SAé 1.01 : Initiation au Développement
Bataille Navale

1 Sujet

1.1 Règles du jeu

Le but de cette SAé est d'implémenter une version jouable en terminal du jeu de la Bataille Navale.

Le jeu se joue à deux joueurs. Ils doivent tour à tour torpiller le camp de leur adversaire afin de couler l'ensemble de ses navires. Le premier a avoir coulé les 5 navires de l'adversaire gagne la partie.

Le jeu se déroule de la façon suivante : les deux joueurs commencent par placer leurs 5 navires sur leur plateau, invisible à l'adversaire.

Ensuite, le jeu se déroule par tours. À chaque tour , le joueur ayant la main annonce une coordonnée (ex: E4) qui est alors révélée par l'adversaire. Si ces coordonnées appartiennent à un navire de son adversaire, celui-ci annonce touché. Si toutes les cases d'un navire ont été touchées, l'adversaire annonce alors coulé. Sinon, l'adversaire annonce à l'eau. Si le joueur touche (ou coule) un navire, il garde la main. Sinon, l'autre joueur peut jouer.

1.2 But de la SAé

La SAé se déroulera du 14 Novembre au 6 Janvier 2023. Par groupes de deux, vous devrez développer un jeu de bataille navale se déroulant entièrement sur terminal, via les entrée et sortie standards. Un fichier exécutable *BatailleNavale.o* est fourni afin d'illustrer le comportement du jeu attendu.

Dans une première partie, vous aurez accès à une bibliothèque *biblioNavale.h* contenant des fonctions facilitant le développement du jeu. Vous devez vous appuyer sur la bibliothèque pour développer votre jeu.

Dans un second temps, vous ne toucherez plus à cette première partie, mais devrez coder la bibliothèque initialement fournie.

1.2.1 Utilisation des ressources

Pour la première partie, les fichiers biblioNavale.h et biblioNavale.o devront être dans votre répertoire courant. Votre code devra inclure la bibliothèque locale, via l'instruction :

```
#include "biblioNavale.h"
```

Pour compiler votre code prenant en compte cette bibliothèque locale, vous devrez ajouter *biblioNavale.o* comme argument de gcc. Exemple :

```
gcc MaBatailleNavale.c -o MaBatailleNavale.o biblioNavale.o
```

Pour la seconde partie, votre bibliothèque navale devra être compilée avec l'option *-c*. Exemple :

```
gcc -c biblioNavale.c
```

Cela créera le fichier biblioNavale.o pouvant être utilisé tel que dans la première partie.

1.3 Dates

Chaque étape devra être terminée avant le jour de la date à **18h**, l'heure du mail u-pec (ou de git) faisant foi. Des points pourront être retirés à partir de la première minute de retard.

Lundi 14 Novembre **Groupes de 2 : M'envoyer les binômes par mail (luc.dartois@u-pec.fr).**

Jeudi 1er Décembre **Fin de la première partie : code principal: *BatailleNavale.c***

Vendredi 6 Janvier **Fin de la seconde partie : bibliothèque *biblioNavale.c***

1.4 Livrables

Le sujet de la SAé est fourni avec :

- Un fichier biblioNavale.h, contenant le header de la bibliothèque biblioNavale.
- Un fichier biblioNavale.o, contenant le code de la bibliothèque déjà compilé.
- Un fichier BatailleNavale.o, permettant de tester le rendu attendu.

1.4.1 Mode de rendu

La SAé devra être placée sur le compte git de l'IUT (dwarves.iut-fbleau.fr/gitiut/) de l'un des binômes du groupe. Les seuls participants à ce projet seront les membres du binômes ainsi que l'utilisateur *dartoisl*.

Vous serez évalués à la fin de chaque partie. Les critères seront la viabilité du jeu, son respect des règles, mais également l'utilisation pertinente des fonctions de la bibliothèque pour la première partie, et la clarté du code et le respect des spécifications pour la deuxième partie.

1.4.2 Rendu Partie 1.

Le rendu de la première partie doit être un fichier BatailleNavale.c utilisant la bibliothèque *biblioNavale.h* et fonctionnant similairement à l'exécutable fourni en exemple. À noter qu'il n'est pas nécessaire d'utiliser toutes les fonctions fournies par la bibliothèque. Néanmoins, un malus sera appliqué si vous recodez une fonctionnalité qui était déjà fournie par la bibliothèque.

Les attendus sont, dans l'ordre :

- Un fichier compilant sans erreur ni warning.
- Initialisation et Affichage des parties révélées des plateaux des 2 joueurs.
- Possibilité pour l'utilisateur de jouer des coups sur un plateau.
- Gestion de la victoire d'un joueur.
- Gestion de deux joueurs différents jouant sur les deux plateaux.
- Gestion correcte du tour (rejouer si touché).
- Détection et gestion des mauvaises entrées utilisateur.
- Détection et affichage du "coulage" d'un bateau (quand toutes ses positions ont été touchées).

1.4.3 Rendus Partie 2.

Le rendu de la seconde partie doit être un fichier biblioNavale2.c contenant tout ou une partie des fonctions de biblioNavale.c. Chaque fonction redéfinie aura le même nom suivi de Deux (ex : afficheDuoDeux).

Vous utiliserez votre BatailleNavale.c (ou à défaut celui fourni à la fin de la partie 1), où vous remplacerez les noms des fonctions que vous avez redéfinies par les noms de votre propre fonction (de façon à pouvoir tester chaque fonction séparément). Tout autre changement effectué pour faire coller le code principal à ce que vous aurez réussi à coder est prohibé.

2 Détails techniques

2.1 Les navires

Chaque joueur possède 5 navires :

- Le *Porte-Avion*, de longueur 5, et symbolisé par la lettre A.
- Le *croiseur*, de longueur 4, et symbolisé par la lettre C.
- Le *Sous-Marin*, de longueur 3, et symbolisé par la lettre S.
- Le *Mous-Sarin*, de longueur 3, et symbolisé par la lettre M.
- Le *Torpilleur*, de longueur 2, et symbolisé par la lettre T.

2.2 biblioNavale.h

La bibliothèque initialement fournie contient les constantes et fonctions ci-dessous :

- `void ajoutNavireAleatoire(char tab[][10],char b,int t);`

  La fonction ajoutNavireAleatoire permet d'ajouter au tableau tab un Navire de taille t, et représenté par le caractère b.

  résultat : t cases consécutives de tab reçoivent le caractère b.

  fonctionnement : La fonction choisit aléatoirement si le navire sera horizontal ou vertical, puis en fonction de cela choisit aléatoirement la position de façon à ce que le navire ne dépasse pas du tableau.

- `int verif(char tab[][10]);`

  La fonction verif vérifie que le tableau tab contient bien les navires demandés.

  résultat : renvoie l'entier 1 si tab contient exactement 17 cases contenant un caractère différent de l'espace

  renvoie 0 sinon.

- `void affiche(char t[][10]);`

  affiche permet d'afficher sur la sortie standard le plateau de jeu t, sous forme de grille de bataille navale.

- `void affichedeuo(char t[][10],char p[][10]);`

  affichedeuo permet d'afficher côte à côte les parties révélées des plateaux t et p, sous forme de grilles de bataille navale.

- `void initPlateau(char plat[][10]);`

  initPlateau permet à un joueur d'initialiser le tableau plat, en y plaçant ses navires.

  résultat : A la fin de la fonction, le tableau plat contient les 5 navires de la bataille navale, placés selon les règles.

  fonctionnement : Le joueur peut placer ses navires de deux façons : - Aléatoirement, auquel cas la fonction utilise ajoutNavireAléatoire pour ajouter tous les navires de façon à obtenir un plateau correct. - A la main, auquel cas le joueur doit dire si son navire est horizontal ou vertical, et donner la coordonnée la plus nord-ouest. Si l'utilisateur place mal son Navire, il lui est demandé de le replacer.

- `int jouerJoueur(char adv[][10]);`

  La fonction jouerJoueur permet à de jouer sur le tableau adv.

  résultat : L'utilisateur entre des coordonnées, le tableau est modifié pour indiquer s'il le coup est "à l'eau" ou "touche" retourne un entier, correspondant au Navire touché : -1 si le coup est à l'eau

0 si le coup touche le Porte-Avion 1 si le coup touche le Croiseur 2 si le coup touche le Sous-Marin 3 si le coup touche le Mous-Sarin 4 si le coup touche le Torpilleur -2 si la case contient un caractère non reconnu (ne devrait pas arriver) -3 si le coup est hors plateau

- `char* navire(int i);`

  navire renvoie une chaîne de caractères correspondant à Navire, selon l'entier entré : résultat : selon l'entier i donnée en argument, navire renvoie :

  - Porte-Avion si i=1,
  - Croiseur si i=2,
  - Sous-Marin si i=3,
  - Mous-Sarin si i=4,
  - Torpilleur si i=5.