LIFAP1 – TD 12 : Mémory texte … puis graphique

Objectifs : Terminer le codage du Mémory version texte
Commencer à traiter la version graphique.

Version texte

1. Écrire en C/C++ une procédure demandant au joueur de choisir deux cases et d'afficher le contenu de ces deux cases en les restituant dans la grille complète.

```
void affiche_choix (char grille_mot[TAILLE_GRILLE][TAILLE_GRILLE][15], char
solution[TAILLE_GRILLE][TAILLE_GRILLE][15], int l1,int c1, int l2, int c2)
{
  int i,j;
  for(i=0;i<TAILLE_GRILLE;i++)
  {
      for(j=0;j<TAILLE_GRILLE;j++)
      {
        if ( ((i==l1)&&(j==c1)) || ((i==l2)&&(j==c2)))
            cout<<grille_mot[i][j]<<"  ";
        else cout <<solution[i][j]<<"--";
      }
      cout<<endl;
  }
}

void choix_cases(char grille_mot[TAILLE_GRILLE][TAILLE_GRILLE][15], char
solution[TAILLE_GRILLE][TAILLE_GRILLE][15],int &l1,int &c1, int &l2, int &c2)
{
  cout<<"coordonnees de la premiere case";
  cin>>l1>>c1;
  cout<<endl;
  cout<<"coordonnees de la deuxieme case";
  cin>>l2>>c2;
  cout<<endl;
  affiche_choix(grille_mot,solution,l1,c1,l2,c2);
}
```

2. Écrire en C/C++ une fonction de vérification du choix de l'utilisateur. Si les deux cases choisies sont identiques la fonction renverra 0 sinon elle renverra 1.

```
int verification(char grille_mot[TAILLE_GRILLE][TAILLE_GRILLE][15], char
solution[TAILLE_GRILLE][TAILLE_GRILLE][15],int l1,int c1, int l2, int c2)
{
  if (strcmp(grille_mot[l1][c1],grille_mot[l2][c2])==0)
  {
    strcpy(solution[l1][c1],grille_mot[l1][c1]);        // pour version complète du jeu
    strcpy(solution[l2][c2],grille_mot[l2][c2]);        // pour version complète du jeu
    return 0;
  }
    else return 1;
}
```

Leur montrer ici l'utilisation de la comparaison de chaînes.

3. Simuler le jeu à deux joueurs jusqu'à ce que toutes les paires aient été trouvées.

Expliquer que l'on utilise une seconde grille « solution » pour afficher la grille partiellement découverte, et que pour cela, il faut modifier les sous-programmes précédents.

```
void choix_joueur (int &numjoueur)
{
   if (numjoueur==1)
     numjoueur=2;
     else numjoueur =1;
```

```
}

int main (void)
{
  int a1,a2,o1,o2;
  int nb_paires = TAILLE_GRILLE*TAILLE_GRILLE /2;
  int paires_trouvees=0,nb_coup=0;
  int joueur,points_joueur1=0,points_joueur2=0;
  char
memory[TAILLE_GRILLE][TAILLE_GRILLE][15],solution[TAILLE_GRILLE][TAILLE_G
RILLE][15];

  srand(time(NULL));
  init_grille(memory);
  init_grille(solution);
// affiche_grille(memory);
  remplir_grille(memory);
  affiche_grille(solution);
   joueur=rand()%2 + 1;

  while (paires_trouvees<nb_paires)
  {   cout<<"le joueur "<<joueur<<" joue "<<endl;
     choix_cases(memory,solution,a1,o1,a2,o2);
     if (verification(memory,solution,a1,o1,a2,o2)==0)
     {
        paires_trouvees++;
           cout<<"Le joueur "<<joueur<<" marque un point et totalise ";
        if(joueur==1) {
                                points_joueur1++;
                                cout<<points_joueur1<<" points"<<endl;
                                }
                    else
                      {
                                points_joueur2++;
                                cout<<points_joueur1<<" points"<<endl;
                      }


     }
     affiche_grille(solution);
     choix_joueur(joueur);
  }
  if (points_joueur1>points_joueur2)
     cout<<"le joueur 1 a gagné avec "<<points_joueur1<<" points contre
  "<<points_joueur2<<" points."<<endl;
     else if (points_joueur2>points_joueur1)
             cout<<"le joueur 2 a gagne avec "<<points_joueur2<<" points contre
          "<<points_joueur1<<" points."<<endl;
             else cout<<"Egalite entre les deux joueurs !!!"<<endl;
  affiche_grille(solution);
 system("PAUSE");
  return 0;
}
```

Version GrAPiC

Afin de faciliter le codage de l'application en TP sous GrAPiC, nous allons écrire ici quelques sous-programmes.

Dans la version graphique, vous allez utiliser des images plutôt que les mots de la version texte. Les fichiers d'images seront copiés dans le dossier *data* de Grapic et seront nommés de la manière suivante : `Image_Memory_XX.jpg` avec XX un entier compris entre 01 et 18.

Afin de pouvoir positionner aléatoirement les images dans la grille de jeu, il est impératif de pouvoir générer automatiquement le nom du fichier qui sera associé à l'image contenue dans la grille. Pour éviter les doublons (de paires de cartes) il faudra veiller à ce que l'indice choisi de manière aléatoire n'ait pas déjà été sélectionné. On utilisera pour cela un tableau `t_indice_image` tableau de MAX_IMAGES cases de type entier.

1. Ecrire en C/C++ une fonction qui à partir du tableau `t_indice_image` choisit aléatoirement et retourne un indice non utilisé dans ce tableau.

```
int Choisit_num_image (int t_indice_image[MAX_IMAGES])
                {
                   int num_image;
                   do
                   {
                      num_image=rand() % MAX_IMAGES + 1;
                   }
                   while (t_indice_image[num_image-1]!=0);
                   t_indice_image[num_image-1]=1;
                  return num_image ;
                  }
                }
```

2. Ecrire en C/C++ une procédure qui à partir d'un entier passé en paramètre génère le nom du fichier image associé.

```
void Creer_Nom_Fichier (int t_indice_image[MAX_IMAGES] ,char nom_fichier[MAXCHAR])
                {
                   int num_image = Choisit_num_image (t_indice_image) ;
                   char numero[3];

                   strcpy(nom_fichier,"data/memory/Image_Memory_");
                   if (num_image<10)
                   {
                      strcat(nom_fichier,"0");
                      numero[0]='0'+num_image;
                      numero[1]='\0';
                   }
                   else
                   {
                      numero[0]='1';
                      numero[1]=num_image % 10 + '0';
                      numero[2]='\0';
                   }
                   strcat(nom_fichier,numero);
                   strcat(nom_fichier,".jpg");

                }
```

3. Ecrire en C/C++ une procédure qui positionne dans la grille du mémory deux fois l'image dont le nom a été créé précédemment et marque à 1 l'indice utilisé dans le tableau `t_indice_image`.

```
                void place_images(char jeu[MAXX][MAXY][CHMAX])
                {
                   int i,j;
                   int ind_ligne,ind_colonne;
                   char nom_image[MAXCHAR];

                   for (i=0; i<(TGX*TGY)/2; i++)
                   {
                      Creer_Nom_Fichier(nom_image);
```

```
    for (j=0; j<2; j++)
    {
        do
        {
            ind_ligne=rand()%TGX ;
            ind_colonne=rand()%TGY ;
        }
        while (strcmp(jeu[ind_ligne][ind_colonne],"data/memory/vide.jpg")!=0);
        strcpy(jeu[ind_ligne][ind_colonne],nom_image);

    }
}
```