

LO22 : Examen final

21 Juin 2005 (2h)

Aucun document autorisé

Pour faciliter la lecture de vos algorithmes (1 point est réservé à la lisibilité), vous devez :

- placer des commentaires explicatifs dans votre code;
- choisir des noms de variables "parlantes", c-à-d différentes de toto, titi, ...;
- respecter une indentation lisible

EXERCICE 1: Questions théoriques (5 points)

1. Citez quatre types du langage C et la taille en octets qu'ils occupent en mémoire.
2. En quelques mots, expliquez le rôle du segment de code d'un programme C compilé.
3. Quelle est la différence entre un type structuré `struct` et un type structuré `union` ?
4. Quel est le contenu de la variable `a` dans le code suivant si le fichier contient les caractères 127 ? Donnez la valeur réelle ou le moyen de la calculer.

```
short a ;  
fread(&a, sizeof(short), 1, FID) ;
```

5. Qu'est ce que DIR ?

EXERCICE 2: Mémoire (6 points)

Nous voulons implanter un système qui nous permettra de vérifier que toutes les zones mémoire allouées ont bien été libérées.

Pour réaliser cette tâche, nous proposons d'écrire les fonctions :

```
void* spy_malloc(long taille_en_octets)  
void spy_free(void* pointeur)
```

qui réalisent les mêmes tâches que `malloc` et `free` et qui mémorisent dans un tableau tous les pointeurs alloués. Ce tableau est du type suivant :

```
struct memory_allocation {  
    void** tab ;  
    unsigned long nb_cellules ;  
}
```

1. Écrivez les deux fonctions `spy_malloc` et `spy_free`.
2. Proposez une solution pour que tous les appels aux fonctions `malloc` et `free` réalisés en dehors de votre code source soient en fait des appels aux fonctions `spy_malloc` et `spy_free`. Vous n'avez pas le droit de modifier les noms des fonctions appellées.

3. Écrivez un fichier `.h` et un fichier `.c` contenant votre bibliothèque.

EXERCICE 3: Système de complétion (6 points)

Écrivez la fonction `char* read_filename(char* directory)` qui permet de saisir à partir de l'entrée standard le nom d'un fichier se trouvant dans le répertoire passé en paramètre.

Votre fonction devra impérativement proposer un mécanisme de "complétion". Le mécanisme de complétion permet à l'utilisateur, lorsqu'il appuie sur la touche tabulation (`'\t'`, code ASCII 9) de voir le système tenter de compléter sa saisie en fonction des noms de fichiers réellement présents sur le disque.

Par exemple, supposons que le répertoire `/home/moi/tmp` contienne les fichiers `bonjour.doc`, `bonte.pdf` et `alsa.tex`. L'appel à votre fonction sera : `read_filename("/home/moi/tmp")` ;. Si l'utilisateur entre au clavier le caractère `'b'` puis appuie sur la touche tabulation, alors le système devra compléter la chaîne saisie par `"on"` car toutes les occurrences des noms de fichiers commençant par `b` se poursuivent par les deux caractères `on`.

Pour vous aider dans votre rédaction, vous pourrez utiliser les caractères `'\n'` (retour-chariot) et `\b` (recule le curseur d'un caractère gauche).