Mini-projet

Un jeu avec implémentation de structures de données dynamiques

Objectifs du Travail

- Utilisation des structures de données simplement chainées (listes chainées et files)
- Implémentation (en C) d'opérations sur les structures utilisées
- Evaluation de complexité théorique
- Décomposition d'un programme en sous-programmes et conception de la solution globale
- Rédaction d'un rapport

Important - Remise des travaux

- Le travail doit être réalisé en **binôme**
- Le rapport (compte-rendu) doit être fourni au format **pdf** dans un **fichier à part**
- Le code source (un seul fichier) bien commenté et lisible doit être fourni au format **wordPad**
- Le code source + rapport doivent être envoyés à l'adresse **recupsapce@gmail.com**, dans deux fichiers séparés en précisant comme objet du mail « MiniProjet ALGO – nomsEtudiants - groupes »
- Date limite d'envoi des travaux : jeudi **15/01/2026**
- Les affichages de résultats à l'écran doivent être bien alignés et lisibles.
- Les cas de copiages seront sanctionnés par un **Zéro.**

Un grand nombre de problèmes réels ont recours à la représentation de données sous forme de structures linéaires (listes, piles, files, …etc). Dans ce TP, on désire implémenter un jeu de hasard simple, en utilisant les structures de listes chainées et files.

Description du jeu

Un ensemble de joueurs s'inscrivent à un jeu et sont initialement rangés dans une file d'attente F (num joueur, nom joueur, age). La file F contient alors **n** joueurs.

Partie I

Une partie de jeu implique 2 joueurs (pris de la tête de file F), les 2 joueurs demandent au système de générer de *manière aléatoire* une valeur entière, *à tour de rôle*. Si la *somme des chiffres* de la valeur générée est un multiple de 5, le joueur obtient un score de 1, sinon il obtient 0.

Par exemple si la valeur générée est ***230541,*** la somme de ses chiffres est = 15 est un multiple de 5 donc le joueur obtient un score de 1.

La partie est arrêtée dès que *l'écart des points* entre les 2 joueurs atteint **3** points.

La partie est alors enregistrée avec un numéro de partie « Partie 1, Partie 2, … », les numéros et les noms des 2 joueurs ainsi que les scores obtenus. Pour chaque joueur, il faudra sauvegarder le nombre de parties gagnées et le nombre de parties perdues.

Le joueur **gagnant** va jouer une autre partie et le joueur **perdant** est placé en *queue de file* F.

Pour une autre partie de jeu, un joueur de la *tête de file* F est sélectionné et on refait le jeu de la même manière, avec le joueur gagnant.

Après la génération de ***12 valeurs*** aléatoires dans une partie, si l'écart entre les 2 joueurs n'a pas atteint 3 points, la partie est arrêtée.

En cas d'égalité de scores, les deux joueurs sont placés en queue de file F ; et pour une autre partie du jeu, deux autres joueurs sont sélectionnés de la tête de file F.

Règles

Un joueur qui **gagne** trois fois de suite est placé dans une file **F1** de priorité 1

Un joueur qui **perd** 3 fois de suite (ou séparées) est placé dans une file **F3** de priorité 3

Si un joueur perd cinq (05) fois même séparées, il sort du jeu et sera mis dans une **liste LP** de joueurs perdants.

Si un joueur gagne cinq (05) fois même séparées, il sera mis dans une **liste LG** de joueurs gagnants, ordonnée par *ordre décroissant* des points cumulés sur les 5 parties.

Pour les différentes parties du jeu, les joueurs sont sélectionnés selon la priorité de la file :

- De la file **F1** si elle n'est pas vide
- Ensuite de la file **F** si elle n'est pas vide
- Ensuite, de la file **F3** jusqu'à ce qu'elle se vide.

S'il reste un seul joueur dans la file F1, un joueur de la file F sera sélectionné

S'il reste un seul joueur dans la file F, un joueur de la file F3 sera sélectionné

S'il reste un seul joueur dans la file F3 (à la fin), il est déclaré perdant et sera mis dans la liste LP.

Partie II Changement de règles

Au bout de **3n** parties de jeu (3 fois le nombre de joueurs dans la file initiale), il reste encore des joueurs dans les files, on **change la stratégie du jeu** comme suit :

Le joueur demande au système de générer aléatoirement *2 nombres*, si le PGCD des 2 nombres contient au moins un chiffre qui appartient à l'un des 2 nombres générés, le joueur obtient un score de 1, sinon il a zéro.

Par exemple si le système génère les 2 nombres 462 et 910, PGCD (462, 910) = 14, le chiffre 4 est dans le premier nombre (aussi le chiffre 1 est dans 910, il suffit d'avoir un seul) --- donc le joueur obtient un score de 1.

Dès que **l'écart** des points entre les joueurs atteint **3 points**, la partie est arrêtée. Le joueur *gagnant* passe dans la file **F1** et le joueur *perdant* passe dans la file **F3**.

Après la génération de ***16 valeurs*** aléatoires, si l'écart entre les 2 joueurs n'a pas atteint 3 points, la partie est arrêtée, le joueur qui a le score max est placé dans **F** et l'autre est placé dans **F3**.

En cas d'égalité de scores, les deux joueurs sont placés en queue de file F ; et pour une autre partie du jeu, deux joueurs sont sélectionnés (de F1, ensuite de F, ensuite de F3 – voir plus bas).

Si un joueur perd deux (02) fois même séparées, il sort du jeu et sera mis dans une **liste LP** de joueurs perdants.

Si un joueur gagne deux (02) fois *de suite*, il sera mis dans une **liste LG** de joueurs gagnants, ordonnée par *ordre décroissant* des points cumulés sur les 2 parties.

Pour les différentes parties de jeu, les joueurs sont sélectionnés selon la priorité de la file (comme dans la partie I) :

- De la file **F1** si elle n'est pas vide
- Ensuite de la file **F** si elle n'est pas vide
- Ensuite, de la file **F3** jusqu'à ce qu'elle se vide.

S'il reste un seul joueur dans la file F1, un joueur de la file F sera sélectionné

S'il reste un seul joueur dans la file F, un joueur de la file F3 sera sélectionné

S'il reste un seul joueur dans la file F3 (à la fin), il est déclaré perdant et sera mis dans la liste LP.

Arrêt du jeu :

Le jeu s'arrête lorsque tous les joueurs sont dans l'une des deux listes **LP** ou **LG**, les files F, F1 et F3 sont vides.

Si au bout de **2n** parties (avec la nouvelle stratégie), il reste encore des joueurs dans les files, les joueurs de **F1** sont mis dans la liste **LG** et les joueurs de **F et F3** sont placés dans la liste **LP**.

Questions

I- Conception algorithmique

Décrire en algorithmique tous les types de données utilisés dans ce jeu.

Décrire les fonctions et procédures nécessaires à la réalisation de ce jeu (il faudra réfléchir à la manière de décomposer la solution), évaluez la complexité des différentes opérations.

Ecrire l'algorithme principal du jeu global

II – Implementation en langage C

Implémenter toutes les structures de données en C

Implémenter les différentes fonctions/procédures ainsi que le programme C du jeu global

Pour chaque partie, afficher les détails de progression du jeu et les résultats de fin de chaque partie

Afficher l'état des files F, F1 et F3 et des listes LP et LG à la fin de chaque partie

A la fin du jeu, il faudra afficher les 3 premiers gagnants avec leurs scores respectifs

Autres requêtes

Quels sont les joueurs qui n'ont remporté aucune partie

Avec la première stratégie,

Afficher les joueurs ayant remporté 1, 2, et 3 parties

Afficher les joueurs ayant perdu 1, 2 et 3 parties

Avec la deuxième stratégie,

Afficher les joueurs ayant remporté 1, 2 parties

Afficher les joueurs ayant perdu 1, 2 parties

PS : horodater les parties (heure de début, heure de fin)

***PS :** Pour les tests, prévoir une action qui construit automatiquement la file initiale des joueurs par programme (on ne va pas saisir les données au clavier).*

Indications

- L'interface principale du programme doit monter le numéro de la partie du jeu, les numéros et noms des joueurs, la date de début et date de fin de la partie, la progression de la partie et le résultat de la partie de jeu.
- A chaque fin de partie, montrer les états des files F, F1 et F2 et les listes LP et LG
- Montrer le changement de stratégie et la progression du jeu avec la seconde stratégie.

Compte-rendu à rédiger (10 à 15 pages max)

Une page de garde est nécessaire
Un rapport doit être rédigé en suivant le plan ci-après :

1. **Introduction**
2. **Objectifs du travail**
3. **Partie I** (voir plus haut)
4. **Partie II (**fournir le code C bien commenté dans un fichier wordPad à part**)**
5. **Conclusion** (conclure par rapport à l'utilisation des structures de données et la complexité des opérations)

   Peut-on conclure par rapport à l'équité du jeu ?

Annexe : Montrer des captures d'écran illustrant l'exécution du programme (choisir quelques captures les plus significatives et sans redondance)