Projet de fin de Semestre

Rogue-Like dans le terminal

1 Consignes

1.1 Modalités

- Groupes de 4 (sauf exception validée par les encadrants) : 25% de la note du module
- Analyse et conception en TD *présence vérifiée*
- Rapport de conception de 6 pages maximum **Date à fixer en TD**
- Implémentation en TP *présence vérifiée*
- Démonstration/Evaluation dernier créneau de TP **Semaine 50**
- Rendu final sur madoc **Vendredi 19 décembre**

1.2 Livrables

Rapport de conception de 6 pages maximum **Date à fixer en TD** : Un fichier contenant
- les noms, prénoms et N°Etudiant des membres du groupe
- les types retenus et leur rôle
- les rôles, préconditions et signatures des algos et sous algos identifiés
- le regroupement en modules des types et sous algos ainsi que le graphe de dépendances
- la planification et la répartition des tâches

Rendu final sur madoc **Vendredi 19 décembre** : Une archive contenant
- les fichiers source `.cpp` et `.hpp` nécessaires pour les programmes et les tests
- un fichier texte `lisez_moi.txt` donnant le mode d'emploi avec des exemples d'exécution
- les fichiers d'entrée (grille, équipements ...) utilisés dans les exemples d'exécution
- un compte-rendu texte synthétique précisant les noms, prénoms et N°Etudiant des membres du groupe et tâches choix, limites et spécificités de votre projet, ainsi que des informations qui vous semblent utiles pour son évaluation

1.3 Evaluation du travail en groupe

- Groupes hétérogènes souhaités
- Nécessite de savoir expliquer et convaincre
- Du travail visible doit être fait durant les TDs TPs
- **Chaque membre doit posséder une copie à jour du projet**
- **Chaque membre doit pouvoir répondre aux questions**
  - où se trouve telle fonctionnalité dans le code ?
  - quelles alternatives ont été évaluées mais non choisies ?
  - quel impact aurait tel changement dans le code ?
  - quel impact aurait tel changement dans la spécification ?

1.4 Rappel du barème

Couverture fonctionnelle : note sur 20 dépendant des fonctions réalisées Puis malus allant jusqu'à :

- Compilation (100%)
- Warnings (20%).
- commentaires judicieux et indentation. (20%)
- Noms et casse des identifiants. (20%)
- Types des variables judicieux. (20%)
- Préconditions (20%)
- Les interactions avec l'utilisateur doivent être soignées. (20%)
- jeu de tests exécutable (20%)

N'utiliser ces critères qu'à la toute fin de rédaction d'un programme ne fonctionne pas.
Pratiquer des cycles courts (Spécification - Transcription - Test) pour chaque fonctionnalité.