d'utilisateur>/ftemp.txt`
Ce dossier et ce fichier sont-ils sur le disque dur de l'ordinateur ou autre part ? Comment savoir où sont stockés physiquement ces éléments, à l'aide de quelle commande ?

31 Trois ordinateurs A B C. On commence avec A. Utilisation du serveur NFS.
Pour les questions suivantes, utilisez trois ordinateurs: A, B C.
Connectez vous physiquement (avec un clavier, une souris et un écran) sur l'ordinateur A.
Sur A, créez un fichier text.txt contenant le texte "mon texte sur NFS" dans votre répertoire personnel.
Vérifiez que le fichier existe et vérifiez son contenu. Pour cela, sur A, utilisez la commande :
`cat ~/text.txt`

32 Trois ordinateurs A B C. On continue sur B et sur C. Utilisation du serveur NFS.
Connectez-vous à B (physiquement ou à distance) et vérifiez que le fichier text.txt est également présent dans votre répertoire personnel. Pour cela, sur B, utilisez la commande :
`cat ~/text.txt`
De même, connectez-vous à C et vérifiez que text.txt est aussi présent.
Remarquez que vous n'avez pas copié le fichier mais qu'il est présent sur A, B et C grâce au serveur NFS.

33 Trois ordinateurs A B C. On commence avec A. Utilisation des disques locaux.
Déconnectez vous de B et de C et revenez sur l'ordinateur A.
Sur A, créez un dossier /tmp/<votre nom d'utilisateur> et un fichier local.txt contenant le texte "mon texte sur disque local" dans ce dossier /tmp/<votre nom d'utilisateur>.
Vérifiez que le fichier existe et vérifiez son contenu. Pour cela, sur A, utilisez la commande :
`cat /tmp/<votre nom d'utilisateur>/local.txt`

34 Trois ordinateurs A B C. On continue sur B et sur C. Utilisation des disques locaux.
Connectez-vous à B et C (physiquement ou à distance) et vérifiez que le dossier <votre nom d'utilisateur> ainsi que le fichier local.txt **ne sont pas** présent dans /tmp . Pour cela vérifiez avec la commande:
`ls /tmp`

35 Depuis A, copier de A vers B avec les disques locaux.
Comment, **à partir de A**, transférer le fichier /tmp/local.txt **sur B** (dans `/tmp/<votre nom d'utilisateur>/local.txt`) en utilisant scp ? Vérifiez que le fichier est bien présent sur B. Attention: si vous avez une erreur "no such file or directory" (ou l'équivalent français),

vous devez d'abord créer le répertoire `/tmp/<votre nom d'utilisateur>/` avec la commande `mkdir -p` associée à un ssh pour l'ordinateur distant.

36 Depuis A, copier de B vers C avec les disques locaux.
Comment, **à partir de A**, transférer le fichier **de B** (depuis `/tmp/<votre nom d'utilisateur>/local.txt`) **vers C** (dans `/tmp/<votre nom d'utilisateur>/local.txt`) ? Vérifiez que le fichier est bien présent sur C. De même que la question précédentes, vous devez créer les répertoires `/tmp/<votre nom d'utilisateur>/` correspondants.

Etape 5: lancer des programmes java à distance manuellement.

Prérequis et documentation pour les questions suivantes :

- **exporter un .JAR exécutable (Runnable JAR File)**
- **utiliser les packages et savoir lancer un programme Java en ligne de commande**

37 Un premier programme SLAVE sous Eclipse
Faire un programme Java nommé "SLAVE" qui calcule 3+5, affiche le résultat, sous Eclipse (Pour lancer Eclipse: Menu applications>développement), lancer ce programme dans Eclipse (flèche verte "exécuter")

38 Exportation en JAR
Exporter le programme Java en slave.jar exécutable (Java ARchive dite Runnable) sous Eclipse. Attention de bien vérifier que le JAR est de type "Runnable"/"exécutable".

39 Exécution sur disque dur local
Créez le répertoire `/tmp/<votre nom d'utilisateur>/`
Copiez slave.jar exécutable dans le répertoire `/tmp/<votre nom d'utilisateur>/`
Testez et Lancer le slave.jar en ligne de commande sur votre ordinateur local.

40 Copie du JAR et exécution distante
Depuis la machine A contenant `/tmp/<votre nom d'utilisateur>/slave.jar`
Créez à distance sur la machine B (s'il n'existe pas) un répertoire /tmp/`<votre nom d'utilisateur>/`
Copiez slave.jar sur la machine B dans le répertoire /tmp/`<votre nom d'utilisateur>/`
Exécutez à distance (depuis A sur la machine B) le slave.jar.

Etape 6: lancer des programmes en ligne de commande depuis java et afficher la sortie standard et la sortie d'erreur.

Prérequis et documentation pour les questions suivantes :

- utiliser le processBuilder en java **http://docs.oracle.com/javase/7/docs/api/java/lang/ProcessBuilder.html .**

- lancer un exécutable (ou une commande) en ligne de commande depuis un programme écrit en java
- connaître les sorties standard et les sorties d'erreurs

41 Un programme MASTER java qui lance un autre programme en ligne de commande!
Ecrire un programme java nommé "MASTER" qui lance la commande suivante en utilisant ProcessBuilder:
`ls -al /tmp`
Récupérer et afficher la sortie de cette commande.
Vous devez utiliser ProcessBuilder de cette façon:
`ProcessBuilder pb = new ProcessBuilder("ls", "-al", "/tmp");`

42 Un programme MASTER java qui gère les erreurs de lancement d'un autre programme en ligne de commande.
Modifiez votre programme"MASTER" pour qu'il affiche la sortie d'erreur en cas d'erreur lors de l'exécution de la commande. Testez la sortie d'erreur avec une commande qui échoue avec un sortie d'erreur. Essayez par exemple d'exécuter la commande "ls /jesuisunhero".
Explications: si on tape la commande "ls /jesuiunhero", le dossier /jesuisunhero n'existant pas, on aura une erreur de type "impossible d'accéder à /jesuisunhero: aucun fichier ou dossier de ce type." qui s'affiche dans la sortie d'erreur. En effet, il y a deux sorties: les sorties standards (sans erreur) et les sorties d'erreurs.
Vous devez utiliser ProcessBuilder de cette façon:
`ProcessBuilder pb = new ProcessBuilder("ls", "/jesuisunhero");`
Sur `pb`, vous pouvez récupérer le flux de la sortie standard et le flux de la sortie d'erreur.
Attention de ne pas rester bloqué en attente d'une sortie standard (sans erreur)! En effet, rien ne sera écrit dans la sortie standard.
Ici il n'y aura qu'une sortie dans la sortie d'erreur! Une solution consiste à utiliser un thread qui s'occupe de la sortie standard et un autre thread qui s'occupe de la sortie d'erreur.

43 Un programme MASTER java qui lance un slave.jar en ligne de commande.
Modifiez votre programme "MASTER" pour qu'il lance "SLAVE" slave.jar situé sur la même machine que "MASTER" dans le dossier
`/tmp/<votre nom d'utilisateur>/slave.jar`
(voir la question correspondante pour slave.jar).

Etape 7: gérer les timeout du MASTER.

44 Un SLAVE qui simule un calcul de 10 secondes.
Modifiez votre programme SLAVE pour qu'il simule une attente de 10 secondes avant d'afficher le résultat du calcul 3+5. Pour cela utilisez
`Thread.sleep(10000);`
Vérifiez le bon fonctionnement du SLAVE et constatez qu'il y a 10 secondes entre le démarrage du SLAVE et l'affichage du résultat. Attention de ne rien afficher avant les 10 secondes pour que la question suivante fonctionne correctement.

Générez de nouveau le slave.jar. Copiez-le en écrasant le slave.jar dans le dossier `/tmp/<votre nom d'utilisateur>/slave.jar`
Testez le lancement à partir de MASTER (le même que à cette question).

45 Gérer les timeout au niveau du MASTER.
Modifier le MASTER pour qu'il attende que quelque chose soit écrit dans la sortie standard (sans erreur) ou dans la sortie d'erreurs du SLAVE pendant un certain temps maximum. Au bout du temps imparti le MASTER considère un timeout.
Il arrête les threads s'occupant des sorties et stoppe le processus créé avec ProcessBuilder.
TEST1 : Testez le bon fonctionnement du timeout en lançant le SLAVE avec un timeout de 2 secondes sur les sorties (standard et d'erreur). Le timeout étant plus court (au niveau du MASTER 2 secondes) que le temps de calcul du SLAVE (10 secondes), le MASTER doit arrêter les threads et le processus.
TEST 2: Testez ensuite avec un timeout de 15 secondes, il ne devrait pas y avoir de timeout.
TEST 3: Testez ensuite en changeant le SLAVE pour qu'il écrive non plus dans la sortie standard (sans erreur) mais dans la sortie d'erreur. Pour cela, remplacez dans le Slave les *System.out.print*… par *System.err.print*...
Aide:
Une solution pour gérer les timeout consiste à utiliser une structure *ArrayBlockingQueue* donnée en paramètre de chaque thread s'occupant de lire la sortie standard (sans erreur) ou la sortie d'erreur. Chaque thread va écrire ce qui est lu depuis les sorties dans cette structure en utilisant la méthode *put*. Le timeout de 2 secondes peut alors être paramétré lors de la récupération des éléments insérés dans la structure en utilisant la méthode *poll* sur l'*ArrayBlockingQueue* de cette manière:
poll(2, TimeUnit.SECONDS);

Etape 8: déployer automatiquement le programme SLAVE sur un ensemble de machines.

46 Un programme DEPLOY : Test de connection SSH multiple
Créer un fichier texte à la main contenant : les adresses IP et/ou les noms des machines que nous voulons utiliser pour notre système réparti (par exemple toutes les machines de cette salle de TP), avec un nom ou une IP par ligne dans le fichier.
Créer un nouveau programme java DEPLOY qui lit ce fichier ligne par ligne et teste si la connection SSH fonctionne bien sur chacune des machines.
Pour vérifier que la connection fonctionne bien, vous pouvez faire afficher le nom de la machine distante (en exécutant la commande hostname à distance) et vérifier que l'affichage a effectivement lieu, sans erreurs. Réutilisez des parties de codes de la cinquième étape.

47 Un programme DEPLOY : copie de slave.jar multiple
Modifier votre programme DEPLOY pour qu'il copie le slave.jar dans `/tmp/<votre nom d'utilisateur>/` sur les ordinateurs dont la connection SSH fonctionne.

Pour cela, vous devez utiliser la commande `mkdir -p` pour créer les répertoires dans /tmp s'ils n'existent pas déjà.
Vérifiez ensuite manuellement que le fichier a bien été copié sur les ordinateurs distants.
Lors des copies, faites attention au caractère " / " à la fin d'un chemin :
/tmp/toto est un chemin vers un fichier nommé toto
/tmp/toto/ est un chemin vers un dossier nommé toto.

Etape 9: MapReduce - SPLIT et MAP

Prérequis et documentation pour les questions suivantes :

- **connaître l'architecture de MapReduce**

48 Un MASTER qui déploie les splits

Créez trois fichiers correspondants à des "splits" dans le répertoire temporaire
`/tmp/<votre nom d'utilisateur>/splits`
S0.txt S1.txt S2.txt.
S0.txt contient:
`Deer Beer River`
S1.txt contient:
`Car Car River`
S2.txt contient:
`Deer Car Beer`

Modifiez votre MASTER pour qu'il copie les fichiers de splits dans 3 ordinateurs différents.
Attention, le répertoire `/tmp/<votre nom d'utilisateur>/splits` doit être créé sur les 3 ordinateurs s'il n'existe pas. Cette création peut se faire de manière automatique.
De la même manière que le programme DEPLOY, le MASTER va copier ces splits vers 3 ordinateurs dont la connection SSH fonctionne.

49 Un SLAVE qui fait la phase de map

Modifiez le SLAVE pour qu'il calcul un map à partir d'un split.
Pour cela, il prend un nom de fichier "Sx.txt" en entrée depuis le dossier `splits` et calcule un fichier "UMx.txt" en sortie dans le dossier `maps`, avec x variant (ici de 1 à 3). Le nom du fichier sera donné comme argument args du main:
public static void main (String[] args)

Testez dans un terminal le slave.jar comme suit:

```
cd /tmp/<votre nom d'utilisateur>/
java -jar slave.jar /tmp/<votre nom d'utilisateur>/splits/S0.txt
```

Le fichier `/tmp/<votre nom d'utilisateur>/maps/UM0.txt` doit être créé contenant
Dear 1
Beer 1
River 1

Testez le fonctionnement de votre SLAVE avec le fichier S1.txt contenant
Car Car River
Testez dans un terminal le JAR comme suit:

```
cd tmp/<votre nom d'utilisateur>/
java -jar slave.jar /tmp/<votre nom d'utilisateur>/splits/S1.txt
```

Le fichier `/tmp/<votre nom d'utilisateur>/maps/UM1.txt` doit être créé contenant
Car 1
Car 1
River 1

50 Un MASTER qui lance les SLAVE pour la phase de map.
Modifiez le MASTER pour qu'il fasse la partie 2 de l'algorithme principal du master. Affichez le dictionnaire "UMx - machines". Attention, n'affichez que le nom du fichier UMx et son emplacement. Le contenu des UMx reste sur les slaves et n'est pas envoyé au MASTER.
Exemple d'affichage:
UM1 - c129-02
UM2 - c129-06
etc ...

51 Un SLAVE qui affiche les clés en sortie
Modifiez le SLAVE pour qu'il affiche uniquement les clés sur la sortie standard lors du calcul Sx -> UMx (ne modifiez pas la création du fichier UMx). Testez le fonctionnement de votre SLAVE avec un fichier S1.txt contenant

```
Car Car River
```

Testez dans un terminal le JAR comme suit:

```
cd tmp/<votre nom d'utilisateur>/
java -jar slave.jar /tmp/<votre nom d'utilisateur>/splits/S1.txt
```

L'affichage doit être
Car
River

Et le fichier UM1 doit toujours être créé.

52 Un MASTER qui affiche les clés
Modifiez le MASTER pour qu'il fasse la partie 3 de l'algorithme principal du master. Pour recevoir les clés, vous utiliserez l'affichage dans la sortie standard du SLAVE (récupéré grâce à l'inputstream du process). Testez en affichant le dictionnaire "clés - listes des UMx".
Exemple d'une entrée dans le dictionnaire:
clé Car - liste d'UMs: <UM1, UM2>
(pour une clé, on a plusieurs UMx)

53 Un MASTER attend la fin de la phase de map.
Modifiez le MASTER pour qu'il fasse la partie 4 de l'algorithme principal du master. Testez avec un affichage final "phase de MAP terminée" , cet affichage doit s'effectuer uniquement quand tout est fini pour la phase de map (création des fichiers UMx et création complète des dictionnaires "UMx-machines" et "clés-UMx").

Etape 10: MapReduce - SHUFFLE et REDUCE

54 Un SLAVE qui effectue le reduce (partie 1)
Modifiez le SLAVE pour qu'il prenne plusieurs fichiers UMx en entrée et calcule un fichier SMx en sortie. Attention, votre SLAVE doit toujours être compatible avec le calcul de Sx -> UMx. Le mode de fonctionnement doit donc être donné en argument args du main: mode 0 pour "calcul de Sx -> UMx" ou mode 1 pour "calcul de UMx -> SMx".
En fonction du mode, le SLAVE sait s'il reçoit un Sx ou des UMx comme arguments (args) du main.
De plus, la clé et le nom du fichier de sortie SMx doivent également être passés en arguments du main.

Testez le fonctionnement de votre SLAVE SHAVADOOP avec deux fichiers UM1.txt et UM2.txt
UM1.txt contenant:

```
Car 1
Car 1
River 1
```

UM2.txt contenant:

```
Dear 1
Car 1
Bear 1
```

Testez dans un terminal le JAR comme suit, l'argument " 1 " correspond au mode de fonctionnement du slave, ici le mode " 1 " est celui qui calcule un SMx en output à partir de plusieurs UMx en input. N'oubliez pas d'ajouter le mode "0" par ailleurs.

```
cd tmp/<votre nom d'utilisateur>/
java -jar slave.jar 1 car /tmp/<votre nom
d'utilisateur>/maps/UM1.txt /tmp/<votre nom
d'utilisateur>/maps/UM2.txt
```

Le fichier `/tmp/<votre nom d'utilisateur>/maps/SM1.txt` doit être créé avec comme contenu:

```
Car 1
Car 1
Car 1
```

55 Un SLAVE qui effectue le reduce (partie 2)
Modifiez le SLAVE pour qu'il calcule un RMx juste après le calcul de SMx.
Testez dans un terminal le JAR comme suit:

```
cd tmp/<votre nom d'utilisateur>/
java -jar slave.jar 1 car /tmp/<votre nom
d'utilisateur>/maps/UM1.txt /tmp/<votre nom
d'utilisateur>/maps/UM2.txt
```

Le fichier `/tmp/<votre nom d'utilisateur>/reduces/RM1.txt` doit être créé contenant
`Car 3`

Attention: on travaille maintenant dans le dossier "reduces"

56 Un MASTER qui lance le shuffle
Ouvrez le document de l'architecture pour bien comprendre cette question.
Modifiez le MASTER pour qu'il copie les UMx au bon endroit pour la phase de reduce. Pour cela, il doit utiliser les deux dictionnaires "UMx-machines" et "clés-UMx". Pour chaque clé, on a plusieurs UMx. Ces UMx peuvent être sur des machines différentes. Il faut rapatrier ces UMx sur une seule machine "slave" qui va calculer le reduce.
Modifiez le MASTER pour qu'il copie, pour chaque clé, les UMx des différentes machines sur une seule.

57 Un MASTER qui lance le reduce
Modifiez le MASTER pour qu'il fasse la partie 5 de l'algorithme principal du master.
Vous pouvez tester en affichant temporairement le dictionnaire "RMx - machines".

58 Un MASTER qui affiche le résultat final
Modifiez le MASTER pour qu'il fasse les parties 6,7 et 8 de l'algorithme principal du master.

59 Un projet Hadoop-Mapreduce complet
Modifiez votre projet pour qu'il fonctionne avec des plus gros fichiers en input.