

Nom : _____

PROGRAMMATION STRUCTURÉE
420-1Q2-SW

Examen final
AUTOMNE 2024

Pondération 40%

SPÉCIFICATIONS :

- Les noms et les types de variables doivent être cohérents, en camelCase, sans caractères spéciaux.
- Pour les questions de code, celui-ci doit compiler pour éviter des pénalités.
- L'examen est d'une durée de 3h00. Vous avez toutefois 4h30 pour le faire.
- NE PAS déplacer l'emplacement du fichier .cpp qui est unique à chaque poste de travail. À la fin de l'examen, vous devez simplement vous assurer qu'une sauvegarde a été faite avant de quitter en laissant le poste et la session ouverts.
- Sauf pour les variables spécifiées dans les questions, vous pouvez déclarer et utiliser les variables de votre choix.
- Vous avez le droit d'utiliser les vecteurs et les `std::array` si vous les préférez aux tableaux classiques, tant que vous respectez les demandes des fonctions.
- Seule la feuille de notes imprimée et remplie de toutes les notes que vous voulez, recto-verso, est permise. Aucune autre documentation et aucun accès à internet ne sont permis.
- Vous avez accès à Visual studio 2022 et aux fonctionnalités de débogueur qu'il apporte en tout temps dans l'examen.
- En tout temps, la répétition de code dont la logique est déjà traitée dans une autre fonction sera pénalisée.

Mise en contexte :

Vous allez créer une mini-application de gestion de mises sportives qui va permettre de suivre et d'analyser les mises effectuées par différents joueurs au cours de l'année. Chaque joueur peut enregistrer ses mises hebdomadaires, qui sont ensuite utilisées pour calculer des statistiques comme la mise maximale et la moyenne des mises. Les fonctionnalités incluent :

- La création de profils pour les joueurs avec leurs mises hebdomadaires.
- L'affichage des informations détaillées pour chaque joueur, y compris leur nom, la semaine courante, et leurs mises.
- L'enregistrement des statistiques des joueurs dans un fichier texte pour un suivi à long terme.
- La gestion flexible des mises, avec un maximum de trois mises par joueur par semaine.

Pour y parvenir, vous devez suivre les étapes ci-dessous. **Ne pas toucher** à la définition (le contenu) de la fonction présente dans le fichier cpp (*semainesDepuisDebutAnnee*). Elle est disponible pour utilisation seulement. Elle retourne le nombre de semaines écoulées depuis le début de l'année.

Travail à faire :

QUESTION 1. (3 points)

Créer une struct `T_Joueur` qui permet de mettre

- le nom,
- le numéro de la semaine
- le tableau de 3 mises en argent d'un joueur pour la semaine.

Par défaut, le nom est vide, le numéro de la semaine est à -1 et les 3 mises sont à 0. Bien penser aux types de vos champs.

Note : Pour tester ce bout de code, faire la question 8a)

QUESTION 2. (9 points)

Faire la fonction *calculerMoyenne*, qui prend un tableau du même type que les mises. La fonction retourne la moyenne des mises de ce tableau, peu importe le nombre d'éléments dans le tableau.

Note : Pour tester ce bout de code, faire la question 8b)

QUESTION 3. (10 points)

Faire la fonction *miseMaximale*, qui prend un tableau du même type que les mises et retourne la valeur maximale des mises. Si nécessaire, voici les valeurs minimal et maximales pour les 3 types de base qu'on a vus :

Int : INT_MAX, INT_MIN

Float : FLT_MAX, FLT_MIN

Double: DBL_MAX, DBL_MIN

Note : il est suggéré de tester cette fonction, elle sera utilisée dans une autre fonction plus tard. Ce n'est pas obligatoire, mais fortement suggéré.

QUESTION 4. (14 points)

Créer la fonction *ajouterMises* qui prend en paramètre un T_Joueur **qui doit être modifié aussi dans l'environnement appelant**. La fonction **ne retourne rien**. Dans la fonction, on fait ceci :

- Demander en console à l'utilisateur s'il veut faire une mise (l'utilisateur répondra par 'o' ou 'n').
- Si l'utilisateur veut faire une mise, on affecte la valeur du numéro de la semaine dans le champ *numeroSemaine* du joueur. Pour avoir le numéro de semaine, utiliser la fonction *semaineDepuisDebutAnnee*.
- Ensuite, tant que le nombre de mises ne dépasse pas la capacité du tableau de mises (3), indiquer le # du montant à entrer, prendre la réponse et mettre la réponse dans la mise à la case appropriée du tableau. On repose ensuite la question pour savoir s'il y a une autre mise à faire.

Exemple 1 (si on entre 3 mises)

```
Voulez-vous faire une mise? (o ou n): o
Montant de la mise #1 : 34.5
Ajouter une autre mise? (o ou n): o
Montant de la mise #2 : 6.5
Ajouter une autre mise? (o ou n): o
Montant de la mise #3 : 12.4
```

Après la 3^e réponse 'o', on sort de la fonction et on passe au prochain traitement.

Exemple 2 (on n'entre pas 3 mises (les valeurs restantes seront 0 par défaut si la question 1 a été faite correctement))

```
Voulez-vous faire une mise? (o ou n): o
Montant de la mise #1 : 34
Ajouter une autre mise? (o ou n): n
```

Après la réponse 'n', on sort de la fonction et on passe au prochain traitement

- *Important* : on ne doit pas demander plus de mises que la taille du tableau de mise du joueur.
- *Détail* : Aucune vérification n'est demandée pour s'assurer qu'il y a déjà des mises ou non. Aucune vérification pour les saisies de clavier incorrecte non plus.

QUESTION 5. (10 points)

Faire la fonction *creerJoueur*, qui ne prend aucun paramètre et qui **retourne** un T_Joueur rempli avec les champs. Dans cette fonction,

- on doit d'abord demander en console quel est le nom du joueur. Le nom du joueur sera entré par l'utilisateur et affecté au champ *nom* du joueur qu'on a créé.
- pour remplir les autres champs, **vous devez utiliser la fonction que vous avez déjà codée pour faire cette tâche**. Rappel : Une répétition ici de la logique déjà mise dans une autre fonction vous pénalisera.

Note : Pour tester ce bout de code, faire la question 8c)

QUESTION 6. (9 points)

Créer la fonction *afficher* qui prend **UNT_Joueur** en paramètre et qui n'a pas de retour. Cette fonction devra afficher en console

- Le nom du joueur
- S'il y a au moins une mise de faite, afficher le numéro de semaine et la ou les mises. Si on n'a pas au moins une mise, écrire seulement « Aucune mise de faite cette semaine »

Voici un exemple (à gauche) où on avait 2 joueurs dont un qui avait 3 mises et un qui avait une seule mise (donc les autres sont à 0 par défaut). À droite, on a un exemple d'un joueur qui n'a pas eu de mise.

```
*****
Nom: Marc
Semaine en cours: 47
Mise #1 :34.5
Mise #2 :44.5
Mise #3 :58
*****
Nom: Damien
Semaine en cours: 47
Mise #1 :5.6
Mise #2 :0
Mise #3 :0
```

```
*****
Nom: Karine
Aucune mise faite à ce jour.
```

Note : Pour tester ce bout de code, faire les questions 8d) et e)

QUESTION 7. (12 points)

Faire la fonction *ecrireStats* qui prend un **tableau** de T_Joueur en paramètre et écrit dans un fichier nommé mesStats.txt. Cette fonction retourne un booléen (true si l'écriture dans le fichier s'est bien passée, false sinon)

- L'écriture doit écraser le contenu précédent, s'il y en a un
- Pour chaque joueur, il faut écrire dans le fichier :
 - o le # du joueur et son nom
 - o la semaine en cours
 - o la mise maximale de ce joueur en utilisant la fonction prévue à cet effet.

Note : Pour tester ce bout de code, faire la questions 8f)

QUESTION 8. (23 points)

Appels **dans le main()** :

a) (5 points) Créer 2 joueurs directement dans le main(), sans utiliser aucune fonction.

Le premier sera nommé joueur1 avec les caractéristiques suivantes :

- o Nom : Marc
- o Semaine en cours (utiliser la fonction *semainesDepuisDebutAnnee* pour le trouver)
- o 3 mises : 34.5, 44.5, 58

Le deuxième sera nommé joueur2 avec les caractéristiques suivantes :

- o Nom : Julien
- o Semaine en cours (utiliser la fonction *semainesDepuisDebutAnnee* pour le trouver)
- o 3 mises : 12.5, 17.25, 10

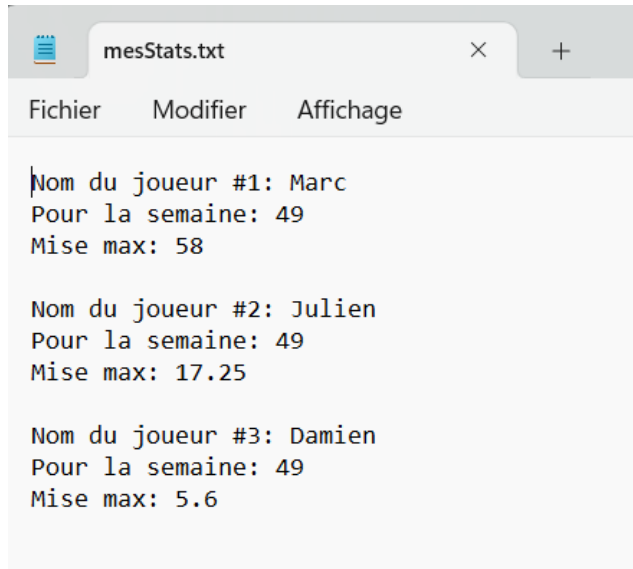
- b) (2 points) Afficher en console la moyenne du joueur 1 (Marc) en utilisant la fonction *calculerMoyenne*

```
La moyenne du joueur 1:  
45.6667
```

- c) (3 points) Créer un joueur nommé joueur3 en appelant la fonction *creerJoueur*. Le joueur devra avoir les caractéristiques suivantes :
- Nom : Damien
 - Entrer **une seule mise** de 5.6 (après une mise, répondre 'n' et les 2 dernières mises doivent rester automatiquement à 0)
 - La valeur du numéro de semaine (50 pour le 19 décembre 2024) devrait se remplir tout seul
- d) (3 points) Créer un tableau de T_Joueur nommé mesJoueurs. L'initialiser en mettant vos 3 joueurs dedans.
- e) (4 points) Parcourir votre tableau mesJoueurs et afficher en console le contenu. La fonction *afficher* doit obligatoirement être utilisée. N'oubliez pas que ce tableau pourrait avoir plus de 3 joueurs! Vous devriez avoir 50 pour la semaine en cours. Il est aussi conseillé de tester l'Affichage avec un joueur qui n'a aucune mise!

```
*****  
Nom: Marc  
Semaine en cours: 49  
Mise #1: 34.5  
Mise #2: 44.5  
Mise #3: 58  
*****  
Nom: Julien  
Semaine en cours: 49  
Mise #1: 12.5  
Mise #2: 17.25  
Mise #3: 10  
*****  
Nom: Damien  
Semaine en cours: 49  
Mise #1: 5.6  
Mise #2: 0  
Mise #3: 0
```

- f) (2 points) Appeler la fonction *ecrireStats*. Vérifiez que le fichier s'est bien créé et que son contenu est ceci :



```
Nom du joueur #1: Marc
Pour la semaine: 49
Mise max: 58

Nom du joueur #2: Julien
Pour la semaine: 49
Mise max: 17.25

Nom du joueur #3: Damien
Pour la semaine: 49
Mise max: 5.6
```

- g) (2 points) Manuellement, dans le `main()`, modifiez la 2^e mise de Marc. Celle-ci devra être changée (écrasée) pour 19.5. Aucune fonction à utiliser.
- h) (2 points) Manuellement, dans le `main()`, modifiez le nom de Marc pour Marc-André. Aucune fonction à utiliser.

Joyeuses Fêtes! Ce fut un plaisir de vous enseigner!

