Introduction à la Programmation

G1: monasse*(at)*imagine.enpc.fr G2: nicolas.audebert*(at)*onera.fr
G3: alexandre.boulch*(at)*onera.fr G4: maxime.ferrera*(at)*onera.fr
G5: laurent.bulteau*(at)*u-pem.fr G6: pierre-alain.langlois*(at)*enpc.fr

— TP #3 —

1 Tableaux

Dans ce TP, nous allons programmer un jeu de Mastermind, où l'utilisateur doit deviner une combinaison générée aléatoirement par l'ordinateur. Le joueur dispose d'un nombre déterminé d'essais. A chaque essai d'une combinaison, l'ordinateur fournit deux indices : le nombre de pions correctement placés et le nombre de pions de la bonne couleur mais incorrectement positionnés.

1.1 Mastermind Texte

1. *Récupérer la solution de départ :*
   Télécharger l'archive `Tp3_Initial.zip` sur la page du cours, la décompresser dans un répertoire faisant apparaître les noms des deux élèves (en cas de binôme) et ouvrir le projet `MasterMind`. Etudier ce projet.
2. *Représenter une combinaison :*
   Nous prendrons ici une combinaison de 5 pions de 4 couleurs différentes. La couleur d'un pion sera représentée par un entier compris entre 0 et 3. Pour une combinaison de 5 pions, nous allons donc utiliser un tableau de 5 entiers.

   ```
   int combin[5]; // tableau de 5 entiers
   ```

3. *Afficher une combinaison :*
   Programmer une fonction permettant d'afficher une combinaison donnée à l'écran. La manière la plus simple de faire consistera à faire afficher les différents chiffres de la combinaison sur une même ligne les uns à la suite des autres.
4. *Générer une combinaison aléatoirement :*
   En début de partie, l'ordinateur doit générer aléatoirement une combinaison à faire deviner à l'utilisateur. Nous allons pour cela utiliser les fonctions déclarées dans le fichier cstdlib , notamment la fonction rand() permettant de générer un nombre au hasard entre 0 et RAND_MAX. Afin d'obtenir un nombre entre 0 et n, on procédera de la manière suivante :

   ```
   x = rand()%n;
   ```

   Pour que la séquence de nombres générée ne soit pas la même d'une fois sur l'autre, il est nécessaire d'initialiser le générateur avec une graine variable. La manière la plus simple de procéder consiste à utiliser l'heure courante. La fonction time() déclarée dans le fichier ctime permet de l'obtenir.
   En fin de compte, la fonction suivante nous permet donc de générer une combinaison :

   ```
   #include <cstdlib>
   #include <ctime>
   using namespace std;

   void genereCombinaison(int combin[5])
   {
       for (int i=0; i<5; ++i)
           combin[i] = rand()%4;    // appels au generateur
   }
   ...
   ```

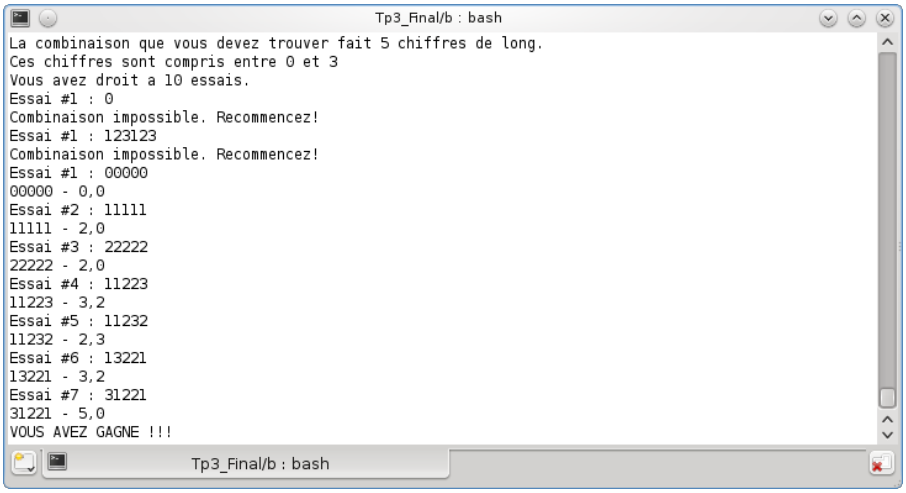

FIGURE 1 – Master mind à la console...

```
srand((unsigned int)time(0)); // initialisation
genereCombinaison(combin);
```

5. *Changer la complexité du jeu :*
   Rapidement, vous allez devenir des experts en Mastermind. Vous allez alors vouloir augmenter la difficulté. Il suffira alors d'allonger la longueur de la combinaison, ou d'augmenter le nombre de couleurs possibles. Cela vous est d'ores et déjà très facile si vous avez pensé à définir une constante globale pour chacune de ces deux grandeurs. Si ce n'est pas le cas, il est grand temps de le faire. Définissez par exemple :

   ```
   const int nbcases = 5;   // longueur de la combinaison
   const int nbcoul = 4;    // nombre de couleurs différentes
   ```

   Reprenez le code que vous avez déjà écrit en utilisant ces constantes. Il est très important de stocker les paramètres constants dans des variables, cela fait gagner beaucoup de temps lorsque l'on veut les modifier.

6. *Saisie d'une combinaison au clavier :*
   La fonction suivante, que nous vous demanderons d'admettre, saisit une *chaîne de caractères* (string) au clavier et remplit le tableau combi[] avec les chiffres que les nbcases premiers caractères de la chaîne représentent.

   ```
   void getCombinaison(int combi[nbcases])
   {
     cout << "Votre essai: ";
     string s;
     cin >> s;
     for (int i=0;i<nbcases;i++)
        combi[i]=s[i]-'0';
   }
   ```

   Attention, le `cin` ne fonctionne pas depuis le terminal intégré de QtCreator (d'ailleurs la fenêtre s'appelle Application *Output*, ne supporte pas l'Input), il faut lui dire d'utiliser un terminal externe. Aller dans l'onglet Projects, rubrique Run et cliquer le bouton "Run in Terminal".

   Dans le cadre de notre Mastermind, il s'agit de modifier cette fonction pour qu'elle contrôle que la chaîne rentrée est bien de bonne taille et que les chiffres sont bien entre `0` et nbcoul−1. L'essai devra être redemandé jusqu'à ce que la combinaison soit valide. On utilisera entre autres la fonction s.size() qui retourne la taille de la chaîne `s` (la syntaxe de cette fonction sera comprise plus tard dans le cours...)

   Une petite explication sur s[i]−'0'. Les caractères (type `char`) sont en fait une valeur numérique, le code ASCII. '0' donne le code ASCII du caractère zéro. On peut le vérifier en faisant par exemple

```
cout << (int)'0' << endl;
```

(Il faut convertir en entier, sinon on lui demande d'afficher un `char`, et donc il affiche le caractère associé au code ASCII, 0!) Cette valeur est 48. Il se trouve que les chiffres de 0 à 9 ont des codes ASCII consécutifs : '0'=48, '1'=49, '9'=57. Le type string de la variable s se comporte comme un tableau de char. Ainsi s[i]−'0' vaut 0 si s[i] est le caractère 0, '1'-'0'=1 si c'est le caractère 1, etc.

7. *Traitement d'une combinaison :*
   Il faudrait maintenant programmer une fonction comparant une combinaison donnée avec la combinaison à trouver. Cette fonction devrait renvoyer deux valeurs : le nombre de pions de la bonne valeur bien placés, puis, dans les pions restant, le nombre de pions de la bonne valeur mais mal placés.
   Par exemple, si la combinaison à trouver est 02113 :
   00000 : 1 pion bien placé (0xxxx), 0 pion mal placé (xxxxx)
   20000 : 0 pion bien placé (xxxxx), 2 pions mal placés (20xxx)
   13133 : 2 pions bien placés (xx1x3), 1 pion mal placé (1xxxx)
   13113 : 3 pions bien placés (xx113), 0 pion mal placé (xxxxx)
   12113 : 4 pions bien placés (x2113), 0 pion mal placé (xxxxx)
   ...
   Pour commencer et pouvoir tout de suite tester le jeu, programmer une fonction renvoyant uniquement le nombre de pions bien placés.
8. *Boucle de jeu :* Nous avons maintenant à notre disposition toutes les briques nécessaires [1], il n'y a plus qu'à les assembler pour former un jeu de mastermind. Pensez par ailleurs à ajouter la détection de la victoire (quand tous les pions sont bien placés), et celle de la défaite (quand un nombre limite d'essais a été dépassé).
9. *Version complète :* Compléter la fonction de traitement d'une combinaison pour qu'elle renvoie également le nombre de pions mal placés.

1.2 Mastermind Graphique

Le jeu de Mastermind que nous venons de réaliser reste malgré tout très peu convivial. Nous allons y remédier en y ajoutant une interface graphique.

1. *Etude du projet de départ :*
   Passer dans le projet `MastermindGraphique` [2]
   Les fonctions graphiques sont déjà définies. Elles fonctionnent selon un principe de division de la fenêtre graphique en lignes. La fonction :

   ```
   void afficheCombinaison(int combi[nbcases], int n);
   ```

   permet d'afficher la combinaison *combi* sur la ligne *n*. Au début du programme, on laisse en haut de la fenêtre graphique autant de lignes libres que le joueur a d'essais pour y afficher le déroulement du jeu. On affiche en bas de la fenêtre graphique un mini mode d'emploi qui résume les correspondances entre touches et couleurs.
2. *Mastermind graphique :*
   Réinsérer dans ce projet les fonctions de génération aléatoire d'une combinaison et de comparaison de deux comparaisons écrites précédemment. Puis reprogrammer la boucle principale du jeu en utilisant l'affichage graphique.
3. *Ultime amélioration :*
   On souhaite pouvoir effacer une couleur après l'avoir tapée, au cas où l'on se serait trompé. Etudier les fonctions

   ```
   int Clavier();
   ```

   et

1. même si la fonction de traitement d'une combinaison est pour l'instant incomplète
2. Sous Visual, penser à le définir comme projet de démarrage pour que ce soit lui qui se lance à l'exécution (son nom doit apparaître en gras dans la liste des projets). Sous QtCreator, sélectionnez-le avec l'icône représentant un écran sur la gauche de la fenêtre. 

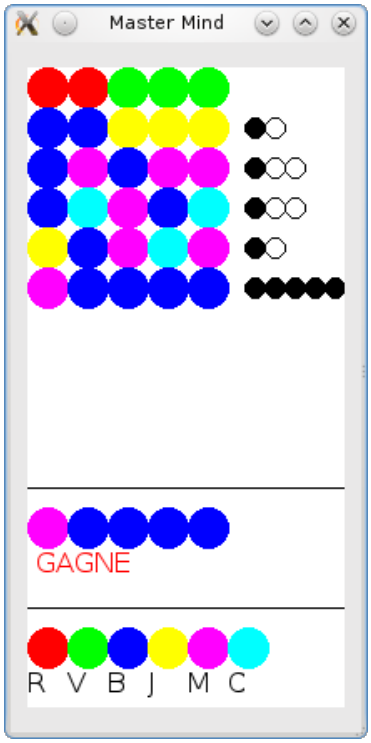

FIGURE 2 – Master mind graphique...

```
void getCombinaison(int[], int);
```

La première prend déjà en compte la touche Retour arrière, mais pas la seconde qui la considère comme une erreur de frappe. Modifier cette dernière en conséquence.