Projet Informatique – Sections Electricité et Microtechnique

Printemps 2019 :*Obstructed Dodgeball* © R. Boulic

Rendu2 (dimanche 19 mai 23h59)

1. Buts, forme du lancement du programme, organisation du code et tests

But : simulation, performances, rapport.

SyntaxeS du lancement de l'exécutable pour le dernier rendu (section 7.3.1 de la donnée) :
Les syntaxes des rendus précédents, de même que la détection d'erreurs, doivent continuer à fonctionner. En plus, pour le dernier rendu, le mode suivant est pratique pour la mise au point et les tests des algorithmes sans lancer l'interface graphique :

```
./projet Step nom_fichier_in.txt nom_fichier_out.txt
```

Le mode **Step** sert à 1) ouvrir le fichier dont le nom suit le mot-clef « Step» et initialiser l'état du jeu puis 2) à exécuter **une seule mise à jour** de l'état du jeu et 3) à sauvegarder l'état obtenu dans le fichier dont le nom suit en dernier argument. Le programme s'arrête dès la première erreur trouvée dans le fichier à lire. Il s'arrête aussi après l'écriture du fichier.

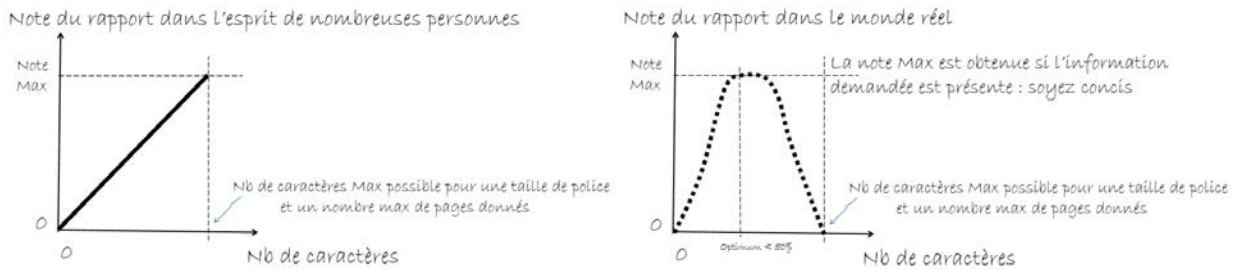
Architecture à adopter (fig 9b de la donnée) :
Même architecture que le rendu2.

Evaluation du dernier rendu : en plus d'évaluer les critères sur la qualité du code (ne pas oublier de corriger tout warning obtenu au rendu1), nous effectuerons une évaluation manuelle de votre programme comme suit avec un ensemble de fichiers de deux types :

- des fichiers simples pour valider chaque élément des règles à mettre en œuvre
- des fichiers plus lourds, avec par exemple nbCell grand, pour évaluer la performance et la robustesse du code sur une grande variété de cas.

2. Rapport final

Entre 2 (min) et 4 pages maximum, avec une police 11 au minimum et 14 au maximum, interligne simple
(SVP : ne gaspillez pas de papier avec des pages de titre et de table des matières).



Le rapport est écrit en français ou en anglais ; une orthographe ou une grammaire défaillante peut induire les correcteurs en erreur. Le rapport contient :

Architecture logicielle et description de l'implémentation:

- Dessin de l'architecture logicielle seulement si elle est différente de celle de la **figure 9b**. Si c'est le cas, justifier les différences.
- Dessin de la hiérarchie de classes si vous en avez créé une (pas demandé).
- Préciser la structuration des données ; où étaient gérés les ensembles d'entités ?

- Décrire et justifier la répartition des tâches d'une mise à jour entre les modules du Modèle. Qui calcule quoi ? Quand la matrice des distances est-elle calculée pour la première fois ? Quand est-elle recalculée ?
- Donnez une estimation du coût calcul et mémoire lors du calcul de la mise à jour d'*un pas de la simulation*. Indiquez seulement ***le terme dominant*** en fonction des paramètres nbPlayer, nbBall, nbObstacle, nbCell; ne PAS remplacer les paramètres par N.
- Illustration avec 8 à 12 images, éventuellement recadrées, de l'évolution de la simulation sur un exemple comportant au moins deux joueurs qui évitent un obstacle en se dirigeant l'un vers l'autre puis lancent une balle. Vous pouvez utiliser Alt-PrtSc pour récupérer une image de l'exécution de votre programme.

Méthodologie et conclusion : comment avez-vous organisé votre travail à plusieurs, indiquer la personne responsable de chaque module et comment vous avez organisé le travail au sein du groupe (par quels modules avez-vous commencé, comment les avez-vous testés, comment le feriez-vous maintenant avec le recul.

Quel était le bug le plus fréquent, pourquoi ? et celui qui vous a posé le plus de problème et comment a-t-il été résolu, ...). Pour conclure fournissez une brève auto-évaluation de votre travail et de l'environnement mis à votre disposition (points forts, points faibles, améliorations possibles).

Le rapport final doit être inclus dans le fichier archive du rendu final (en format pdf).
Il doit aussi être *imprimé et apporté en INJ 141 avant le lundi 20 mai à midi.*

3. Forme du dernier rendu

Pour chaque rendu **UN SEUL membre d'un groupe** (noté **SCIPER1** ci-dessous) doit télécharger un fichier **zip** sur moodle (pas d'email). Le non-respect de cette consigne sera pénalisé de plusieurs points. Le nom de ce fichier **zip** a la forme :

SCIPER1_ SCIPER2.zip

Compléter le fichier fourni **mysciper.txt** en remplaçant 111111 par le numéro SCIPER de la personne qui télécharge le fichier archive et 222222 par le numéro SCIPER du second membre du groupe.

Le fichier archive du rendu1 doit contenir (**aucun répertoire**) :

- Fichier texte édité **mysciper.txt**
- Votre fichier **Makefile** produisant un executable **projet**
- votre code source (.cc et .h)
- **le fichier pdf du rapport**

*On doit pouvoir produire l'exécutable **projet** à partir du Makefile après décompression du contenu du fichier zip.*

Auto-vérification : Après avoir téléchargé le fichier zip de votre rendu sur moodle (upload), récupérez-le (download), décompressez-le et assurez-vous que la commande make produit bien l'exécutable et que celui-ci fonctionne correctement.

Exécution sur la VM: votre projet sera évalué sur la VM du second semestre (disponible depuis le 6 mars).

Backup : Vous êtes responsable de faire votre copie de sauvegarde du projet. Il y a un backup automatique seulement sur votre compte myNAS. Sur la VM, vous devez activer vous-même le backup (icone « engrenage » en haut à droite, choisir system settings, choisir backup et activer cette fonction en précisant les paramètres). Une alternative est de s'envoyer la dernière version du code source par email.

Gestion du code au sein d'un groupe :

- vous pouvez envisager d'utiliser **gdrive.epfl.ch** pour définir un répertoire partagé par les 2 membres du groupe et pas plus. Il n'y a pas d'éditeur de code en mode partagé.
- L'option c4science n'est recommandée que pour ceux qui maitrisent déjà git. Si vous l'utilisez, vous devez supprimer l'accès public (par défaut) à votre code.