[LC080]

Ecriture de gros programmes
Création et utilisation de bibliothèques

L'écriture de gros programmes en un seul fichier source peut présenter plusieurs inconvénients :

- risque d'espace mémoire insuffisant à l'édit° ,
- efficacité moindre de l' éditeur
- risque d' espace mémoire insuffisant à la compilation,
- compilat° longue · puisqu'elle doit être refaite entièrement après la moindre correction.

Les problèmes d'espaces mémoires se font moins sentir actuellement, car les ordinateurs sont plus facilement équipés de mémoires importantes. Ils peuvent être encore sensibles pour de vraiment gros programmes. Ils peuvent être aussi remplacés par des problèmes de lecture/écriture disques si les fichiers sont traités progressivement ou si on utilise de la mémoire virtuelle.

Ce qu'il ne faut pas faire : l'inclusion de fichier .c

L'inclusion de fichier **.c** en utilisant la directive **#include** permet de supprimer les deux premiers inconvénients (édition), mais ne résout rien pour les deux suivants (compilation). En fait cette possibilité est une fausse solution, qu'il est absolument *impératif* d'éviter.

prog.c

```
typedef enum {FAUX,VRAI} bool;
int a,b,c; float d,e,z;
#include "prog1.c"
#include "prog2.c"
#include "cvideo.c"
main()
{
...
z=fonc3()+fonc5(); clignot();
...
}
```

prog1.c

```
#define CLS {putchar(27);...}
void fonc1(void)
{
...
d=fff(e)+fonc4(a+b); clignot();
...
}
void fonc2(int x,int y)
{
int k; CLS
...
k=fonc5(); invers_video();
...
}
float fff(long w)
{... ... ...}
```

prog2.c

```
float fonc3(void)
{
...
c=fonc4(a)
...
}
int fonc4(int p)
{
...
...
}
float fonc5(void)
{
...
...
}
```

cvideo.c

```
void invers_video(void)
{
...
...
}
void clignot(void)
{
...
...
}
```

Remarquons que le fait de terminer les noms de fichiers inclus par **.h** ne change rien puisque, si ces fichiers contiennent des définitions de variables et/ou de fonctions, ce ne sont pas de vrais *fichiers d'entêtes* (voir plus loin la *bonne solution*).

Une solution convenable : compilations séparées puis édition de liens

Cette solution existe dans la plupart des langages informatiques, et permet même de mélanger les langages. Chaque module peut (en prenant des précautions) être programmé dans le langage que l'on veut : C, pascal, fortran , assembleur. En général on mixe de cette façon un langage évolué et un assembleur.

prog.c

```
        void invers_video(void);
typedef enum {FAUX,VRAI} bool;
int a,b,c; extern. float z;
 static    float d;
 extern    void fonc1(void);
    "   void fonc2(int x,int y);
    "       float fonc3(void);
    "       int fonc4(int p);
    "       float fonc5(void);
main()
{
...
z=fonc3()+fonc5(); clignot();
...
}
```

prog2.c

```
          int a,b,c;
          void fonc1(void);
           int fonc4(int p);
float z;
float fonc3(void)
{
...
c=fonc4(a)
...
}
int fonc4(int p)
{
...
...
}
float fonc5(void)
{
...
...
}
```

prog1.c

```
#define CLS {putchar(27);...}
 extern  int a,b,c;
 static float d,e,fff(long);
 extern  int fonc4(int p);
 extern. float fonc5(void);
void fonc1(void)
{
...
d=fff(e)+fonc4(a+b); clignot();
...
}
void fonc2(int x,int y)
{
int k; CLS
...
k=fonc5(); invers_video();
...
}
static float fff(long w)
{
 ...
 ...
}
```

cvideo.c

```
void invers_video(void)
{
...
...
}
void clignot(void)
{
...
...
}
```

```
cc prog.c prog1.c prog2.c cvideo.o

cc prog*.c cvideo.o

cc prog*.c cvideo.o -o prog
```

prgm de sortie

La bonne solution : compilations séparées avec inclusion de fichiers d'entêtes (.h) puis édition de liens

Dans les fichiers d'entêtes, destinés à être inclus, on pourra trouver :

- des déclarations de fonctions et variables externes,
- des définitions de macros ,
- des définitions de types ,
- des inclusions d'autres fichiers d'entête ,
- etc...

prog.h

```
/* variables définies dans prog.c */
  extern  int a,b,c;
/* variables définies dans prog1.c */
  extern   void fonc1(void);
  extern void fonc2(int x,int y);
/*  variables et fonctions définies
          dans prog2.c */
extern    float z;
  extern   float fonc3(void);
  extern   int fonc4(int p);
  extern   float fonc5(void);
```

cvideo.h

```
#define CLS {......}
...
typedef enum {FAUX,VRAI} bool;
...
#include <stdio.h>
extern   void
        invers_video(void),
                clignot(void),
                surbrill(void);
```

Les déclarations externes pourront alors être faites par des inclusions de ces fichiers d'entêtes :

prog.c

```
# include   "cvideo.h"
# include   "prog.h"
# include   <ctype.h>
int a,b,c;
main()
{
...
z=fonc3()+fonc5(); clignot();
...
}
```

prog1.c

```
# include "cvideo.h"
# include "prog.h"
static  float d,e,fff(long);
void fonc1(void)
{
...
d=fff(e)+fonc4(a+b); clignot();
...
}
void fonc2(int x,int y)
{
int k; CLS
...
k=fonc5(); invers_video();
...
}
static float fff(long w)
{
  :::
}
```

prog2.c

```
# include  "cvideo.h"
# include  "prog.h"
float z;
float fonc3(void)
{
...
c=fonc4(a)
...
}
int fonc4(int p)
{
:::
}
float fonc5(void)
{
...
...
}
```

cvideo.c

```
void invers_video(void)
{
...
...
}
void clignot(void)
{
...
...
}
```