CÉGEP SHAWINIGAN
Du savoir et des gens

Version A24

420-W0Q-SW-A24 - Projet Battleship

Type de travail : En équipe (2-3)
Pondération : 40%

Délai accordé et remise

Délai accordé

Selon le calendrier présent, vous avez jusqu'au 4 décembre à 23h59 pour remettre le travail. Après cette date, une pénalité de 10% par jour de retard est appliqué conformément à la politique du département.

Remise

Par Git. Vous devrez remettre votre code de façon continue tout le long de ce projet.

Vous devez vous assurer que l'enseignant a accès à votre code source à la date de remise.

NB. Vous pouvez faire autant de commits que vous voulez. Sauf demandes spécifiques, le dernier commit avant la date limite sera considéré pour la correction.

Conseils avant de débuter

Prenez le temps de **bien lire et comprendre le(s) énoncé(s)**.

CÉGEP SHAWINIGAN
Du savoir et des gens

Règles et directives

- C'est un travail en équipe. Donc vous devez travailler ensemble pour atteindre les différents objectifs.
- Le code en français ou en anglais est accepté, mais vous devez être CONSTANTS dans votre choix. Si vous codez en français, faites le partout. Si vous codez en anglais, faites le partout.
- Indiquez toujours vos références lorsque c'est possible. Par exemple, si vous trouvez un snippet de code sur internet, vous devez en indiquer la provenance en commentaires.
- Vous devez **respecter les standards et conventions**!
  - Code bien indenté
  - Code uniforme (Ex: Code appliqué de la même façon dans la solution)
    - Sauts de lignes
    - Tabulations vs espaces
  - Nomenclature du bon format (PascalCase, camelCase, ...)
  - Nomenclature adéquate (Noms appropriés et pertinents)
    Ex. : "Toto" pour un compteur n'est pas très pertinent.

BON SUCCÈS

Objectif :
développer une version console du jeu **Battleship (Bataille Navale / Touché Coulé)** en C#. Ce projet vous permettra de pratiquer et d'évaluer l'ensemble des concepts de programmation vus en classe : conditions, boucles, fonctions, tableaux et structs.

Description du jeu :
Dans ce jeu, deux joueurs (un humain et l'ordinateur) placent secrètement leurs bateaux sur une grille de 10x10. Chaque joueur, selon un principe tour à tour, tentent de localiser et couler les bateaux de l'autre en tirant sur des coordonnées spécifiques. Le premier joueur à couler tous les bateaux de l'adversaire gagne. (https://www.youtube.com/watch?v=pbgJbX5ADRQ)

Règles du jeu :

- **Grille de jeu :** La grille est de taille 10x10, représentée par des lettres de A à J pour les colonnes et des nombres de 1 à 10 pour les lignes (ex. B4).
- **Bateaux :** Chaque joueur a 5 bateaux de différentes tailles :
  - Porte-avions (5 cases)
  - Croiseur (4 cases)
  - Contre-torpilleur (3 cases)
  - Sous-marin (3 cases)
  - Patrouilleur (2 cases)
- **Tour de jeu :**
  - Le joueur entre les coordonnées d'une attaque (ex. A5), et le jeu indique si la tentative est un **Touché**, un **Manqué** ou un **Coulé**.
  - Le jeu continue jusqu'à ce qu'un des joueurs ait coulé tous les bateaux de l'adversaire.

Fonctionnalités à implémenter :

1. **Initialisation de la grille :**
   - Chaque joueur (humain et ordinateur) doit placer ses bateaux sur une grille de 10x10. Les bateaux peuvent être placés horizontalement ou verticalement, mais ils ne peuvent pas se chevaucher.
   - Vous pouvez laisser l'ordinateur placer ses bateaux de manière aléatoire.
2. **Affichage de la grille :**
   - Afficher deux grilles : une pour les attaques du joueur (avec les résultats touché/manqué) et une autre pour la position des bateaux du joueur.
   - L'ordinateur n'affiche que les attaques du joueur (sans révéler l'emplacement de ses bateaux).
3. **Tour de jeu :**
   - Le joueur entre les coordonnées de tir (par exemple, D5), et le jeu lui indique si le tir est un touché ou un manqué.
   - L'ordinateur joue son tour de manière aléatoire, mais sans attaquer une case déjà ciblée.
4. **Gestion des bateaux coulés :**
   - Le jeu doit informer le joueur lorsqu'un bateau est entièrement coulé.
5. **Fin du jeu :**
   - Le jeu se termine lorsqu'un joueur a coulé tous les bateaux adverses. Le gagnant est annoncé.
6. **Rejouer ou quitter :**
   - À la fin d'une partie, proposer au joueur de recommencer une nouvelle partie ou de quitter le jeu.

Éléments techniques :

- **Structs :** Utilisez des structs pour représenter les bateaux avec des attributs tels que la taille, la position initiale, et si le bateau est coulé ou non.
- **Tableaux :** Utilisez un tableau à deux dimensions pour représenter les grilles de jeu. Chaque case doit indiquer si elle contient une partie d'un bateau, si elle a été touchée, ou si elle a été manquée.
- **Fonctions :** Implémentez des fonctions pour :
  - Placer les bateaux
  - Vérifier si un tir est touché, manqué ou coulé
  - Afficher les grilles
  - Vérifier la fin du jeu

N.B.: Assurez vous de bien **tester** votre jeu avant de le remettre. Chaque entrée de l'utilisateur doit être validée correctement. Une nouvelle entrée doit être demandée en cas d'erreur jusqu'à ce une entrée valide soit faite et que le programme puisse continuer. Un programme livré ne devrait pas "planter" ou arrêter de s'exécuter de façon imprévue.

Pondération :
Les retours des équipiers (En plus de ceux de l'enseignant) influenceront également vos notes de façon générales.

- **Fonctionnalité :**
  - Le jeu doit fonctionner correctement, en suivant les règles du Battleship.
  - La gestion des attaques et des résultats (touché, manqué, coulé) doit être précise.
- **Qualité du code :**
  - Le code doit être bien structuré, lisible et commenté.
  - Les fonctions doivent être utilisées de manière appropriée pour éviter la répétition de code.
- **Interface utilisateur :**
  - L'interface en console doit être claire et permettre une bonne interaction avec le joueur (affichage des grilles, instructions, résultats).
- **Utilisation des concepts C# :**
  - Vous devez démontrer une bonne compréhension des structures de données (tableaux, structs) et des contrôles de flux (conditions, boucles).
- **Originalité et améliorations :**
  - Des points supplémentaires seront attribués si vous proposez des fonctionnalités supplémentaires ou des améliorations (par exemple : niveaux de difficulté, IA plus intelligente, statistiques de jeu, interface utilisateur avancée, etc.). **Vous devez spécifiez clairement dans votre documentation et commentaires** quelles fonctions supplémentaires vous avez mis en place et expliquer leur fonctionnement.

Les retours des équipiers (En plus de ceux de l'enseignant) influenceront également vos notes de façon générales. Une pondération individuelle sera appliquée sur le résultat de base. Ex: si le projet donne un résultat de 80% et qu'un individu obtient un résultat personnel de 80% -> Note individuelle sera 80% de 80% = 64%

Critères	**/40**
Menus en boucle et avec validations	/2
Clarté des menus et évènements (Messages)	/2
Affichage des grilles (Attaques et bateaux)	/3
Disposition des bateaux (Manuel pour joueur)	/3
Disposition des bateaux (Aléatoire pour l'ordinateur)	/3
Gestion du tour du joueur	/3
Gestion du tour de l'ordinateur	/3
Conditions de frappe	/3
Conditions de victoire	/3
Utilisation des fonctions, tableaux et structs	/5
Respect des standards et bonnes pratiques	/5
Documentation et explication des règles	/5
(**Bonus**) Fonctions supplémentaires	/5

Détails des critères

Critères techniques et fonctionnels (35 points)

1. Menus en boucle et avec validations (2 points) :
   - Les menus doivent permettre à l'utilisateur de naviguer facilement et offrir des options claires pour rejouer ou quitter.
   - La validation des entrées est cruciale pour éviter les erreurs.
2. Clarté des menus et des événements (2 points) :
   - Les messages affichés (comme "Touché !" ou "Manqué !") doivent être clairs et bien formatés pour guider l'utilisateur.
3. Affichage des grilles (3 points) :
   - Deux grilles doivent être affichées : une pour les attaques et une pour les bateaux.
   - Les informations doivent être bien organisées et lisibles.
4. Disposition des bateaux (6 points) :
   - **Manuel pour le joueur (3 points) :** Le joueur doit pouvoir placer ses bateaux de manière intuitive en entrant des coordonnées.
   - **Aléatoire pour l'ordinateur (3 points) :** Les bateaux doivent être positionnés aléatoirement sur la grille sans chevauchement.
5. Gestion des tours (6 points) :
   - **Tour du joueur (3 points) :** Le joueur doit pouvoir attaquer une case et recevoir un retour clair (touché, manqué, coulé).
   - **Tour de l'ordinateur (3 points) :** L'ordinateur doit attaquer une case aléatoire sans répéter ses tirs.
6. Conditions et règles (6 points) :
   - **Conditions de frappe (3 points) :** Le programme doit correctement évaluer si une attaque touche un bateau ou non.
   - **Conditions de victoire (3 points) :** Le jeu doit se terminer lorsqu'un joueur a coulé tous les bateaux de l'adversaire.
7. Utilisation des structures de données et des concepts C# (5 points) :
   - Les structs, fonctions, et tableaux doivent être utilisés efficacement pour structurer le code et éviter la répétition.

Qualité et documentation (15 points)

1. Respect des standards et bonnes pratiques (5 points) :
   - Le code doit être propre, bien structuré, et suivre les conventions (nommage des variables, indentation, commentaires, etc.).
2. Documentation et explication des règles (5 points) :
   - Le projet doit inclure une documentation claire expliquant les règles du jeu et le fonctionnement du code.
3. Fonctions supplémentaires (Bonus, 5 points) :
   - Des points supplémentaires peuvent être accordés pour des fonctionnalités originales ou des améliorations :
     - IA plus intelligente
     - Statistiques de jeu (nombre de coups, ratio de précision)
     - Interface utilisateur avancée (par exemple, utilisation de couleurs)
