Devoir surveillé du 26 mai 2007

CSH : Initiation au C et au shell

Première année

Tous documents interdits. Les exercices sont indépendants. La correction tiendra compte de la qualité de la rédaction et de la présentation. Barème approximatif.

★ **Exercice 1.** On souhaite écrire une fonction qui renvoie la position de la première occurence d'un caractère (paramètre `c`) dans une chaîne de caractères (paramètre `str`), ou -1 si le caractère n'est pas présent. Voici le prototype de la fonction :

```
int premier_cara(char c, char *str) ;
```

▷ **Question 1. (2 pts)** Écrire cette fonction

▷ **Question 2. (1 pt)** Écrire une fonction `main` permettant de tester votre travail. Exemple d'exécution :

```
1 $ ./occurence o Bonjour
2 La première occurence de 'o' dans 'Bonjour' est à la position 2.
3 $
```

★ **Exercice 2. (2 pts)** La fonction `strdup` de la bibliothèque standard permet de duppliquer une chaîne : elle réserve un emplacement mémoire suffisament grand pour contenir la chaîne, puis effectue la copie.

▷ **Question 3.** Réimplémentez `strdup` sans utiliser de fonction de la bibliothèque standard à part malloc.

★ **Exercice 3. (2 pts)** Écrire un script shell `est_arrive utilisateur` qui teste toutes les dix secondes si l'`utilisateur` en question s'est connecté, et s'arrête avec un message adéquat si c'est le cas. `utilisateur` est un nom de login UNIX. On rappelle que la commande `who` affiche les utilisateurs actuellement connectés sous la forme :

```
1 rouyerj2 pts/3        May 24 11:01
2 cario2   pts/46       May  2 08:59
3 costa15  pts/71       May  3 10:39
4 alexand2 pts/11       May 23 10:51 (arc.loria.fr)
```

★ **Exercice 4. (2 pts)** Écrire un script shell `danger dir` qui cherche tous les sous-répertoires de `dir` dans lesquels tout le monde peut écrire. 1 pt de bonus si votre solution mène une recherche récursive.

★ **Exercice 5. (2pts)** Écrivez un fichier de commandes de nom ***sauvegarde.sh*** permettant de :

- Créer un répertoire de nom ***Archives*** dans votre répertoire personnel (s'il n'y existe pas déjà) ;
- Copier dans ce répertoire ***Archives*** tous les fichiers exécutables qui se trouvent dans le répertoire dont le nom est donné en argument.
  Si le répertoire donné en argument n'existe pas, afficher un message d'erreur.

★ **Exercice 6. (3pts)** On suppose que l'on dispose d'un fichier calepin.txt, contenant des noms et des numéros de téléphone rangés de la manière suivante :

calepin.txt

```
1 DUPONT Jean,05.61.75.18.47,21/08/1975,jean.dupont@free.fr
2 MARTIN Yvonne,02.23.34.45.56,26/02/1977,yvonne.martin@cegetel.fr
3 ...
```

▷ **Question 4. ($\frac{1}{2}$pt)** Écrire un script (ou une commande) qui effectue la recherche des personnes s'appelant DURAND et qui restitue leur addresse éléctronique.

▷ **Question 5. ($\frac{1}{2}$pt)** Écrire un script (ou une commande) qui compte les gens ayant 20 ans cette année.

▷ **Question 6. (2 pts)** Écrire un script (ou une commande) classant les personnes de la plus vieille à la plus jeune.
Indication : `sort -n` trie les lignes passées en argument dans l'ordre numérique du premier argument.

NOM : PRÉNOM :

★ **Exercice 7.** Voici un programme C. Il compile et s'exécute normalement sans erreur.

```c
#include <stdlib.h>
#include <stdio.h>
// Définition des fonctions
void f1 ( int a, int *b ) {
  a = *b;
}
void f2 ( int *b, int c ) {
  *b = c;
}
void f3 ( int *a, int c ) {
  f4 ( &a, c );
}
void f4 ( int **b, int a ) {
  **b = a;
}
```

```c
// Fonction principale
int main () {
  int x = 5, y = 7, z = 9;
  f1 ( x, &y );
  printf ( "x = %d, y = %d, z = %d\n", x, y, z );
  f2 ( &x, y );
  printf ( "x = %d, y = %d, z = %d\n", x, y, z );
  f3 ( &y, z );
  printf ( "x = %d, y = %d, z = %d\n", x, y, z );
  return 0;
}
```

▷ **Question 7. (3 pts)** Qu'affiche ce programme lors de son exécution ?

★ **Exercice 8. QCM** Répondez sur la feuille fournie. Il peut y avoir plusieurs cases valides par ligne ; les réponses fausses seront pénalisées.

▷ **Question 8. (2 pts)** Quel est le type de chacune des variables dans cet extrait de programme ?

```c
int *a,b;
char **c, *d[12];

typedef struct {
  char *marque;
  int nb_places;
  float consomation;
} voiture_t, *voiture;

voiture_t e,f[12];
voiture   g,h[12];
```

variable	entier	flottant	structure	pointeur	tableau	(écriture invalide)
a	☐	☐	☐	☐	☐	☐
b	☐	☐	☐	☐	☐	☐
c	☐	☐	☐	☐	☐	☐
d	☐	☐	☐	☐	☐	☐
e	☐	☐	☐	☐	☐	☐
e.nb_places	☐	☐	☐	☐	☐	☐
e->nb_places	☐	☐	☐	☐	☐	☐
f	☐	☐	☐	☐	☐	☐
f.nb_places	☐	☐	☐	☐	☐	☐
f->nb_places	☐	☐	☐	☐	☐	☐
f[0].nb_places	☐	☐	☐	☐	☐	☐
f[0]->nb_places	☐	☐	☐	☐	☐	☐
g	☐	☐	☐	☐	☐	☐
g.nb_places	☐	☐	☐	☐	☐	☐
g->nb_places	☐	☐	☐	☐	☐	☐
h	☐	☐	☐	☐	☐	☐
h.nb_places	☐	☐	☐	☐	☐	☐
h->nb_places	☐	☐	☐	☐	☐	☐
h[0].nb_places	☐	☐	☐	☐	☐	☐
h[0]->nb_places	☐	☐	☐	☐	☐	☐

▷ **Question 9. (1 pt)**

- Si dans un fichier *makefile* on trouve la ligne suivante : `toto : tutu`
  - ☐ cela veut dire qu'il faut reconstruire `toto` chaque fois que `tutu` change
  - ☐ cela veut dire qu'il faut reconstruire `tutu` chaque fois que `toto` change
- Dans un fichier d'entêtes (.h), on peut trouver :
  - ☐ Des définitions de fonctions
  - ☐ Des prototypes de fonctions
  - ☐ Des définitions de types
- L'expression `if [ -x toto -a -d toto ]` teste si :
  - ☐ on a le droit d'exécuter et d'effacer `toto`
  - ☐ on a le droit d'exécuter `toto` et si sa taille est non nulle
  - ☐ `toto` est un répertoire dont on peut lire le contenu
  - ☐ `toto` est un répertoire dans lequel on peut entrer