Jeu MasterMind
Projet Informatique

F. AIT SALAHT

farah.aitsalaht@u-paris10.fr

Université Paris Ouest, Nanterre La Défense

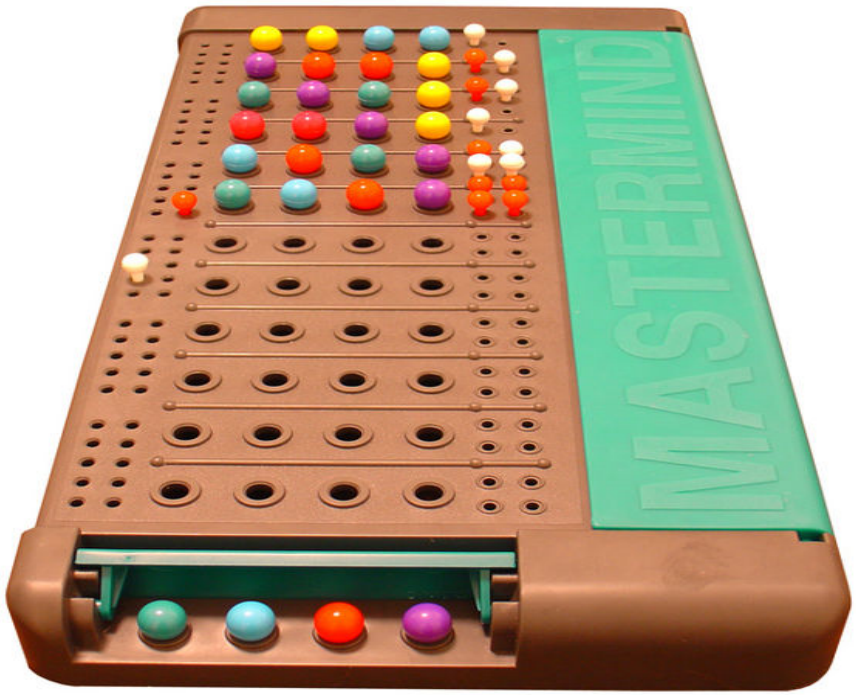
2015-2016

Sommaire

Sommaire

Jeu de Mastermind

► Le Mastermind est un jeu de réflexion entre deux joueurs dont le but est de trouver un code. Il se présente sous la forme d'un plateau ayant 12 rangées de 4 trous pouvant accueillir des pions de couleurs. Le nombre de couleurs est 6 et sont généralement : rouge, jaune, bleu, vert, orange, noir.
► Il y a également des fiches blanches et rouges (ou noirs selon les versions du jeu) qui seront utilisés pour donner des indications.



Règles du jeu

- Un des participants (joueur 1) choisit secrètement 4 boules de couleur parmi l'ensemble des 6 pions de couleurs qu'il place dans un ordre précis
- Le but de l'autre joueur (joueur 2) est de deviner la combinaison qu'à proposé son adversaire
- Pour cela, à chaque tour, le joueur 1 suggère une combinaire afin de trouver la bonne combinaison ou se rapprocher le plus possible de la solution
- Le nombre de rangées pour trouver la solution est limité
- Une fois les pions placés, le joueur 1 indique le nombre de pions de bonne couleur :
  1. **Bien placés** en utilisant les fiches rouges (ou noirs selon les versions du jeu)
  2. **Mal placés** avec les fiches blanches.
- **Fin de la partie :** Deux situations terminent une partie :
  - Le joueur n'a pas réussi à trouver la solution : la partie est perdue.
  - Le joueur a trouvé la solution : la partie est gagnée.

Démarche à suivre

Afin d'implémenter proprement ce jeu et avoir un code (script) et un programme bien lisible et structuré, une décomposition du jeu en un ensemble de fonctions est vivement recommandée.
Nous pouvons distinger les fonctions suivantes :

- **usage() :** affiche un texte informatif sur le jeu;
- **choisirSolution() :** détermine aléatoirement la combinaison secrète;
- **saisirEssai() :** assure la saisie d'une combinaison proposée par le joueur;
- **verifierEssai() :** vérifie si la combinaison proposée par le joueur correspond à la combinaison secrète et détermine le nombre de pions de la bonne couleur bien placés et mal placés;
- **afficherResultat() :** affiche l'état du tour (le numéro de tour, le nombre de tours restant et le nombre de pions de la bonne couleur bien placés et mal placés) et de la manche si elle est finie (en dévoilant la combinaison secrète en cas de défaite du joueur);

Sommaire

Programme - Décomposition du programme

1. **Entête du programme:** Intégrer les bibliothèques à utiliser

```
1 #include <stdio.h>
2 #include <stdlib.h>
```

2. **Intégrer les variables globales utilisées dans le programme :**

```
1 #define maxEssais 10 //Definir le nombre maximal d'essais
      du joueur a 10
2 //Variables globales
3 int nb_essais; //Compteur sur le nombre d'essais effectues
4 int solution[4];//Tableau contenant la solution recherchee
5 int Choix[4];//Tableau reprenant la combinaison proposees
      par le joueur
6 int chiffreM; // Le nombre de chiffre mal places
7 int chiffreB; // Le nombre de chiffre bien places
8 int PartieEnCours; //Vaut 1 si la partie est toujours en
      cours et 0 sinon
```

Fonctions utilisées

1 Afficher un texte informatif sur le jeu

```
1  void usage()
2  {
3      printf(" \t\tBienvenue au jeu MasterMind \t\t \n ");
4      printf("Vous avez %d tentatives pour trouver la combinaison
           secrete\n",maxEssais);
5      printf("Entrer votre combinaison (4 chiffres. 1-6)\n\n");
6  }
```

2 Définir une combinaison secrète (choix aléatoire) Fonction qui génère un code aléatoire dans un tableau

```
1  void  choisirSolution  ()
2  {
3      int i;
4      PartieEnCours = 1;
5      nb_essais = 0;
6      for (i = 0; i < 4; i++) solution[i]=rand()%6 +1;
7  }
```

Fonctions utilisées

3 Entrer la proposition du joueur

Cette fonction permet de demander à l'utilisateur de saisir une combinaison

```
1  void saisirEssai()
2  {
3      do{
4          printf("Tentative %d\t\nPremier nombre :",nb_essais);
5          scanf("%d",&Choix[0]);
6          printf("\nDeuxieme nombre :");
7          scanf("%d",&Choix[1]);
8          printf("\nTroisieme nombre :");
9          scanf("%d",&Choix[2]);
10         printf("\nQuatrieme nombre :");
11         scanf("%d",&Choix[3]);
12     }
13     while ((Choix[0] < 1 || Choix[0] > 6) ||
14             (Choix[1] < 1 || Choix[1] > 6) ||
15             (Choix[2] < 1 || Choix[2] > 6) ||
16             (Choix[3] < 1 || Choix[3] > 6));
17     nb_essais++;
18 }
```

Fonctions utilisées

4 **Vérification de la proposition du joueur** Fonction permettant de compter le nombre de pion bien et mal placé

```
1  void verifierEssai()
2  {
3      int i,j;
4      int SolutionValide[4], ChoixValide[4];
5      chiffreB = 0;
6      chiffreM = 0;
7      for(i=0; i<4; i++){//compte ne nombre d'elements bien place
8          if(solution[i] == Choix[i]){
9              ChoixValide[i] = 1;
10             SolutionValide[i] = 1;
11             chiffreB++;
12         }
13         else{
14             ChoixValide[i] = 0;
15             SolutionValide[i] = 0;
16         }
17     }
18     for(i=0; i<4; i++){//compte le nombre d'elements mal place
19         if(SolutionValide[i] == 0){
20             for(j=0; j<4; j++){
21                 if(i!= j){
22                     if(solution[i] == Choix[j] && ChoixValide[j] == 0){
23                         ChoixValide[j] = 1;
24                         ChoixValide[i] = 1;
25                         chiffreM++;
26                     }
27                 }
28             }
29         }}}
```

Fonctions utilisées

5 Afficher le résultat Fonction permettant d'afficher les combinaisons

```
1  int afficherResultat()
2  {
3      int i;
4      printf("\t\t\t Vous avez %d elements bien places et %d elements mal places\n",chiffreB,chiffreM);
5      if (chiffreB == 4)
6      {
7          printf("Vous avez gagne!!!!\nau bout de : %d essais",nb_essais);
8          PartieEnCours = 0;
9          return (0);
10     }
11     else
12     {
13         if (nb_essais >= maxEssais)
14         {
15             printf("\nVous avez perdu :(\n La solution etait : ");
16             for ( i=0; i<4; i++)
17             {
18                 printf("%d ",solution [i]);
19             }
20
21             PartieEnCours = 0;
22             return (0);
23         }
24     }
25 }
```

Fonction principale et résultat

6 Fonction main

```
1  int main (void)
2  {
3      srand(time(NULL));
4      usage();
5      choisirSolution ();
6
7      while (PartieEnCours)
8      {
9          saisirEssai();
10         verifierEssai();
11         afficherResultat();
12     }
13 }
```

```
                        Bienvenue au jeu MasterMind
  Vous avez 12 tentatives pour trouver la combinaison secrete
Entrer votre combinaison (4 chiffres separe par un espace. 1-6)

Tentative 1     1 2 1 6
                        Vous avez 0 elements bien places et 1 elements mal places
Tentative 2     2 3 4 2
                        Vous avez 3 elements bien places et 0 elements mal places
Tentative 3     2 3 4 1
                        Vous avez 3 elements bien places et 0 elements mal places
Tentative 4     2 3 4 5
                        Vous avez 3 elements bien places et 0 elements mal places
Tentative 5     2 3 4 3
                        Vous avez 3 elements bien places et 0 elements mal places
Tentative 6     2 3 4 4
                        Vous avez 4 elements bien places et 0 elements mal places
Vous avez gagne!!!!
 au bout de : 6 essais
--------------------------------
Process exited after 64.97 seconds with return value 0
Appuyez sur une touche pour continuer...
```

À retenir

Lorsque vous abordez un projet, vous devez prévilégier le processus suivant :

1. Analyser bien l'application demandée : définition des entrées/sorties, des données à manipuler
2. Découper l'application en un ensemble de fonctions et/ou modules avec prévision des tests de validation des fonctionnalités
3. Répartir les fonctions à développer entre chacun si le travail est effectué en groupe
4. Après validation des différentes fonctions utilisées, valider l'ensemble de votre programme

Veillez à commenter votre code pour qu'il soit lisible et compréhensible et organiser correctement votre travail.

Sommaire

Utiliser une bibliothèque graphique pour réaliser vos interfaces de jeux

L'environement qu'on va employer ici utilise plusieurs fichiers : Makefile, graphics.c, graphics.h et couleur.h. Cet environnement est basé sur la librairie graphique SDL : www.libsdl.org.
Comment récupérer la bibliothèque graphique :

1. Télécharger la machine virtuelle *Linux Lubuntu légère 1.6 Go* sur le lien suivant : ***http://www.cartnum.uvsq.fr/telechargements-302338.kjsp?RH=1385031999419&RF=1385032066334.***
2. Installer Virtualbox (voir sur google)
3. Une fois Virtualbox instalée, importer la machine virtuelle *Linux Lubuntu légère 1.6 Go* en allant dans Fichier –> importer

Un ensemble d'exemples sont mis à disposition dans le dossier UVSQ_graphics pour faire ses premiers pas.
Le document IN100_poly_td.pdf reprend en annexe toutes les fonctions disponible dans la bibliothèque graphique.