

Projet Info1 : Liste

Difficulté : F \rightarrow Facile, N \rightarrow Neutre, D \rightarrow Difficile

- Projet 1 : Blind test musiques (F)
- Projet 2 : Blind test sport (F)
- Projet 3 : Le 421 simplifié (N)
- Projet 4 : Le compte est bon (N)
- Projet 5 : Boite à rythme (F)
- Projet 6 : Bataille navale (D)
- Projet 7 : Jeu Simon (F)
- Projet 8 : Jeu des couleurs (F)
- Projet 9 : Pocker (D)
- Projet 10 : Jeu du son (N)
- Projet 11 : Le morpion (D)
- Projet 12 : Son midi (N)
- Projet 13 : Puissance 4 (D)
- Projet 14 : Le bandit manchot (N)

<http://list.alwaysdata.net/Info1/GO.html>

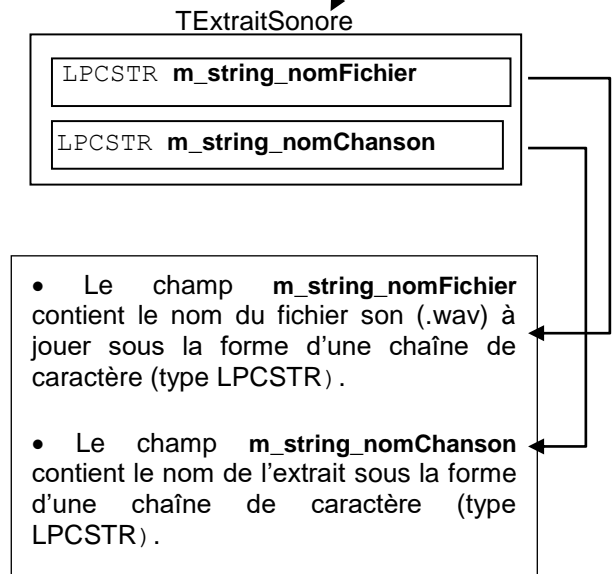
Projet 1 : Blind test musiques (F)

Vous devez déclarer un tableau `l_enrTab_song` de 21 cases de type `TExtraitSonore`. Le type `TExtraitSonore` possède la représentation ci-dessous. Le type `TExtraitSonore` existe déjà dans `PrjInfo2022.h`.

Ensuite, utilisez la ligne de code : `InitTabMusic(l_enrTab_song, 21);` pour initialiser ce tableau.

Une fois initialisé le tableau contient les valeurs suivantes :

	m_string_nomFichier	m_string_nomChanson
<code>l_enrTab_song[0]</code>	"Song0.wav"	"2Unlimited"
<code>l_enrTab_song[1]</code>	"Song1.wav"	"AlainBashung"
<code>l_enrTab_song[2]</code>	"Song2.wav"	"AndreaBocelli"
<code>l_enrTab_song[3]</code>	"Song3.wav"	"BritneySpears"
<code>l_enrTab_song[4]</code>	"Song4.wav"	"CelineDion"
<code>l_enrTab_song[5]</code>	"Song5.wav"	"Coolio"
<code>l_enrTab_song[6]</code>	"Song6.wav"	"Corona"
<code>l_enrTab_song[7]</code>	"Song7.wav"	"Haddaway"
<code>l_enrTab_song[8]</code>	"Song8.wav"	"LosDelRio"
<code>l_enrTab_song[9]</code>	"Song9.wav"	"LouBega"
<code>l_enrTab_song[10]</code>	"Song10.wav"	"LouiseAttaque"
<code>l_enrTab_song[11]</code>	"Song11.wav"	"Muse"
<code>l_enrTab_song[12]</code>	"Song12.wav"	"MyleneFarmer"
<code>l_enrTab_song[13]</code>	"Song13.wav"	"Paradisio"
<code>l_enrTab_song[14]</code>	"Song14.wav"	"PatrickBruehl"
<code>l_enrTab_song[15]</code>	"Song15.wav"	"Reel2Real"
<code>l_enrTab_song[16]</code>	"Song16.wav"	"RickyMartin"
<code>l_enrTab_song[17]</code>	"Song17.wav"	"RobertMiles"
<code>l_enrTab_song[18]</code>	"Song18.wav"	"Zebda"
<code>l_enrTab_song[19]</code>	"Song19.wav"	"ZoukMachine"
<code>l_enrTab_song[20]</code>	"Song20.wav"	"Gala"



Le jeu se joue à 2 et on impose un tableau pour stocker les points de chaque joueur. Les joueurs jouent en même temps. Avant de débiter le jeu, le programme tire au hasard 10 chiffres compris entre 0 et 20. Il est conseillé de ranger ces 10 chiffres dans un tableau `l_enrTab_num`.

La chanson qui correspond au premier chiffre du tableau `l_enrTab_num` est alors jouée.

Le joueur 1 doit appuyer sur la touche `P` lorsqu'il a trouvé l'interprète, le joueur 2 sur `A`. L'extrait s'arrête alors et l'ordinateur demande au joueur qui a appuyé sa réponse (réponse orale).

Le joueur dit sa réponse puis appuie sur `R` pour voir la bonne réponse s'afficher. L'ordinateur demande si la réponse donnée était la bonne (réponse par touche `O` ou `N`). Si c'est le cas le joueur marque un point.

Si aucun joueur n'a trouvé la réponse à la fin de l'extrait, les joueurs appuient sur `S` pour passer à la suite.

Le jeu continue avec la chanson suivante.

Amélioration 1 : Le joueur qui a appuyé pour donner le nom de l'interprète ne possède que 3 secondes pour dire sa réponse (prévoir un compte à rebours avec un *Beep* à chaque seconde). Au bout des 3 secondes, la réponse apparaît et le joueur a perdu.

Amélioration 2 : Si un joueur n'a pas donné la bonne réponse sur une chanson, il ne peut pas jouer sur la suivante, donnant un avantage certain à l'autre joueur....

Aide pour ce projet :

```
void main(void) {
    TExtraitSonore l_enrTab_song[21];    //Déclaration du tableau de chansons
    bool l_bool_test;
    int iBcl = 0;
    char l_char_lettre;
    InitTabMusic(l_enrTab_song, 21);      //Initialisation du tableau de chansons
    for(iBcl=0; iBcl<21; iBcl++){
        PlaySoundA(l_enrTab_song[iBcl].m_string_nomFichier, NULL, SND_ASYNC);    //Joue la chanson de la case iBcl du tableau
        do{
            l_bool_test = kbhit();
        }while(l_bool_test == 0);    //Attente d'appui sur une touche afin de laisser la chanson se jouer
        l_char_lettre = getch();    //Récupère la touche appuyée
        PlaySoundA(NULL, NULL, SND_PURGE);    //Arrête la chanson
        printf("%s\n", l_enrTab_song[iBcl].m_string_nomChanson);    //Affiche le nom de la chanson de la case iBcl du tableau
    }
}
```

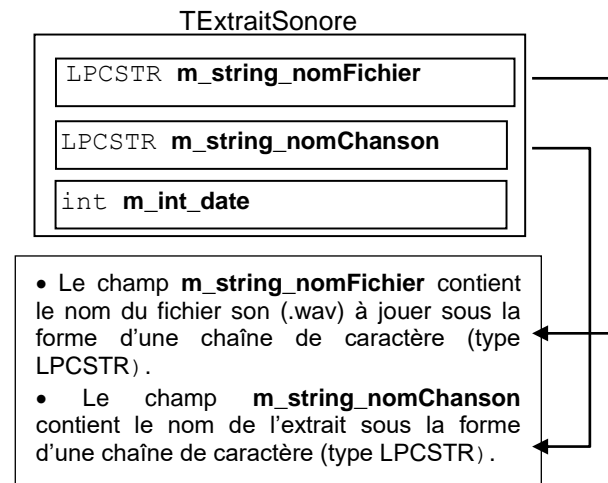
Projet 2 : Blind test sport (F)

Vous devez créer un tableau `I_enrTab_sport` de 11 cases de type `TExtraitSonore`. Le type `TExtraitSonore` possède la représentation ci-dessous. Le type `TExtraitSonore` existe déjà dans `PrjInfo2022.h`

Ensuite, utilisez la ligne de code : `InitTabSport(I_enrTab_sport, 11);` pour initialiser ce tableau.

Une fois initialisé le tableau contient les valeurs suivantes :

	m_string_nom Fichier	m_string_ nomChanson	m_int_ date
<code>I_enrTab_sport[0]</code>	"Sport0.wav"	"JoDeneriaz"	2006
<code>I_enrTab_sport[1]</code>	"Sport1.wav"	"CdmHand"	1995
<code>I_enrTab_sport[2]</code>	"Sport2.wav"	"CdeSainte"	1976
<code>I_enrTab_sport[3]</code>	"Sport3.wav"	"RdmLavillenie"	2014
<code>I_enrTab_sport[4]</code>	"Sport4.wav"	"CdeLemaitre"	2010
<code>I_enrTab_sport[5]</code>	"Sport5.wav"	"CdeOm"	1993
<code>I_enrTab_sport[6]</code>	"Sport6.wav"	"CdmPavard"	2018
<code>I_enrTab_sport[7]</code>	"Sport7.wav"	"JoPerec"	1996
<code>I_enrTab_sport[8]</code>	"Sport8.wav"	"CdmPetit"	1998
<code>I_enrTab_sport[9]</code>	"Sport9.wav"	"CdmRugby"	1999
<code>I_enrTab_sport[10]</code>	"Sport10.wav"	"TdFrance"	1989



Dans un premier temps, le jeu se joue à 2 mais on impose un tableau pour stocker les points de chaque joueur. Les joueurs jouent à tour de rôle et sont tous les 2 crédités de 100 points au départ.

Le programme choisit un extrait au hasard. Cet extrait est un commentaire sonore d'un grand évènement sportif français. A la fin de l'extrait, le programme demande au joueur de saisir la date de cet évènement (année XXXX). Le programme enlève le nombre de points au joueur avec la formule ($\text{année réelle} - \text{année saisie}$) mais n'affiche pas l'année réelle, seulement le nombre de points restant.

On passe ainsi de suite de joueur en joueur. Le perdant est celui qui arrive premier en dessous de 0.

Un même extrait peut revenir plusieurs fois, les joueurs doivent être attentifs.

Amélioration 1 : Un joueur peut très bien renseigner une date avant la fin de l'extrait (appui sur `S` pour stopper la lecture de l'extrait).

Amélioration 2 : Par la suite, on donnera un malus : Perte de un point supplémentaire par seconde écoulée entre le début de la lecture de l'extrait et l'appui sur `S`.

Aide pour ce projet :

```
void main(void) {
    TExtraitSonore I_enrTab_sport[11];           //Déclaration du tableau d'extraits sonore
    bool I_bool_test;
    int iBcl = 0;
    char I_char_lettre;
    InitTabSport(I_enrTab_sport, 11);             //Initialisation du tableau d'extraits sonore
    for(iBcl=0; iBcl<21;iBcl++){
        PlaySoundA(I_enrTab_sport[iBcl].m_string_nomFichier,NULL,SND_ASYNC); //Joue l'extrait sonore de la case iBcl du tableau
        do{
            I_bool_test = kbhit();
        }while(I_bool_test == 0);                 //Attente d'appui sur une touche afin de laisser l'extrait sonore se jouer
        I_char_lettre = getch();                  //Récupère la touche appuyée
        PlaySoundA(NULL,NULL,SND_PURGE);          //Arrête l'extrait sonore
        printf("%s : %i\n",I_enrTab_sport[iBcl].m_string_nomChanson, I_enrTab_sport[iBcl].m_int_date); //Affiche le nom et
        l'année de l'extrait sonore de la case iBcl du tableau
    }
}
```

Projet 3 : Le 421 simplifié (N)



Le 421 est un jeu de dé qui se joue à 2 joueurs et 3 dés.

Le premier joueur lance les 3 dés (simulation grâce à la fonction rand() et l'opérateur modulo %). Les trois chiffres doivent être rangés dans un tableau.

Il peut alors rejouer un ou plusieurs des 3 dés afin de former la combinaison la plus haute (voir les combinaisons possibles ci-dessous). Il joue ainsi entre 1 et 3 fois pour former la meilleure combinaison. C'est lui qui décide le nombre de lancers. A chaque nouveau lancé, il indique le numéro du ou des dés qu'il veut rejouer (exemple 23 pour rejouer les dés 2 et 3).

Une fois qu'il a fini (soit qu'il a effectué ces 3 jets maximum soit qu'il décide d'arrêter avant), le deuxième joueur joue avec les mêmes modalités : Il doit obligatoirement jouer en autant de coup que le premier joueur.

Le vainqueur de la manche est celui qui a la combinaison la plus élevée. Le vainqueur prend alors à son adversaire le nombre de points correspondant à sa combinaison.

Chaque joueur possède 30 points au départ ; le vainqueur final est celui qui a réussi à prendre tous les points de son adversaire.

A chaque nouvelle manche, c'est le vainqueur de la manche précédente qui joue en premier.

Les combinaisons sont (par ordre croissant de priorité) :

Combinaison	Points
Pas de combinaison dans une manche : C'est le nombre le plus grand formé par les 3 dés qui l'emporte.	1 point
Une paire (2 dés de la même valeur)	Valeur du dé de la paire. En cas de paires identiques, c'est le 3 ^{ème} dé qui décide du vainqueur.
Une triplète : (3 dés de la même valeur)	Valeur du dé multiplié par 3.
Combinaison 4, 2, 1 (somme des dés = 7 et multiplication des dés = 8)	21 points

Dans un premier temps, la reconnaissance des combinaisons ne sera pas gérée par le programme. A chaque fin de manche, le programme demande le vainqueur et le nombre de points à transférer.

Dans un second temps, le programme doit être capable de reconnaître la meilleure combinaison et de gérer les points lui-même. Pour reconnaître les combinaisons il est conseillé dans un premier temps de ranger les dés dans un ordre croissant.

On impose qu'un joueur soit représenté par le type structuré suivant (ce type existe déjà dans PrjInfo2022.h) :

TJoueur421

```
int m_int_points
int m_intTab_jetDes[3]
```

Et que les 2 joueurs soient déclarés sous la forme d'un tableau de TJoueur421 à 2 cases.

Aide pour ce projet :

```
void main(void) {
    TJoueur421 l_enrTab_joueurs421[2];    //Déclaration du tableau de 2 joueurs
    bool l_bool_joueur = 0;
    int iBcl;
    char l_char_lettre;

    srand(time(NULL));

    do{
        printf("Joueur %i joue\n", l_bool_joueur);

        //Boucle for qui lance 3 dés et mémorise dans le tableau du joueur et affiche :
        for(iBcl = 0; iBcl < 3; iBcl++){
            l_enrTab_joueurs421[l_bool_joueur].m_intTab_jetDes[iBcl] = rand();    //Utilisez % !!!
            printf("De %i de joueur %i = %i\n", iBcl+1, l_bool_joueur, l_enrTab_joueurs421[l_bool_joueur].m_intTab_jetDes[iBcl]);
        }
        printf("appuyez sur une touche pour passer au joueur suivant.\n");
        l_char_lettre = getch();    //Attente saisie touche
        l_bool_joueur = !l_bool_joueur;    //Pour passer au 2ème joueur
    }while(1);    //Boucle infinie
}
```

Projet 4 : Le compte est bon (N)

Cahier des charges :

On veut réaliser le jeu *le compte est bon* de France Télévision.

Règles du jeu :

Le programme doit choisir aléatoirement 6 nombres parmi une liste donnée et un nombre aléatoire compris entre [100 et 999]. Tous ces nombres sont affichés à l'écran. Les 6 nombres sont placés dans un tableau. Le programme lance une temporisation de 60 secondes (5 lors des essais). Sur les 3 dernières secondes un bip doit retentir à chaque seconde (fonction Beep()).

Les 6 nombres doivent être choisis dans le tableau suivant (à déclarer et initialiser ainsi) :

```
int l_tabInt_nombres[28] = {1,1,2,2,3,3,4,4,5,5,6,6,7,7,8,8,9,9,10,10,25,25,50,50,75,75,100,100};
```

Durant les 60 secondes, l'utilisateur doit essayer de résoudre le compte (sur papier dans un premier temps) :

Exemple pour 645 avec 50 6 7 1 5 2 l'utilisateur peut faire $6 + 7 = 13 * 50 = 650 - 5 = 645$. Il ne peut pas, bien sûr, utiliser 2 fois le même nombre.

Au bout des 60 secondes, le programme demande si l'utilisateur a réussi (saisie de O ou N) : +1 point si O, -1 point si N. Il faut arriver à 3 pour gagner.

Amélioration 1 :

Les 6 nombres tirés aléatoirement au début doivent être un des nombres du tableau mais pas 2 fois le même :

Effectivement, une fois un nombre tiré aléatoirement, il ne doit plus pouvoir l'être...

On se propose de procéder ainsi : A chaque fois qu'un tirage a été effectué, il faut décaler les cases supérieures dans le tableau de façon à écraser la valeur tirée. Le tirage suivant se fera alors sur une case de moins.

Exemple : Si au premier tirage (effectué sur les 28 cases), la case numéro 15 est choisie. Il faut décaler ainsi :

1	1	2	2	3	3	4	4	5	5	6	6	7	7	8	8	9	9	10	10	25	25	50	50	75	75	100	100
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	-----	-----

Au tirage suivant, le tirage se fera sur 27 cases. Et ainsi de suite...

Il est conseillé de travailler sur un tableau image de façon à réinitialiser ce tableau image avec le tableau de base.

Amélioration 2 : Le calcul doit être saisi et non plus fait au brouillon :

Exemple pour 645 avec 50 6 7 1 5 2 (les valeurs 50,6,7,1,5,2 doivent être placées dans un tableau) :

Le joueur doit saisir les entiers 6 et 7, puis la lettre + pour indiquer qu'il veut une addition avec les chiffres 6 et 7.

L'ordinateur doit alors effectuer l'opération, modifier le tableau et afficher :

645 avec 50 13 0 1 5 2

Puis le joueur saisie les entiers 50 et 13 et la lettre * pour indiquer qu'il veut une multiplication avec les chiffres 50 et 13.

L'ordinateur doit alors effectuer l'opération, modifier le tableau et afficher :

645 avec 650 0 0 1 5 2

Puis le joueur saisie les entiers 650 et 5 et la lettre - pour indiquer qu'il veut une soustraction avec les chiffres 650 et 5.

L'ordinateur doit alors afficher : *Le compte est bon !!!*

L'utilisateur saisie -1 s'il n'a pas su faire le compte.

Projet 5 : Boîte à rythme (F)

Une boîte à rythme est un instrument de musique qui génère des sons de base (principalement de batterie) numérisés, à intervalle de temps régulier. On obtient alors une suite de sons qui se répète inlassablement.

Exemple pour une valse lente à 3 temps : La liste est : son 116 – son 115 – son 115 Le tempo est 60 sons par minute.

Cahier des charges :

Les sons seront des sons Midi. Dans le protocole Midi, chaque instrument possède un code.

Pour les instruments de percussion, on utilisera ici les codes instruments suivants :

Son 1 : code 113 : Tinkle Bell	Son 4 : code 116 : Woodblock	Son 7 : code 119 : Synth Drum
Son 2 : code 114 : Agogo	Son 5 : code 117 : Taiko Drum	Son 8 : code 120 : Reverse Cymbal
Son 3 : code 115 : Steel Drums	Son 6 : code 118 : Melodic Tom	Son 9 : code 121 : Guitar Fret Noise

Pour déclarer et activer le flux Midi il faut :

Déclarer dans le main() une variable de type HMIDIOUT et appeler la fonction midiOutOpen() :

```
HMIDIOUT l_hMidiOut_fluxMidi;
midiOutOpen(&l_hMidiOut_fluxMidi, 0, 0, 0, CALLBACK_NULL);
```

} A placer en tout début de main()

Pour jouer un son Midi il faut alors appeler la fonction JouerSonBaR() :

```
JouerSonBaR(113, _hMidiOut_fluxMidi, 70);
```

Joue le son de code 113 (Tinkle Bell) sur le flux midi à 70% du volume total.

Le programme devra demander une séquence de son à jouer. La saisie se fera sous la forme d'un seul entier à 8 décimales. Il est conseillé décomposer cet entier et de placer les 8 entiers dans un tableau.

Exemples :

- Pour la saisie 13314599, la séquence à jouer est :
son1 son3 son3 son1 son4 son5 son9 son9
- Pour la saisie 10015522, la séquence à jouer est :
son1 silence silence son1 son5 son5 son2 son2
- Pour la saisie 13300000, la séquence à jouer est :
son1 son3 son3 silence silence silence silence silence
- Pour la saisie 133, la séquence à jouer est :
son1 son3 son3

ATTENTION :

Entre chaque son, une temporisation doit être insérée pour respecter le tempo.

L'utilisateur devra aussi saisir le tempo (entier) [20 à 120] et le volume [0 à 100].

Le tempo représente le nombre de son à jouer en une minute. Si le tempo saisi n'est pas compris dans la plage spécifiée, le programme doit demander une nouvelle saisie.

Ensuite, le programme doit jouer la séquence en boucle. Pour arrêter la séquence, l'utilisateur devra avoir la possibilité d'appuyer sur la touche **F** ou **Q**. Le tempo ainsi que la séquence jouée doivent être affichés en permanence.

Amélioration 1 :

L'utilisateur devra avoir la possibilité de modifier le tempo pendant que la séquence est jouée : S'il appuie sur la touche **+**, le tempo doit augmenter de 10. S'il appuie sur la touche **-**, le tempo doit diminuer 10. Le tempo doit rester entre 20 et 120.

De la même façon l'utilisateur devra avoir la possibilité de modifier le volume : S'il appuie sur la touche **F**, le volume doit augmenter de 10. S'il appuie sur la touche **Q**, le volume doit diminuer 10. Le volume doit rester entre 0 et 100.

Amélioration 2 : Les saisies des caractères 'H', 'L', '+' ou '-' doivent être prises en compte en temps réel sans attendre la fin de la séquence jouée.

Aide pour ce projet :

```
void main(void) {
    HMIDIOUT l_hMidiOut_fluxMidi; //déclaration du flux Midi nommé l_hMidiOut_fluxMidi
    int l_tabInt_Son[8]={113,120,117,116,119,118,115,114}; //déclaration et initialisation d'une séquence fixe
    int iBcl;
    char l_char_lettre;
    midiOutOpen(&l_hMidiOut_fluxMidi, 0, 0, 0, CALLBACK_NULL); //Ouverture du flux midi l_hMidiOut_fluxMidi

    for(iBcl = 0; iBcl<8;iBcl++){
        //Joue le son l_tabInt_Son[iBcl] sur le flux midi l_hMidiOut_fluxMidi avec un volume à 100% du maximum :
        JouerSonBaR(l_tabInt_Son[iBcl],l_hMidiOut_fluxMidi,100);
        l_char_lettre = getch(); //Attente saisie touche
    }
}
```

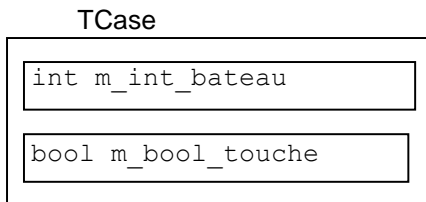
Projet 6 : Bataille navale (D)

La mer (une mer par joueur) où se trouvent les bateaux doit être représentée sous la forme d'un tableau à deux dimensions de type TCase :

	1	2	3	4	5	6
A						
B						
C						
D						
E						
F						

Les bateaux sont 2 bateaux de 3 cases.

Le Type TCase (existe déjà dans PrjInfo2022.h) permet de savoir si un bateau est placé sur la case du tableau et si ce bateau a été touché :



Le membre m_int_bateau permet de savoir si un bateau est placé sur la case. Le numéro stocké dans ce champ représente le numéro du bateau (0 s'il n'y a pas de bateau).

Le membre m_bool_touche permet de savoir si la case a déjà été touchée (0 pour non touché, 1 pour touché).

Dans un premier temps : Un seul joueur. Il faut donc déclarer une mer (1 tableau) et initialiser la mer entière avec des 0 dans tous les membres (utiliser 2 boucles for imbriquées). Pensez à afficher tous les membres pour vérifier.

Puis le programme doit placer aléatoirement les deux bateaux en position verticale ou horizontale. Ils ne doivent pas se chevaucher ni être coupés en deux.

Le joueur doit couler les bateaux en proposant des coordonnées. S'il touche un bateau, le programme indique "Toucher". Si les trois cases du même tableau sont touchées, il indique "Couler".

Quand les 2 bateaux sont coulés, le programme indique "Gagner" et il donne le nombre de tentative, et le temps de jeu.

Amélioration 1 : Une fois que votre programme fonctionne avec un joueur, passez au mode 2 joueurs (nécessite donc 2 tableaux) : Les 2 joueurs jouent à tour de rôle. Chaque joueur place ses bateaux au début.

Amélioration 2 : Un des joueurs est remplacé par une intelligence artificielle...

Aide pour ce projet :

```
void main(void) {
    TCase l_tabEnr_mer[10][10];           //Déclaration de la mer
    int iBclX, iBclY;
    for(iBclX=0;iBclX<10;iBclX++){        //Boucle des abscisses
        for(iBclY=0;iBclY<10;iBclY++){    //Boucle des ordonnées
            l_tabEnr_mer[iBclX][iBclY].m_bool_touche = 0;    //Initialisation à 0
            l_tabEnr_mer[iBclX][iBclY].m_int_bateau = 0;     //Initialisation à 0
            printf("%i ",l_tabEnr_mer[iBclX][iBclY].m_int_bateau); //Affichage
        }
        printf("%n\n");
    }
}
```


Projet 7 : Jeu Simon (F)

Qu'est-ce que le jeu *Simon* ? C'est un jeu pour tester la mémoire de 2 personnes.

- La première personne saisit un caractère (chiffre ou lettre) au clavier, que l'on visualise à l'écran.
- La deuxième personne doit alors saisir le même caractère + un autre caractère. Seul le deuxième caractère est affiché à l'écran. Si la séquence est fausse, le programme indique que la 2^{ème} personne a perdu et la 1^{ère} personne a gagné : le jeu s'arrête.

Si la séquence est juste, le programme autorise la 1^{ère} personne à rejouer : le jeu continue.

- La première personne doit alors saisir le 1^{er} caractère + 2^{ème} caractère + un nouveau caractère. Seul le dernier caractère est affiché à l'écran. Si la séquence est fausse, le programme indique que la 1^{ère} personne a perdu et la 2^{ème} personne a gagné : le jeu s'arrête.

Si la séquence est juste, le programme autorise la 2^{ème} personne à rejouer : le jeu continue.

- La deuxième personne doit alors saisir le 1^{er} caractère + 2^{ème} caractère + 3^{ème} caractère + un nouveau caractère. Seul le dernier caractère est affiché à l'écran. Si la séquence est fausse, le programme indique que la 2^{ème} personne a perdu et la 1^{ère} personne a gagné : le jeu s'arrête.

Si la séquence est juste, le programme autorise la 1^{ère} personne à rejouer : le jeu continue...

Pour ce projet on impose :

- Un tableau de 50 *char* pour stocker la séquence (cela devrait être suffisant).
- 2 offres de jeux sont possibles :
 - Jeu 1 ; on joue contre l'ordinateur. Comme on est sûr de perdre, on affiche seulement le nombre de coups joués en fin de partie.
 - Jeu 2 ; on joue à 2 joueurs : on doit compter le nombre de coups joués par chaque joueurs.

Amélioration 1 : On pourra créer plusieurs niveaux de difficultés dans le jeu contre l'ordinateur en lui faisant faire des erreurs plus ou moins souvent.

Amélioration 2 : On ajoutera par la suite un temps limite pour saisir la séquence.

Projet 8 : Jeu des couleurs (F)

L'ordinateur devra choisir une suite de 10 couleurs parmi blanc, vert, rouge et bleu. Ces couleurs devront s'afficher en ligne pendant 5 secondes :

Puis elles devront disparaître. Ensuite, l'utilisateur devra saisir ces couleurs dans l'ordre, les unes après les autres (0 pour noir, 1 pour bleu, 2 pour vert et 3 pour rouge).

A chaque fois qu'une saisie sera correcte, un son (fonction Beep()) devra être émis, et un point comptabilisé.

A chaque fois qu'une saisie sera incorrecte, un autre son (fonction Beep()) devra être émis, et un point retiré.

A la fin, la séquence tirée au hasard devra réapparaître ainsi que la séquence saisie par l'utilisateur. Le nombre de points devra être affiché ainsi que la durée totale pour entrer la séquence.

Les séquences doivent être placées dans des tableaux.

Amélioration 1 : Une fois que votre programme fonctionne avec un joueur, passez au mode 2 joueurs : Chaque joueur saisit sa solution en une seule fois. Le programme affiche alors le vainqueur et les trois séquences.

Amélioration 2 : L'utilisateur possède 3 secondes pour saisir une couleur. Au-delà, la saisie est comptée fausse, et le programme demande la saisie suivante. Familier

Aide pour ce projet :

```
void main (void){
    int iBcl;
    char l_tabchar_couleur[4]={0x7F,0x10,0x20,0x40};           //Tableau contenant les codes couleurs
    Color(0x0F);                                                //Choix couleur de base
    for(iBcl = 0;iBcl<4;iBcl++){
        Color(l_tabchar_couleur[iBcl]);                       //Choix de la couleur
        printf(" ");                                           // Afficher d'un carré
    }
    Color(0x0F);                                                //retour couleur de base
}
```

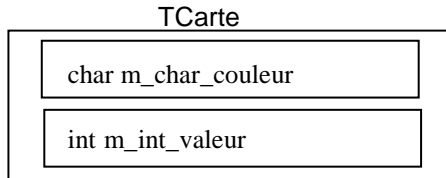
Projet 9 : Pocker (D)

Il s'agit de recréer le célèbre jeu avec des règles simplifiées.

Dans un premier temps on se limitera à un seul joueur. Les combinaisons de cartes gagnantes seront uniquement :

- La paire : 2 cartes de même valeur. Cette combinaison fait gagner 1 point.
- Le brelan : 3 cartes de même valeur. Cette combinaison fait gagner 2 points.
- Le full : 2 cartes de même valeur + 3 cartes de même valeur. Cette combinaison fait gagner 3 points.
- Le carré : 4 cartes de même valeur. Cette combinaison fait gagner 5 points (mais c'est rare).

Vous avez aussi à disposition le type TCarte (existe déjà dans PrjInfo2022.h) :



Vous devrez utiliser un tableau de 32 TCarte (l_tabEnr_jeu32) : Ce tableau est à déclarer et initialiser (voir ci-dessous). Il contient toutes les cartes d'un jeu de 32 cartes classique.

Remarques : m_char_couleur: 'H' pour coeur, 'P' pour pique, 'T' pour trèfle, 'C' pour carreau
m_int_valeur: 7, 8, 9, 10, 11 pour valet, 12 pour dame, 13 pour roi, 1 pour as

Cahier des charges (avec un seul joueur) :

Vous devez déclarer un 2^{ème} tableau (l_tabEnr_jeu5) de 5 TCarte pour stocker les cartes du joueur.

Au démarrage du programme, les 5 cartes doivent être choisies au hasard dans le tableau l_tabEnr_jeu32 et mémorisées dans l_tabEnr_jeu5. Attention, si une carte est choisie une fois, elle ne peut l'être une 2^{ème} fois.

Exemple : Si au premier tirage (effectué sur les 32 TCarte), la TCarte numéro 19 est choisie. Il faut décaler ainsi :

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
---	---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Au tirage suivant, le tirage se fera sur 31 TCarte. Et ainsi de suite...

Il est conseillé de travailler sur un tableau image de façon à réinitialiser ce tableau image avec le tableau de base.

Ces 5 cartes doivent s'afficher en indiquant pour chacune d'elles la valeur et la couleur de la carte.

A ce moment, le joueur doit pouvoir choisir les cartes à retirer de son jeu en entrant les numéros de ces cartes sous la forme d'un seul entier (exemple : 134 pour retirer les cartes 1, 3 et 4). Le programme doit alors procéder à un nouveau tirage pour ces cartes.

La nouvelle main du joueur est de nouveau affichée ainsi que la combinaison gagnante éventuelle. Le nombre de points cumulé est affiché.

Le joueur peut alors choisir de continuer ou quitter (C ou Q au clavier).

Amélioration : Le jeu doit pouvoir se jouer deux. La meilleure combinaison gagne le jeu.

- Conseils :**
- Pour trouver les combinaisons plus facilement, il faut penser à classer le tableau l_tabEnr_jeu5 dans l'ordre des m_int_valeur croissants.
 - Pour vérifier le fonctionnement du programme, vous pourrez remplacer le tirage aléatoire par un tirage non aléatoire donnant une main gagnante au joueur.

Aide pour ce projet :

```
void main (void){
    TCarte l_tabEnr_jeu32[32];           //Déclaration du jeu de 32 cartes
    int iBcl;
    InitJeu32(l_tabEnr_jeu32,32);       //Initialisation du jeu de 32 cartes
    for(iBcl=0;iBcl<32;iBcl++){
        //affichage de la carte iBcl du jeu :
        printf("%c, %i\n",l_tabEnr_jeu32[iBcl].m_char_couleur, l_tabEnr_jeu32[iBcl].m_int_valeur);
    }
}
```

Projet 10 : Jeu du son (N)

- L'ordinateur joue une note de musique au hasard parmi Do, Ré et Mi.
- Le joueur doit répéter cette note (touche **K** pour Do, **L** pour Ré et **M** pour Mi).
- L'ordinateur rejoue la première note et une autre hasard.
- Le joueur doit répète la première et le 2^{ème} note.

Ainsi de suite jusqu'à ce que le joueur se trompe dans la séquence à répéter. L'ordinateur affichera le nombre de coup joué et le temps.

Conseil :

Placer la séquence dans un tableau de 50 cases (cela devrait suffire avant que le joueur se trompe).
Il faut placer une temporisation entre les notes (1 seconde).

Amélioration : Avec 2 joueurs. Les 2 joueurs ont chacun 5 points au départ. Les joueurs jouent l'un après l'autre. Le joueur qui se trompe perd 1 point à chaque erreur. Le premier joueur à 0 perd. On impose un tableau de joueur pour stoker les points.

Aide pour ce projet :

```
void main(void) {
    HMIDIOUT l_hMidiOut_fluxMidi;           //déclaration du flux Midi nommé l_hMidiOut_fluxMidi
    int iBcl;
    char l_char_lettre;
    midiOutOpen(&l_hMidiOut_fluxMidi, 0, 0, 0, CALLBACK_NULL);    //Ouverture du flux midi l_hMidiOut_fluxMidi

    for(iBcl = 60; iBcl<=64;iBcl=iBcl+2){
        //Joue la note iBcl sur le flux midi l_hMidiOut_fluxMidi avec un volume à 100% du maximum :
        //60:Do, 62:Ré, 64:Mi
        JouerSonJeu(iBcl,l_hMidiOut_fluxMidi,100);
        l_char_lettre = getch();             //Attente saisie touche
    }
}
```

Projet 11 : Le morpion (D)

Vous connaissez les règles de ce jeu (sinon allez voir sur Internet).

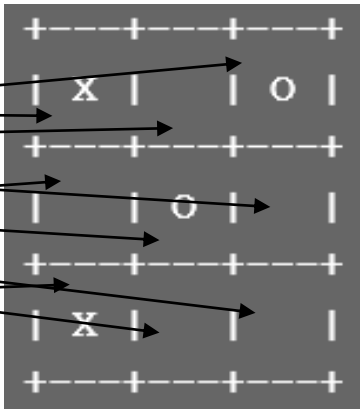
L'affichage de la grille se fera dans la console en utilisant des caractères de base ('X' pour un joueur et 'O' pour l'autre). Chaque joueur joue à tour de rôle tant que 3 caractères ne sont pas alignés. Si un joueur joue sur une case déjà occupée, il perd son tour. Bien sûr c'est votre programme qui doit vérifier que 3 caractères sont alignés.

Le jeu se joue avec le pavé numérique du clavier : En mémoire, la grille est représentée par un tableau de 9 entiers. La case du tableau vaut 0 pour case vide, 1 pour un 'X' (joueur 1) et 2 pour un 'O' (joueur 2) :

Clavier pour jouer :



Grille :



Grille :

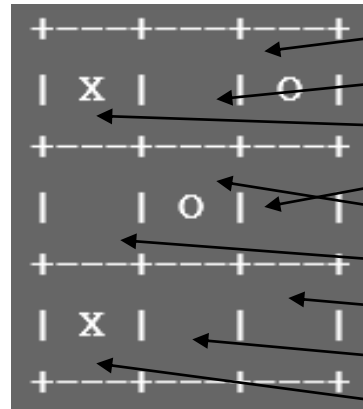


Tableau :

2	[8]
0	[7]
1	[6]
0	[5]
2	[4]
0	[3]
0	[2]
0	[1]
1	[0]

Dans le projet, déclarez le tableau et initialisez-le avec les valeurs ci-contre.

Conseils :

1. Faire dans un premier temps une boucle infinie munie d'un compteur qui compte le nombre de passage dans cette boucle. A chaque passage dans la boucle, si le compteur est paire, afficher "joueur 1 joue", sinon "joueur 2 joue".

Attention, que ce soit le joueur 1 ou le joueur 2 qui joue, le programme doit être le même :

Je ne veux pas :

```
do{
    if((l_int_cpt%2)==0){
        ...
        //code pour le joueur 1
        ...
    }
    else{
        ...
        //code pour le joueur 2
        ...
    }
    l_int_cpt++;
}while((1) ;
```

Je veux :

```
do{
    l_int_joueur = (l_int_cpt%2) + 1 ;
    printf("Joueur %i joue", l_int_joueur);
    ...
    //Code qui doit être le même pour le
    joueur 1 et le joueur 2. C'est la
    variable l_int_joueur qui fait la
    différence.
    ...

    l_int_cpt++;
}while((1) ;
```

2. Ajouter la saisie de l'entier du joueur sans vérifier si la case existe ou si la case est libre.

Si c'est le joueur 1 qui joue affectez la case du tableau correspondant à la saisie avec un 1.

Si c'est le joueur 2 qui joue affectez la case du tableau correspondant à la saisie avec un 2. Affichez la grille.

3. Ajoutez le fait que le joueur perd son tour s'il joue une case qui n'existe pas ou si la case est déjà occupée. Perdre son tour cela veut dire qu'il ne faut pas affecter la case du tableau jouée par le joueur et passer au joueur suivant.

4. Vérification des alignements :

Le joueur 1 gagne lorsque la multiplication des 3 cases d'une ligne, d'une colonne ou d'une diagonale vaut 1.

Le joueur 2 gagne lorsque la multiplication des 3 cases d'une ligne, d'une colonne ou d'une diagonale vaut 8.

Aucun joueur n'a gagné et le jeu est donc terminé si la multiplication de toutes les cases ne vaut pas 0.

Pensez à utiliser un ou plusieurs for pour les tests.

Amélioration : Possibilité de jouer contre l'ordinateur qui possède donc une intelligence artificielle pour ne pas perdre et essayer de gagner. Pour cette partie, demandez des conseils à thierry.suaton@univ-smb.fr

Projet 12 : Son midi (N)

Mode 1 : L'utilisateur doit d'abord saisir le N° correspondant à son instrument (parmi 128 voir page suivante).

Les touches **Q Z S E D F T G Y H U J K** sont affectées à une note (13 touches pour couvrir l'octave 4). Dès que l'utilisateur appuie sur une touche, la note correspondant est jouée avec l'instrument sélectionné. L'utilisateur peut, s'il le souhaite, changer d'instrument en cours en appuyant sur la touche 'm' puis en entrant le nouveau code.

Mode 2 : Dans un tableau (de 100 TNote) il faut enregistrer la séquence jouée :

Appui sur **W** pour démarrer l'enregistrement

Appui sur **X** pour arrêter l'enregistrement

Appui sur **C** pour rejouer la séquence à l'identique.

Conseils :

Pour le mode 1 : Afin d'éviter un switch interminable pour faire correspondre une touche à un code note, vous devez déclarer un tableau de caractères contenant les codes ascii des touches :

```
char cTabCodage[OCTAVE] = {'q','z','s','e','d','f','t','g','y','h','u','j','k'};
```

Lorsqu'une touche est saisie, vous devez chercher l'indice de cette touche dans ce tableau. Le code note est alors cet indice + 60 (octave 4). Pour améliorer, vous pourrez ensuite affecter une autre partie du clavier à la voie 2.

Pour le mode 2 : Il faut utiliser un type structuré TNote (existe déjà dans .h) :

TNote	
char	m_char_note;
int	m_int_duree;

Le champ m_char_note permet de stocker le code de la note ('q','z','s','e'...).

Le champ m_int_duree permet de stocker la durée de la note (avant que la prochaine ne soit jouée).

Aide pour ce projet :

```
void main(void) {
    HMIDIOUT l_hMidiOut_fluxMidi;           //déclaration du flux Midi nommé l_hMidiOut_fluxMidi
    int iBcl;
    char l_char_lettre;
    midiOutOpen(&l_hMidiOut_fluxMidi, 0, 0, 0, CALLBACK_NULL); //Ouverture du flux midi l_hMidiOut_fluxMidi
    //Choisir l'instrument 10 (Glockenspiel) pour le canal 1 du flux midi l_hMidiOut_fluxMidi :
    ChoisirInstrument(1, 10, l_hMidiOut_fluxMidi);

    for(iBcl=60; iBcl<=72; iBcl++){
        //Joue la note iBcl sur le canal 1 du flux midi l_hMidiOut_fluxMidi avec un volume à 100% du maximum :
        JouerNote(1, iBcl, 100, l_hMidiOut_fluxMidi);
        l_char_lettre = getch();           //Attente saisie touche
        //Arrête la note iBcl sur le canal 1 du flux midi l_hMidiOut_fluxMidi
        ArrêterNote(1, iBcl, l_hMidiOut_fluxMidi);
    }
}
```

Utilisation norme Midi : Code des notes :

Octave	Note Numbers											
	C	C#	D	D#	E	F	F#	G	G#	A	A#	B
-1	0	1	2	3	4	5	6	7	8	9	10	11
0	12	13	14	15	16	17	18	19	20	21	22	23
1	24	25	26	27	28	29	30	31	32	33	34	35
2	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59
4	60	61	62	63	64	65	66	67	68	69	70	71
5	72	73	74	75	76	77	78	79	80	81	82	83
6	84	85	86	87	88	89	90	91	92	93	94	95
7	96	97	98	99	100	101	102	103	104	105	106	107
8	108	109	110	111	112	113	114	115	116	117	118	119
9	120	121	122	123	124	125	126	127				

Utilisation norme Midi : Code des instruments

Piano Timbres: 1 Acoustic Grand Piano 2 Bright Acoustic Piano 3 Electric Grand Piano 4 Honky-tonk Piano 5 Rhodes Piano 6 Chorused Piano 7 Harpsichord 8 Clavinet	Chromatic Percussion: 9 Celesta 10 Glockenspiel 11 Music Box 12 Vibraphone 13 Marimba 14 Xylophone 15 Tubular Bells 16 Dulcimer	Organ Timbres: 17 Hammond Organ 18 Percussive Organ 19 Rock Organ 20 Church Organ 21 Reed Organ 22 Accordion 23 Harmonica 24 Tango Accordion
Guitar Timbres: 25 Acoustic Nylon Guitar 26 Acoustic Steel Guitar 27 Electric Jazz Guitar 28 Electric Clean Guitar 29 Electric Muted Guitar 30 Overdriven Guitar 31 Distortion Guitar 32 Guitar Harmonics	Bass Timbres: 33 Acoustic Bass 34 Fingered Electric Bass 35 Plucked Electric Bass 36 Fretless Bass 37 Slap Bass 1 38 Slap Bass 2 39 Synth Bass 1 40 Synth Bass 2	String Timbres: 41 Violin 42 Viola 43 Cello 44 Contrabass 45 Tremolo Strings 46 Pizzicato Strings 47 Orchestral Harp 48 Timpani
Ensemble Timbres: 49 String Ensemble 1 50 String Ensemble 2 51 Synth Strings 1 52 Synth Strings 2 53 Choir "Aah" 54 Choir "Ooh" 55 Synth Voice 56 Orchestral Hit	Brass Timbres: 57 Trumpet 58 Trombone 59 Tuba 60 Muted Trumpet 61 French Horn 62 Brass Section 63 Synth Brass 1 64 Synth Brass 2	Reed Timbres: 65 Soprano Sax 66 Alto Sax 67 Tenor Sax 68 Baritone Sax 69 Oboe 70 English Horn 71 Bassoon 72 Clarinet
Pipe Timbres: 73 Piccolo 74 Flute 75 Recorder 76 Pan Flute 77 Bottle Blow 78 Shakuhachi 79 Whistle 80 Ocarina	Synth Lead: 81 Square Wave Lead 82 Sawtooth Wave Lead 83 Calliope Lead 84 Chiff Lead 85 Charang Lead 86 Voice Lead 87 Fifths Lead 88 Bass Lead	Synth Pad: 89 New Age Pad 90 Warm Pad 91 Polysynth Pad 92 Choir Pad 93 Bowed Pad 94 Metallic Pad 95 Halo Pad 96 Sweep Pad
Synth Effects: 97Rain Effect 98Soundtrack Effect 99Crystal Effect 100Atmosphere Effect 101Brightness Effect 102Goblins Effect 103Echoes Effect 104Sci-Fi Effect	Ethnic Timbres: 105Sitar 106Banjo 107Shamisen 108Koto 109Kalimba 110Bagpipe 111Fiddle 112Shanai	Sound Effects: 113Tinkle Bell 114Agogo 115Steel Drums 116Woodblock 117Taiko Drum 118Melodic Tom 119Synth Drum 120Reverse Cymbal
Sound Effects: 121Guitar Fret Noise 122Breath Noise 123Seashore 124Bird Tweet	Sound Effects: 125Telephone Ring 126Helicopter 127Applause 128Gun Shot	

Projet 13 : Puissance 4 (D)

La grille (une seule pour 2 joueurs) doit être représentée sous la forme d'un tableau à deux dimensions de type int :

1	2	3	4	5	6

Chaque case étant de type int, on pourra stocker le code de chaque case. Le code est le suivant :

- 0 : pas de pion joué sur la case
- 1 : pion joué par le joueur 1
- 2 : pion joué par le joueur 2

Chaque joueur possède ces pions (1 ou 2).

Le joueur 1 choisit sa colonne et son pion doit se placer dans la case libre de la colonne choisie la plus basse. Puis on passe au joueur 2...

La grille doit s'afficher en permanence en indiquant tous les pions déjà joués (2 couleurs différentes : une pour les pions 1 du joueur 1, une autre pour les pions 2 du joueur 2).

A chaque fois qu'un joueur joue, l'ordinateur doit vérifier si le pion joué vient de faire une ligne de 4 pions de même couleur avec les pions joués précédemment (en -, en | ou en /).

Amélioration : Coder un mode l'ordinateur en donnant plus ou moins d'intelligence à l'ordinateur.

Conseils :

1. Faire dans un premier temps une boucle infinie munie d'un compteur qui compte le nombre de passage dans cette boucle. A chaque passage dans la boucle, si le compteur est paire, afficher "joueur 1 joue", sinon "joueur 2 joue".

Attention, que ce soit le joueur 1 ou le joueur 2 qui joue, le programme doit être le même :

Je ne veux pas :

```
do{
    if((l_int_cpt%2)==0){
        ...
        //code pour le joueur 1
        ...
    }
    else{
        ...
        //code pour le joueur 2
        ...
    }
    l_int_cpt++;
}while((1) ;
```

Je veux :

```
do{
    l_int_joueur = (l_int_cpt%2) + 1 ;
    printf("Joueur %i joue", l_int_joueur);
    ...
    //Code qui doit être le même pour le
    joueur 1 et le joueur 2. C'est la
    variable l_int_joueur qui fait la
    différence.
    ...

    l_int_cpt++;
}while((1) ;
```

2. Ajouter la saisie de l'entier du joueur sans vérifier si la colonne existe et s'il y reste une case libre.

Si c'est le joueur 1 qui joue affectez la case du tableau correspondant à la saisie avec un 1.

Si c'est le joueur 2 qui joue affectez la case du tableau correspondant à la saisie avec un 2. Affichez la grille.

3. Vérifier que la colonne existe et qu'il y reste une case libre. Vérifier les alignements.


Aide pour ce projet :

```
void main(void) {
    int l_tabInt_grille[6][6];           //Déclaration de la grille
    int iBclX, iBclY;
    for(iBclX=0;iBclX<6;iBclX++){        //Boucle des abscisses
        for(iBclY=0;iBclY<6;iBclY++){    //Boucle des ordonnées
            l_tabInt_grille[iBclX][iBclY] = 0; //Initialisation à 0
            printf("%i ",l_tabInt_grille[iBclX][iBclY]); //Affichage
        }
        printf("%\n");
    }
}
```


Projet 14 : Le bandit manchot (N)



Celui proposé ici possède 3 rouleaux de 11 figures chacun. On impose un tableau à double dimension pour mémoriser les figures des rouleaux (voir aide ci-dessous).



Le joueur possède 100 points au départ. Il peut choisir de miser 1 ou 3 points.

Lors de l'appui sur , une figure par rouleau doit être choisie au hasard.

Ces trois figures sont affichées.

Le joueur gagne si une des combinaisons (voir ci-dessous) apparaît (peu importe l'ordre). Sinon il perd sa mise. Son nombre de points est alors mis à jour.

Le jeu continue ou s'arrête (saisie de  ou .

Amélioration 1 : Si aucune combinaison n'apparaît lors de l'appui sur , le joueur peut conserver une ou plusieurs figures. Il suffit qu'il l'indique par un entier : Exemple : Il saisit le nombre 23 pour **rejouer** les rouleaux 2 et 3, puis . L'ordinateur rejoue les figures demandées dans le rouleau correspondant mais le joueur perd 1 point (ou 3 selon la mise de départ). Et ainsi de suite jusqu'à ce que le joueur gagne une combinaison ou n'ai plus de points à jouer.

Amélioration 2 : Faire un affichage sympathique lorsque les figures des rouleaux sont choisies au hasard (type <https://www.youtube.com/watch?v=GwgKzR3Lcs>) et faire jouer les musiques suivantes :

PlaySoundA("Joue.wav",NULL,SND_SYNC); //Quand le tirage s'effectue

PlaySoundA("Perdu.wav",NULL,SND_SYNC); //Quand il n'y a pas de combinaison gagnante




PlaySoundA("Gagne.wav",NULL,SND_SYNC); //Quand il y a une combinaison gagnante




Conseils :




Ranger les 3 figures tirées au hasard dans un tableau.

En mode *test*, vous pouvez modifier les rouleaux pour donner plus de chance à une combinaison.




Les rouleaux et les combinaisons :**Les combinaisons :**




   (environ 1.5 chance sur 1000) : gain = mise x 500

   : (environ 3 chances sur 1000) : gain = mise x 250


   (environ 1.5 chance sur 100) : gain = mise x 25


   (environ 7 chance sur 100) : gain = mise x 10


   (environ 9 chances sur 100) : gain = mise x 8


   (environ 16 chances sur 100) : gain = mise x 4

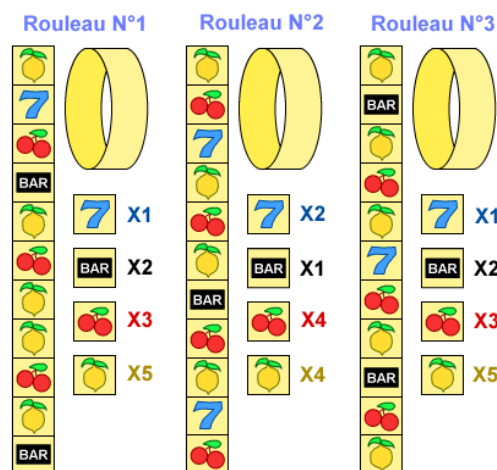
Dans le programme :

 est mémorisé par le caractère '7'

 est mémorisé par le caractère 'B'

 est mémorisé par le caractère 'C'

 est mémorisé par le caractère 'L'



www.casinos-regles-astuces.com

Aide pour ce projet :

```
void main(void) {
    //Déclaration et initialisation des 3 rouleaux :
    char l_tabChar_rouleau[3][11] = {
        {'L','7','C','B','L','C','L','L','C','L','B'},
        {'L','C','7','L','C','L','B','3','L','7','C'},
        {'L','B','L','C','L','7','C','L','B','C','L'};

    int iBclX, iBclY;

    for(iBclX=0;iBclX<3;iBclX++){
        //Boucle des rouleaux
        for(iBclY=0;iBclY<11;iBclY++){
            //Boucle des figures
            printf("%c ",l_tabChar_rouleau[iBclX][iBclY]); //Affichage
        }
        printf("\n");
    }
}
```