UNIVERSITÉ DE PAU ET DES PAYS DE L'ADOUR | COLLÈGE STEE SCIENCES ET TECHNOLOGIES POUR L'ÉNERGIE ET L'ENVIRONNEMENT

Techniques de programmation - TP nº 4 - Programmation modulaire en langage C

Samson Pierre <samson.pierre@univ-pau.fr>

03/10/2021

Ce TP a pour objectif de vous familiariser avec la programmation modulaire en langage C. Pour chaque exercice, sauf indication contraire, aucune fonction n'est autorisée et les mêmes options de compilation que celles du cours doivent être utilisées.

Exercice nº 1

Reprenez le code de l'exercice nº 4 du TP nº 2. Remarquez que vous pouvez organiser ce code en trois modules : le module de la bibliothèque (`library.c` et `library.h`), le module du livre (`book.c` et `book.h`) et le module principal (`main.c`). Réorganisez donc ce code en modules. Compilez séparément chaque module sans l'édition des liens puis compilez le programme.

Voici le résultat attendu :

```
$ gcc -std=c89 -pedantic -Wall -Werror -g -c -o library.o library.c
$ gcc -std=c89 -pedantic -Wall -Werror -g -c -o book.o book.c
$ gcc -std=c89 -pedantic -Wall -Werror -g -c -o main.o main.c
$ gcc -std=c89 -pedantic -Wall -Werror -g -o main.out library.o book.o main.o
$ ./main.out
("Sciences Library", ("monday", "tuesday", "wednesday", "thursday"), (("The C Programming
Language", ("Brian W. Kernighan", "Dennis M. Ritchie"), "Prentice Hall", 1998, 9780131103627), ("C:
The Complete Reference", ("Herbert Schildt"), "McGraw-Hill Education", 2000, 9780072121247)))
("Novel Library", ("tuesday", "wednesday", "thursday", "friday"), (("Harry Potter and the
Philosopher's Stone", ("J. K. Rowling"), "Bloomsbury", 1997, 9780747532699), ("Harry Potter and the
Chamber of Secrets", ("J. K. Rowling"), "Bloomsbury", 1998, 9780747538493)))
$
```

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```

Exercice nº 2

Reprenez le code de l'exercice précédent. Créez un `Makefile` pour ce programme contenant les règles permettant de compiler chaque module ainsi que le programme. Ajoutez à ce `Makefile` les cibles `all` et `clean`. N'utilisez pas dans ce `Makefile` de variables, de variables automatiques, de cibles spéciales ou encore de règles implicites. Vérifiez que lorsqu'un module est modifié seulement les fichiers dépendant de ce module sont recompilés. Vérifiez que lorsqu'aucun module n'est modifié aucun fichier n'est recompilé.

Voici le résultat attendu :

```
$ make
gcc -std=c89 -pedantic -Wall -Werror -g -c -o main.o main.c
gcc -std=c89 -pedantic -Wall -Werror -g -c -o book.o book.c
gcc -std=c89 -pedantic -Wall -Werror -g -c -o library.o library.c
gcc -std=c89 -pedantic -Wall -Werror -g -o main.out main.o book.o library.o
$ touch library.c
$ make
gcc -std=c89 -pedantic -Wall -Werror -g -c -o library.o library.c
gcc -std=c89 -pedantic -Wall -Werror -g -o main.out main.o book.o library.o
$ make
make: Nothing to be done for 'all'.
$ ./main.out
```

```
("Sciences Library", ("monday", "tuesday", "wednesday", "thursday"), (("The C Programming
Language", ("Brian W. Kernighan", "Dennis M. Ritchie"), "Prentice Hall", 1998, 9780131103627), ("C:
The Complete Reference", ("Herbert Schildt"), "McGraw-Hill Education", 2000, 9780072121247)))
("Novel Library", ("tuesday", "wednesday", "thursday", "friday"), (("Harry Potter and the
Philosopher's Stone", ("J. K. Rowling"), "Bloomsbury", 1997, 9780747532699), ("Harry Potter and the
Chamber of Secrets", ("J. K. Rowling"), "Bloomsbury", 1998, 9780747538493)))
$ make clean
rm -fv *.o *.out
removed 'book.o'
removed 'library.o'
removed 'main.o'
removed 'main.out'
$
```

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```

Exercice n° 3

Créez un programme similaire à celui de l'exercice précédent. Utilisez les variables `CC`, `CFLAGS`, `LDFLAGS`, `LDLIBS` et `RM` dans le `Makefile`.

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```

Exercice n° 4

Créez un programme similaire à celui de l'exercice précédent. Utilisez les variables automatiques `$@`, `$<` et `$^` dans le `Makefile`.

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```

Exercice n° 5

Créez un programme similaire à celui de l'exercice précédent. Utilisez la cible spéciale `.PHONY` dans le `Makefile`. Remarquez que dorénavant, lorsque des fichiers `all` ou `clean` sont présents dans le répertoire de travail, les cibles `all` et `clean` fonctionnent correctement.

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```

Exercice n° 6

Créez un programme similaire à celui de l'exercice précédent. Définissez une règle implicite dans le `Makefile` pour la compilation des modules.

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```

Exercice n° 7

Créez un programme similaire à celui de l'exercice précédent. Supprimez la règle implicite dans le `Makefile`. Remarquez que le `Makefile` fonctionne toujours. C'est normal car une règle intégrée à Make existe déjà pour la compilation des fichiers objets. Remarquez toutefois que cette règle intégrée à Make ne fonctionne pas correctement si le fichier d'en-tête seulement d'un module est modifié.

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```

Exercice n° 8

Créez un programme similaire à celui de l'exercice n° 6. Déclarez dans le module de la bibliothèque, dans le fichier `library.c`, une variable globale statique `library_nlibraries` initialisée à la valeur `0`. Modifiez le corps de la fonction `library_create` pour qu'elle incrémente cette variable à chaque bibliothèque créée. Modifiez le corps de la fonction `library_free` pour qu'elle décrémente cette variable à chaque bibliothèque libérée. Créez dans le module de la bibliothèque la fonction `library_count` sans paramètre qui retourne la valeur de cette variable. Déclarez dans le module du livre, dans le fichier `book.c`, une variable globale statique `book_nbooks` initialisée à la valeur `0`. Modifiez le corps de la fonction `book_create` pour qu'elle incrémente cette variable à chaque livre créé. Modifiez le corps de la fonction `book_free` pour qu'elle décrémente cette variable à chaque livre libéré. Créez dans le module du livre la fonction `book_count` sans paramètre qui retourne la valeur de cette variable.

Affichez dans le module principal la valeur retournée par la fonction `library_count` juste avant et juste après la libération de vos bibliothèques. Affichez dans le module principal la valeur retournée par la fonction `book_count` juste avant et juste après la libération de vos livres.

Vérifiez que grâce à la classe de stockage `static`, il est impossible de faire référence avec la classe de stockage `extern` aux variables `library_nlibraries` et `book_nbooks` respectivement en dehors des fichiers `library.c` et `book.c`. C'est normal car la classe de stockage `static` implique un lien interne de la variable. C'est donc un bon moyen d'encapsuler des variables à l'intérieur d'un module en les rendant privées.

Voici le résultat attendu :

```
$ ./main.out
("Sciences Library", ("monday", "tuesday", "wednesday", "thursday"), (("The C Programming
Language", ("Brian W. Kernighan", "Dennis M. Ritchie"), "Prentice Hall", 1998, 9780131103627), ("C:
The Complete Reference", ("Herbert Schildt"), "McGraw-Hill Education", 2000, 9780072121247)))
("Novel Library", ("tuesday", "wednesday", "thursday", "friday"), (("Harry Potter and the
Philosopher's Stone", ("J. K. Rowling"), "Bloomsbury", 1997, 9780747532699), ("Harry Potter and the
Chamber of Secrets", ("J. K. Rowling"), "Bloomsbury", 1998, 9780747538493)))
number of libraries: 2
number of libraries: 0
number of books: 4
number of books: 0
$
```

Voici les fonctions autorisées pour cet exercice :

```
void free(void *ptr);
void *malloc(size_t size);
int printf(const char *format, ...);
void *realloc(void *ptr, size_t size);
```