Projet BPO
Un autre algorithme de recherche, un autre jeu

Le troisième objectif est d'enrichir l'application existante avec un autre algorithme de recherche de chemin et un autre jeu. C'est également l'occasion d'enrichir vos connaissances en lisant quelques pages de documentation. Celles-ci font partie intégrante du cours et sont donc des prérequis à l'examen.

Remarques préliminaires :

- *Un forum est ouvert sur Arche pour répondre aux questions. Évitez les questions individuelles par mail, de sorte que chacun puisse profiter des questions et des réponses.*

- *Il y a deux dépôts distincts à faire sur Arche avant le **vendredi 11 janvier 2019 - 20h***
  - *répertoire **`projetBPO`** zippé contenant les fichiers sources*
  - *archive **`projetBPO.jar`** exécutable, déposé dans l'atelier*

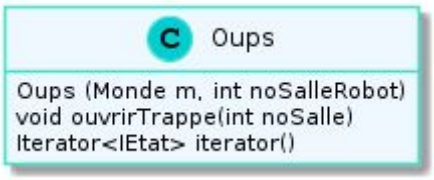
1. Écrire la classe **`LargeurDAbord`**. Vous pouvez vous aider de la page suivante :

   http://fr.wikipedia.org/wiki/Algorithme_de_parcours_en_largeur

2. Comparer les temps d'exécution des deux algorithmes sur le même jeu. Vous pouvez vous appuyer sur la fonction ***`nanoTime`*** de la classe **`System`.**
   http://docs.oracle.com/javase/1.5.0/docs/api/java/lang/System.html#nanoTime%28%29

3. On intègre un nouveau jeu dans l'application : le labyrinthe **oups**, vu en TD. Écrire le jeu **Oups**.

   Merci de respecter quelques contraintes sur les classes **Oups** et **Monde**, vous pouvez bien évidemment ajouter d'autres fonctions :



4. Comparer les temps d'exécution des deux algorithmes sur les deux jeux.

5. Contraintes supplémentaires à respecter :

    - création d'un sous répertoire **projetBPO.jeux.jeudemots** pour y placer les classes **JeuDeMots** et **Dictionnaire**

    - création d'un sous-répertoire **projetBPO.jeux.oups** pour y placer toutes les classes relatives au jeu **Oups**

6. Apprendre à faire une archive jar exécutable (avec intelliJ :

    File > Project Structure … > Projects Settings > Artifacts

    \+ **JAR > From modules with dependencies …**

    - choix de la "**Main class**" : reprendre votre classe contenant la fonction main qui instancie un jeu et un algorithme de recherche...
        - **extract to the target jar** : pour faire une archive autonome
        - *copy to the output directory and link via manifest : pour extraire les librairies et les placer à côté de votre archive*

    gardez le premier choix sélectionné

    Build > Build Artifacts… puis sélectionnez "build" ou "rebuild"

    chemin de l'archive construite :

    projetBPO/out/artifacts/projetBPO_jar/projetBPO.jar

    Tester l'archive avec la commande en ligne :

    java -jar projetBPO.jar

7. Pour vérifier la cohérence de ce que vous avez écrit par rapport aux contraintes fixées, nous mettons à votre disposition une classe cliente sur Arche.

    Copier la classe **client.ClientBPO** dans l'un de vos répertoires, à l'extérieur du projet. Cette classe dispose d'une fonction ***main()*** qui attend deux arguments : **-j** ou **-o** correspondant au choix du jeu (jeu de mots ou oups), puis **-l** ou **-p** correspondant au choix de l'algorithme (largeur ou profondeur).

    Demander l'exécution de cette classe par une commande en ligne dans un terminal, par exemple :

    - **java -classpath projetBPO.jar:. client.ClientBPO -j -p**
    - **java -classpath projetBPO.jar:. client.ClientBPO -o -l**