Sujet 4:

Compilation.

(durée max: 1h)

Jusqu'à présent le script de compilation vient compiler chaque fichier l'un derrière l'autre de manière systématique. Nous avons pris soin de définir l'ordre dans lequel chaque compilation doit avoir lieu, et nous compilons chaque fichier même si celui-ci n'a pas été modifié.
Ceci entraîne un temps de compilation allongé non compatible avec des projets volumineux.

Nous allons utiliser l'outil **`make`** pour simplifier et enrichir la gestion de ce processus de compilation.

`Make` est un outil standard permettant d'automatiser et d'enchaîner des taches de manière simple.

Note: Make est totalement **indépendant** du compilateur `gcc`, et peut être utilisé pour réaliser d'autres tâches que la compilation.

L'appel à `make` en ligne de commande vient lire un fichier du nom de **Makefile**.
La syntaxe associée dans le fichier Makefile suit le modèle suivant:

```
fichier_a_realiser : dependances
     ligne(s) de commandes a executer
```

Explication:
fichier_a_realiser = Le fichier que l'on cherche à créer (ou nom de l'action à réaliser).
dépendances = Le ou les fichiers qui doivent être présents afin de créer ce fichier
ligne(s) de commandes à exécuter = La commande ou les commandes qui doivent être lancées afin de réaliser ce fichier. Chaque ligne est précédée d'une **tabulation** (et non d'espaces). Dans le cas d'actions standards (compilation par ex.) cette ligne peut être inférée automatiquement par **make** et devient alors optionnelle.

- ➔ **Réalisez** les tutoriaux d'entrainements sur le Makefile du niveau minimal.
- ➔ **Ecrivez** le Makefile correspondant au projet limitant l'écriture de lignes pouvant être inférée par make. Observez que le Makefile est plus court que le script de compilation originel.

Lecture de code.

(durée max: 1h)

- ➔ Ecrire la documentation des fichiers `parseur_mode_interactif` et `mode_interactif` afin de vous habituer à lire du code existant comportant des appels systèmes.

Poussée.

(durée max: 2h)

- ➔ **Implémentez** le code des fonctions `poussee_etre_valide` et `poussee_realiser` à partir de l'algorithme que vous avez définit. N'oubliez pas d'ajouter vos fichiers `poussee` a votre `Makefile`.
- ➔ Insérez les appels de `poussee` dans la gestion du déplacement et de l'insertion des nouvelles pièces dans `plateau_deplacement`.
- ➔ Vérifiez le bon fonctionnement de votre code de bout en bout à partir du mode interactif.

Travail en autonomie

(durée estimée: 3h):

- ➔ Fin de réalisation de votre code de `poussee`.