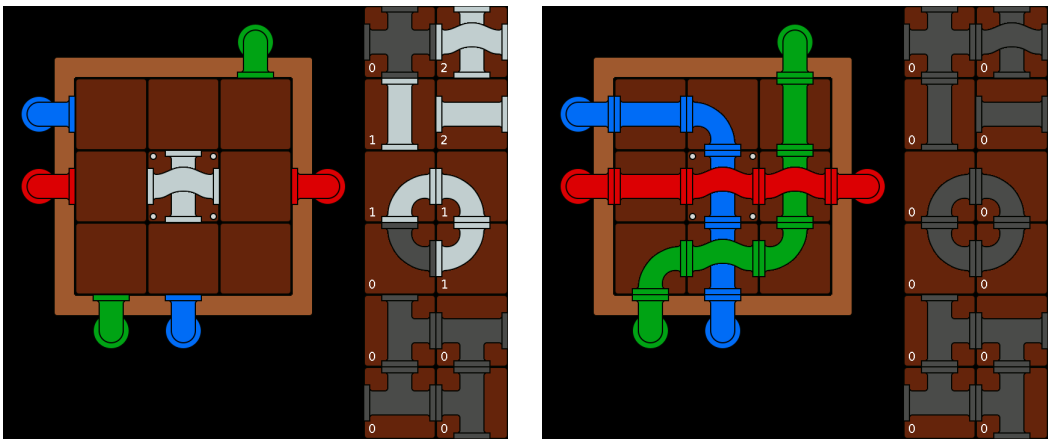
Projet de programmation

Le but de ce projet est d'écrire en `java`, à l'aide de la librairie `swing`, une application permettant de jouer au jeu vidéo "Plumber"[1].

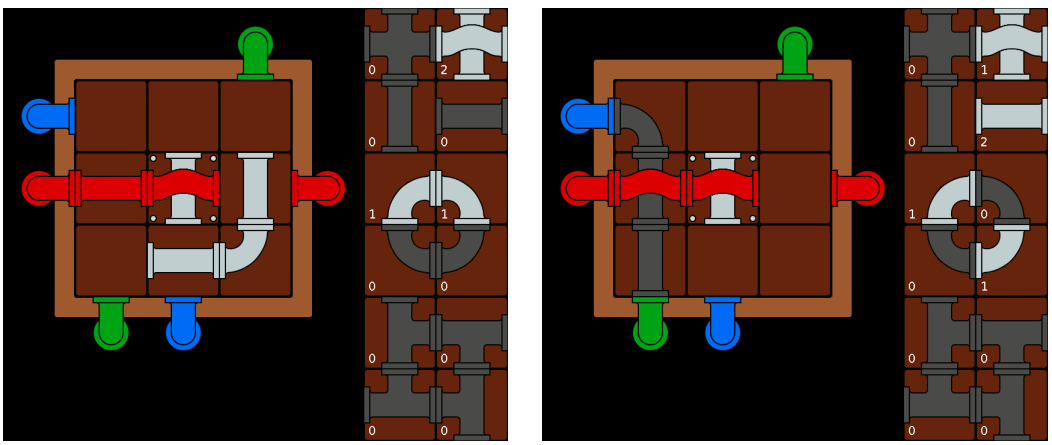
1 Le jeu Plumber

Le jeu Plumber se joue sur un damier rectangulaire de taille quelconque. Sur la bordure du plateau sont disposées des sources d'eau de différentes couleurs. Le joueur dispose d'une collection de tuyaux de différentes sortes, qu'il peut librement placer sur une case vide de manière à connecter certaines des cases au bord de celle-ci. Le but du jeu est de placer tous les tuyaux de la réserve sur le plateau de manière à ce que chaque source d'eau soit connectée via une suite de composantes de tuyaux non vide à au moins une source de même couleur et à aucune de couleur différente, et qu'aucune composante de tuyau connectée à au moins une source ne soit aussi connectée à un espace vide. Noter que cette propriété est trivialement vérifiée si la réserve et le plateau sont vides et si sa bordure ne contient aucune source.

Dans les figures ci-dessous, les couleurs sont celles affichées par le jeu. La première image montre la configuration initiale du plateau de taille 3×3 proposé au joueur. Sur sa partie droite se trouvent les 12 sortes de tuyaux possibles du jeu, avec pour chaque sorte une indication du nombre d'exemplaires de ce tuyau proposées au joueur. Le tuyau de la case centrale, formé de deux composantes reliant deux à deux les bords opposés de cette case, est vissé au plateau et ne peut être déplacé. La seconde image montre l'unique solution de cette configuration initiale.



Les deux images suivantes montrent deux étapes du jeu dans un état intermédiaire. Dans la première, les composantes de tuyaux connectées à une source sont affichées de la couleur de celle-ci, celles qui ne sont connectées à aucune source étant affichées d'une couleur neutre. Dans la seconde, deux sources de couleurs différentes, bleu et vert, sont connectées entre elles : les composantes de tuyaux connectées aux deux sources sont affichées d'une couleur singulière indiquant au joueur qu'il est nécessaire de les déconnecter. Noter que les éléments épuisés ou initialement absents à droite de chaque image sont aussi d'une couleur singulière.



[1] Voir par exemple `https://www.crazygames.fr/jeu/pipe-puzzle` pour une des formes possibles de ce jeu, similaire à celle présentée ici.

2 Implémentation du jeu Plumber en Java

Votre projet devra permettre de jouer au jeu Plumber sur un ensemble de configurations initiales – des *niveaux* encodés par des fichiers textes. Les images de cet énoncé ne font qu'illustrer le principe du jeu et de son interface et ne sont données qu'à titre indicatif : il ne vous est pas demandé de les reproduire de manière exacte dans votre implémentation, mais seulement de respecter la spécification détaillée dans cette section.

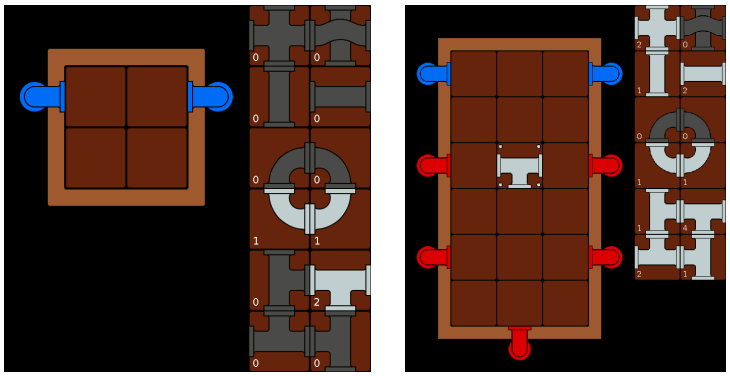
Le design de l'interface, les graphismes, le placement des différents éléments, etc., sont libres, mais les contraintes de comportement ci-dessous devront toutes être respectées, à commencer par la première d'entre elles : le jeu doit être intégralement jouable à la souris, sans jamais avoir besoin de recourir au clavier – ce qui ne vous empêche en rien de définir des raccourcis clavier pour certaines actions.

2.1 L'écran de jeu

L'écran d'accueil du programme devra permettre de sélectionner une banque de niveaux parmi deux banques possibles, et pour la banque choisie, sélectionner un niveau parmi un ensemble de niveaux numérotés par de simples entiers. Une fois un niveau choisi, cet écran sera remplacé par un écran de jeu, affichant dans deux zones distinctes le plateau de jeu dans la configuration initiale de ce niveau, et la réserve de tuyaux.

La réserve doit comporter 12 zones cliquables, une pour chaque sorte de tuyau. Elle doit indiquer en permanence le nombre de tuyaux de chaque sorte encore disponibles, et indiquer de manière visuellement claire que ce nombre est nul sans avoir à lire sa valeur.

Les dimensions de l'écran de jeu ne doivent pas excéder des valeurs maximales choisies de manière à ce qu'il ne recouvre pas une trop grande portion de l'écran de la machine. L'échelle d'affichage sera adaptée de manière à respecter cette contrainte. Par exemple, les portions d'écrans de jeu ci-dessous ont la même hauteur en nombre de pixels : la taille des éléments a été réduite à droite afin de respecter cette contrainte :



L'écran de jeu doit également contenir : un bouton permettant de revenir à l'ecran d'accueil, deux boutons permettant de défaire ou de rétablir les derniers déplacements valides de tuyaux (voir ci-dessous), un bouton permettant de revenir à la configuration initiale du niveau, plus tout autre élément que vous jugerez utile (aide, solution, score, etc).

2.2 Règles de coloration des composantes

A chaque instant du jeu, les sources sont affichées de leur couleur. Une composante de tuyau qui n'est connecté à aucune source est affiché d'une couleur neutre. Si une composante est connectée à au moins une source, et si toutes les sources auxquelles elle est connectée sont de même couleur, cette composante est affichée de cette couleur. Enfin, si une source est connectée à une source de couleur différente, toutes les composantes auxquelles elle est connectée sont affichées d'une couleur singulière.

2.3 Détection de la configuration gagnante

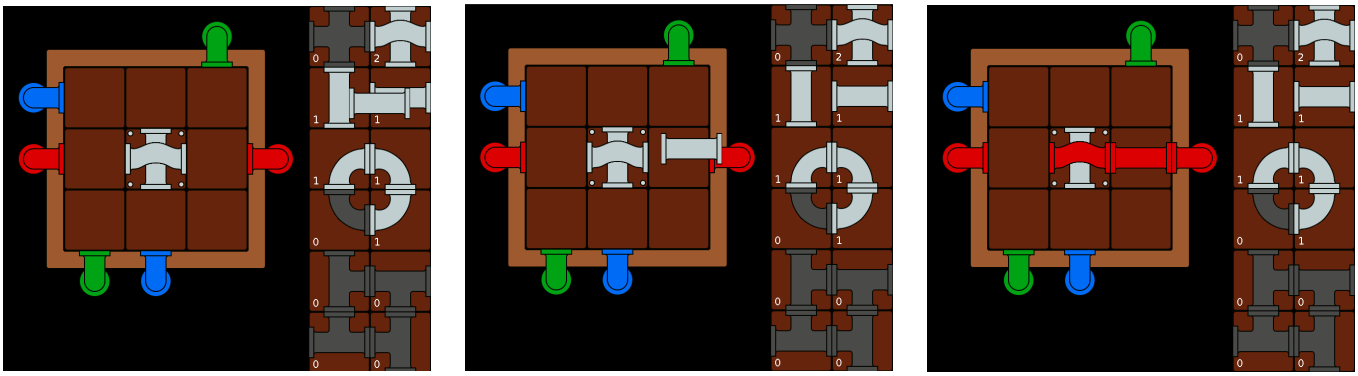
Après chaque nouvelle coloration, le programme devra bien sûr pouvoir détecter si la configuration courante est gagnante selon la règle du jeu présentée dans l'introduction, et si c'est le cas, en informer le joueur, passer s'il existe au niveau suivant, ou revenir à l'écran d'accueil si le niveau joué est le dernier.

2.4 Déplacements de tuyaux

Les tuyaux non vissés doivent pouvoir être transférés par drag and drop de la réserve vers une case vide du plateau, d'une case du plateau vers une autre case vide, ou encore être écartés du plateau et automatiquement remis dans la réserve. Voici l'interaction avec l'interface permettant d'effectuer la première catégorie d'actions, ainsi que le comportement attendu de l'interface :

1. L'utilisateur clique dans une zone de la réserve associée à une sorte de tuyau dont il reste encore au moins un exemplaire, et maintient le bouton de la souris. Un tuyau est extrait de la réserve et apparaît sous son pointeur. Ce tuyau suivra le pointeur pendant toute la durée du geste de l'utilisateur.
2. Il relâche le bouton de la souris au dessus d'une case vide.
3. Le tuyau est automatiquement déplacé par mouvement rectiligne et continu depuis sa position courante jusqu'à sa position centrale dans la case, et est ajouté au plateau à la fin de ce déplacement.

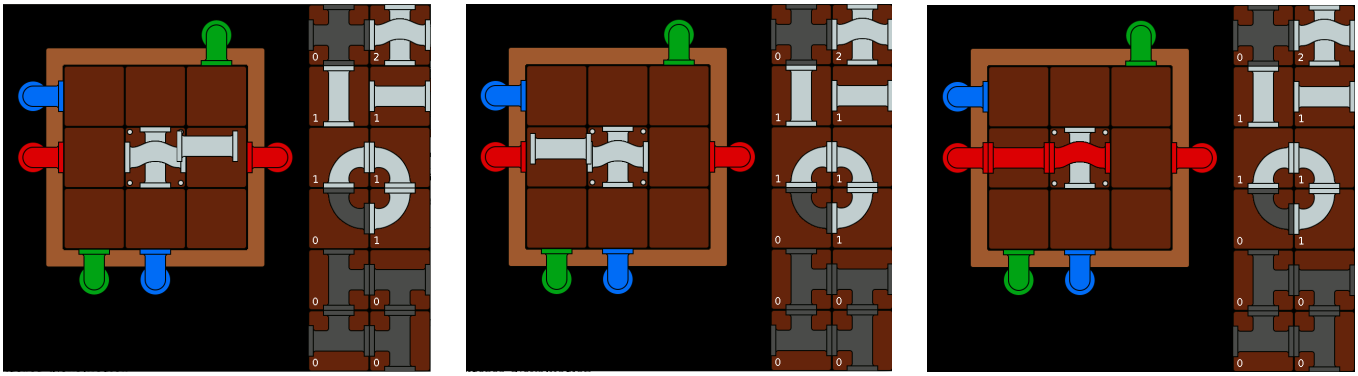
Visuellement :



La seconde catégorie se décrit de manière similaire :

1. L'utilisateur clique sur une case du plateau contenant un tuyau et maintient le bouton de la souris. Le tuyau est extrait du plateau et suivra le pointeur pendant toute la durée du geste.
2. Il relâche le bouton de la souris au dessus d'une case vide.
3. Le tuyau est automatiquement déplacé par mouvement rectiligne et continu depuis sa position courante jusqu'à sa position centrale dans la case, et est ajouté au plateau à la fin de ce déplacement.

Les figures suivantes illustrent ces trois étapes :



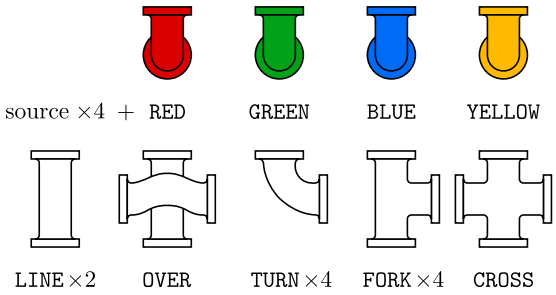
Il est évidemment interdit de déplacer une source, ou d'extraire de la réserve un tuyau dont il ne reste aucun exemplaire. Le cas où, comme-ci dessus, l'utilisateur extrait un tuyau du plateau ou de la réserve mais termine son geste au dessus d'une zone de l'écran qui n'est pas une case vide, se gère de la manière suivante :

1. Si l'utilisateur termine son geste à l'extérieur du plateau, le tuyau revient automatiquement par mouvement rectiligne et continu vers sa zone associée dans la réserve, et est remis dans la réserve à la fin ce déplacement.
2. Si le tuyau a été extrait du plateau mais relaché sur une case non vide, il revient automatiquement par mouvement rectiligne et continu vers sa case d'origine, et est remis dans cette case à la fin de ce déplacement.

Le retour d'un tuyau vers son emplacement initial n'est pas considéré comme un déplacement dans l'historique du jeu : seuls les déplacements de la réserve vers le plateau, du plateau vers la réserve ou d'une case vers une autre case peuvent être annulés ou rétablis.

2.5 Fichiers de niveaux

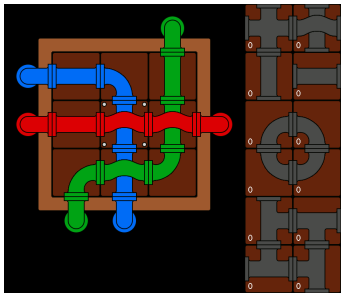
A rotation près, il n'y a qu'une seule source de quatre couleurs possibles, que nous appellerons `RED`, `GREEN BLUE`, `YELLOW`, et cinq sortes de tuyaux que nous appellerons `LINE`, `OVER`, `FORK`, et `CROSS`. Un tuyau ou une source peut être tourné de 90^o vers la droite 1, 2 ou 4 fois avant de retrouver sa fonctionnalité d'origine. La figure ci-dessous montre ces éléments dans ce que nous considérerons comme leur orientation par défaut :



Un *fichier de niveau* est un fichier texte dont le nom est de la forme `pipe`*num*`.p` où *num* est le numéro du niveau. Il est structuré de la manière suivante :

- Le fichier commence par deux entiers h w, suivis de $h \times w$ chaînes séparées par des espacements. Ces chaînes spécifient d'une part le contenu du plateau dans une configuration gagnante, d'autre part le contenu de sa bordure – le plateau contient $(h-2)\times(w-2)$ cases, sa bordure contient 4 coins et $(2\times h)+(2\times w)$ emplacements faisant face à une case, donc pouvant contenir une source.
- La chaîne `"X"` spécifie un élément de bordure vide. Une chaîne commençant par `R`, `G`, `B` ou `Y` spécifie sur la bordure la présence d'une source dont la couleur a ce caractère comme première lettre.
- La chaîne `"."` spécifie qu'une case du plateau est vide. Une chaîne commençant par `L`, `O`, `T F` ou `C`, éventuellement précédé du caractère `*`, spécifie sur une case du plateau la présence d'un tuyau ayant ce caractère comme première lettre. La présence d'une étoile indique que le tuyau est vissé au plateau.
- Un caractère `R`, `G`, `B`, `Y`, `L`, `O`, `T F C` dans une chaîne est immédiatement suivi de `0` (zéro), `1`, `2` ou `3`. Ce nombre indique, pour une source ou un tuyau, le nombre de rotations d'un quart de tour vers la droite que doit subir cet élément dans son orientation par défaut pour atteindre son orientation dans la solution du niveau[2].

Voici par exemple le contenu du fichier encodant le niveau présenté dans l'introduction :



```
5 5
X   X   X   G2  X
B1  L1  T2  L0  X
R1  L1 *O0  O0  R3
X   T1  O0  T3  X
X   G0  B0  X   X
```

Le choix d'un niveau dans l'écran d'accueil sera suivi de la lecture du fichier de niveau associé, afin de reconstruire en mémoire une configuration gagnante du niveau. Les tuyaux non vissés seront simplement transférés dans une réserve initialement vide avant que l'écran de jeu ne soit proposé au joueur.

2.6 Éditeur de niveau

Les deux banques proposées par l'écran d'accueil n'ont pas le même statut. Si l'utilisateur sélectionne la seconde banque, l'écran d'accueil lui donnera la possibilité de jouer à l'un des niveaux de manière ordinaire, mais aussi de passer en mode édition. Les choix d'interfaçage pour effectuer les actions décrites ci-dessous sont libres, mais toutes devront pouvoir être effectuées à la souris de façon fluide. Toute action irréversible – modification effective d'un fichier de niveau, abandon de modifications – devra être confirmée par l'utilisateur.

En mode édition, l'utilisateur doit pouvoir depuis l'écran d'accueil vider un niveau de tous ses tuyaux et sources sans modifier sa géométrie, altérer la géométrie d'un niveau en le vidant implicitement de tous ses éléments – les deux actions mettrons bien sûr à jour le fichier du niveau – ou éditer un des niveaux de la banque sans encore modifier son fichier.

Une demande d'édition de niveau entraîne le passage à un écran de jeu affichant non plus un plateau vide, mais la solution du niveau. La réserve de tuyaux devient dans ce mode illimitée. L'utilisateur doit bien sûr pouvoir déplacer

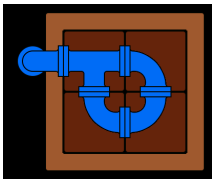
[2]Cette information est inutile pour `CROSS` ou `OVER`, et `0` ou `1` suffisent pour `LINE`, mais il est plus simple d'avoir un encodage uniforme pour toutes les sortes de tuyaux.

les tuyaux comme précédemment, mais aussi : ajouter une source sur la bordure, changer la couleur d'une source, supprimer une source, visser un tuyau posé ou le dévisser. Lors d'une demande de retour à l'écran d'accueil, si le plateau est dans un état gagnant, le fichier du niveau doit être mis à jour. Sinon, les modifications faites sur le niveau seront abandonnées.

2.7 Niveaux aléatoires (en bonus)

Si vous avez complété les parties précédentes, vous pouvez ajouter la possibilité pour l'utilisateur de tirer à tout moment dans l'éditeur de niveau une configuration aléatoire de sources et de tuyaux qui est aussi une solution, ce qui est un problème assez simple.

Vous pouvez aussi lui permettre de chercher une solution pour l'ensemble des sources courantes, sans modifier le nombre de ces sources, leur placement ou leur couleur, ce qui est un problème beaucoup moins simple. Certaines configurations n'ont pas de solution, d'autres peuvent avoir une solution inattendue :



3 Critères d'évaluation

3.1 Qualité de la conception objet et du code

Ce cours est aussi un cours de programmation, plus précisément un cours de programmation objet. Servez-vous de l'héritage, des interfaces de la liaison dynamique et des énumérations partout où ces éléments améliorent la factorisation et la clarté du code. Prenez le temps de réfléchir à la hiérarchie nécessaire, et à la répartition des données et des traitements, et ne programmez pas (jamais) par copier-coller.

3.2 MVC

Vous avez vu en cours l'architecture MVC, et ce type de projet se prête idéalement à son usage : votre code sera à la fois mieux organisé, plus facile à développer, plus facile à débugger, et valorisé par le choix de cette architecture si elle est réalisée dans sa forme la plus académique.

3.3 Ergonomie de l'interface

Comme son intitulé l'indique, ce cours est un cours d'interfaces graphiques. Votre interface doit être souple, intuitive, naturellement utilisable par un utilisateur même s'il la découvre pour la première fois.

Une image `gif`, sa version `svg` et quelques fichiers de niveau vous sont fournis. L'image contient tous les éléments nécessaires pour afficher les éléments de l'écran de jeu sans perdre de temps à les dessiner – l'esthétique des graphismes est sans importance pour ce projet (elle est toujours améliorable) et ne sera pas pris en compte dans l'évaluation. Nous vous invitons à ajouter à votre rendu des niveaux que vous aurez vous-mêmes créés – ce qui devrait être facile une fois l'éditeur complété.

4 Modalités

4.1 Groupes

Le projet doit être impérativement réalisé en binôme. Toutefois, en cas d'impossiblité majeure de satisfaire cette contrainte (groupe en nombre impair, conditions d'études particulières, etc.), nous vous invitons à en discuter le plus tôt possible avec votre enseignant de cours magistral - aucun travail non réalisé en binôme ne sera accepté sans son accord explicite. La répartition des tâches au sein d'un groupe doit être raisonnablement équilibrée. En cas de déséquilibre avéré, les notes finales pourront être individualisées.

4.2 Individualité de chaque projet

De manière évidente, votre code doit être strictement personnel. Nous pouvons tolérer l'adaptation d'exemples repris sur le Swing Tutorial d'Oracle ou dans les introductions des API des classes, mais pas la reprise de code trouvé sur le web ou venant d'une quelconque "aide" trouvée sur un forum[3], encore moins le partage de code entres groupes[4]. Il

[3] Vous savez vous servir d'un moteur de recherche, nous aussi. Les fonctionnalités et comportements décrites dans ce sujet sont précises. Un projet s'écartant singulièrement de l'énoncé nous paraîtra fatalement suspect.

[4] La soutenance de projet est un examen comme un autre. Le plagiat en projet constitue une fraude aux examens, passible au pire de lourdes sanctions discplinaires – ce cas s'est produit plus d'une fois dans cette UFR.

relève de votre responsabilité de faire en sorte que votre code reste inaccessible aux autres groupes : par exemple, si vous vous servez d'un dépôt `git`, ce dépôt doit être privé.

4.3 Forme du rendu

Les modalités et dates de rendu seront précisées ultérieurement. Votre rendu consistera en:

- un code-source écrit en `java`, compilable et utilisable tel quel sous Linux et/ou sur les ordinateurs de l'UFR,
- un fichier texte nommé README contenant vos noms,
- un rapport de quelques pages au format PDF décrivant votre projet, et expliquant et justifiant les choix de conception ou d'implémentation,
- tout autre fichier nécessaire à la compilation et à l'exécution.

Tous ces éléments seront placés dans une unique archive compressée en `.tar.gz`.

L'archive devra s'appeler *nom1*`-`*nom2*`.tar.gz`, et s'extraire dans un répertoire *nom1*`-`*nom2*`/`, où *nom1* et *nom2* sont les noms des deux personnes constituant le groupe. Par exemple, si vous vous appelez Denis Diderot et René Descartes, votre archive devra s'appeler `diderot-descartes.tar.gz` et s'extraire dans un répertoire `diderot-descartes/`.