TP d’algorithmique en langage C

Jeu d’exploration

On souhaite coder un jeu d’exploration en langage C. Il sera possible de voyager de lieu en lieu, de gérer une jauge d’endurance, et d’affronter l’hostilité de l’environnement naturel.

Consignes

Pour entamer le développement de chacune des fonctionnalités, il faut tirer une nouvelle branche ; sauf indication contraire, la fusion est réalisée à la fin de chaque exercice.

Avant de commencer, il convient :

- de supprimer le contenu du répertoire C:\Users\game\Documents\GitHub, et de déconnecter l’utilisateur précédent sur GitHub Desktop si nécessaire ;
- de créer un dépôt **public** et d’en envoyer l’URL à : selene.t.etpa@gmail.com

Exercice 1 - Choix par numéro

1. Tirer une branche “deplacement” pour ne pas développer directement sur le *master*.
2. Créer le fichier explo.c dans le répertoire de travail local. Il contiendra l’ensemble du code de notre projet de jeu d’exploration.
3. Coder un programme permettant d’afficher 2 destinations et permettant de les choisir par leur numéro. Suite au choix, la destination est annoncée comme atteinte.
4. Faire un commit “choix par numero”. Le pousser (push) sur le repository distant.
   Ne pas fusionner votre branche avec le *master* (pas encore).

Exercice 2 - Choix par nom

1. Améliorer le programme de l’exercice 1 en faisant en sorte que le choix ne s’effectue plus par le numéro du lieu mais par son nom.
2. Faire un commit “choix par nom”. Le pousser (push) sur le repository distant.
   Fusionner votre branche avec le *master*.

Exercice 3 - Gestion des lieux

1. Tirer une nouvelle branche “gestion des lieux”.
2. Créer une structure Lieu et le type associé.
   Chaque lieu possède :
   - un nom
   - une description
   - une valeur chiffrée de difficulté (qui servira pour les déplacements)
   - un tableau d’id (les lieux auxquels on aura accès depuis celui-ci)

Selene Tonon
ETPA Rennes & Toulouse

3. Instancier trois lieux au début du main(). Chaque lieu sera désigné par un identificateur de format lieu+numero de lieu (exemples : lieu1, lieu2, lieu3). On considère que les lieux ont une difficulté allant de 1 à 19.
4. Améliorer le programme de l’exercice 2 en codant une fonction “deplacement” permettant de voyager du lieu actuel au lieu de destination. Lors du voyage, on affiche la description du nouveau lieu.
5. Commit. Push. Pull. Merge.

Exercice 4 - Aiguillage

1. Tirer une nouvelle branche “contraintes deplacement”
2. Créer un tableau de lieux dans lequel on stocke les lieux précédemment instanciés. S’en servir pour permettre au joueur de voyager dans les lieux en les référençant par leur index dans le tableau de lieux. Ainsi, lieu12 devient lieux[12].
3. Lors du déplacement, vérifier que le lieu de destination était effectivement éligible. S’il ne l’était pas, ne pas changer le lieu courant, et afficher à la place un message disant que ce déplacement est impossible.
4. Commit. Push. Pull. Merge.

Exercice 5 - Gestion de l’endurance

1. Tirer une nouvelle branche “stamina”
2. Créer une jauge d’endurance, qui vaut 100 en début de partie. Faire en sorte qu’à chaque déplacement, elle soit réduite de la somme de la difficulté des deux lieux concernés. En d’autres termes, le coût du déplacement est égal à la difficulté du lieu de départ, à laquelle on ajoute la difficulté du lieu d’arrivée.
3. Créer une action “Etablir un avant-poste”, accessible dans le cas où on se trouve dans un lieu de difficulté >10. Elle a pour effet de diviser par deux la difficulté du lieu (arrondi à l’inférieur). Elle ajoute à la description du lieu la mention. “\nIl y a un avant-poste dans ce lieu.”
4. Créer une action “repos”, uniquement accessible dans les lieux ayant une difficulté strictement inférieure à 5. Cette action restaure la jauge d’endurance de 50 points, avec un maximum de 100.
5. Commit. Push. Pull. Merge.

Exercice 6 - Bonus

Mettre en place un système permettant d’externaliser la création des différents lieux, dans un document texte situé dans le même répertoire que l’exécutable. Se servir pour cela de la fonction fopen() accessible grâce à l’inclusion de stdio.h.