Examen d'informatique - TP

2005 - 2006

Durée : 1H Sujet n°1

Avant-propos (3 points)

- Tout document autorisé (surtout l'aide en ligne ou `man`).
- **(0.5)** Créez un répertoire `Examen_TP_login` (où `login` est votre nom de login) sur votre compte dans le répertoire `Info`.
- **(1)** Rendez ce répertoire accessible uniquement à vous et aux utilisateurs ne faisant pas partie de votre groupe.
- Déplacez dans ce répertoire.
- **(1)** Créez un fichier `login.txt` sur lequel vous aurez les droits en lecture et écriture, votre groupe n'aura aucun droits et les autres auront les droits en lecture uniquement.
- **(0.5)** Mettez dans ce fichier les commandes qui vous ont permis d'arriver jusqu'ici.
- Toutes les réponses aux questions de l'examen (commandes et résultats) devront apparaître dans ce fichier que vous recopierez à la fin de l'examen dans le répertoire `/home/shared/InfoGene/AGRAL/soumission1`.
  Cet examen sera évalué en fonction de ce qui sera présent dans ce répertoire.

Solutions :

mkdir Examen_TP_login
chmod 707 Examen_TP_login `ou` chmod 705 Examen_TP_login `ou` chmod g-rx Examen_TP_login ...
cd Examen_TP_login
touch login.txt `ou` kate login.txt ...
chmod 604 login.txt `ou` chmod g-r login.txt ...

Commandes UNIX (4 points)

Question 1 (2)

En une ligne de commande, créez dans `Examen_TP_login` un sous-répertoire `repExam`, en lui attribuant les droits suivants :

- Tous les droits pour l'utilisateur et les autres.
- Aucun droit pour le groupe.

N'oubliez pas de mettre la commande que vous avez tapée dans le fichier `login.txt`.
Solutions :

mkdir -m 707 repExam

Question 2 (2)

On veut savoir quel fichier d'en-tête (bibliothèque) inclure dans un programme C pour pouvoir utiliser la fonction `scanf`. Quelle commande utiliser ?
N'oubliez pas de mettre les commandes que vous avez tapées dans le fichier `login.txt`.
Solutions :

man scanf

Programmation en C (11 points)

(1) Récuperer le fichier `/home/shared/InfoGene/AGRAL/prog1.c`
Solutions :

cp /home/shared/InfoGene/AGRAL/prog1.c .

Question 3

- **(1)** Compilez le programme `prog1.c` et créer un exécutable `test1`.
- **(0.5)** Exécutez le, le résultat vous paraît-il correct ?
- **(0.5)** Recompilez le programme avec l'option `-Wall`, que constatez vous ?
- **(1)** Corrigez le programme en conséquence.

Solutions :

- gcc prog1.c -o test1
- Exécuter : ./test1
  Il ne se passe rien alors que le programme devrait afficher les valeurs de p.
- gcc -Wall prog1.c -o test1
  L'option Wall permet d'afficher les warning (avertissement).

  prog1.c:7: attention : parenthèses suggérées autour de l'affectation utilisée comme valeur de vérité

  A la ligne 7 on a une affectation au lieu d'une comparaison dans la condition d'arrêt du while.

  prog1.c:5: attention : unused variable q

  La variable q n'est pas utilisée

Une solution possible :

```
#include <stdio.h>

int main()
{
    int p,q;
    p=5;
    while(p = 0)
    {
        printf("%d\n",p);
        p = p-1;
    }
    return 1;
}
```

Question 4 (7 points)

On souhaite écrire un programme qui affiche toutes les tables de multiplications de 0 à 9.

- **(3)** Écrire un programme `mult1.c` qui affiche la table de multplication de 2, comme montré dans l'exemple suivant :

```
----- table de 2 ------
2 x 0 = 0
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18
```

- Reprendre le programme précédent, l'enregistrer dans un fichier `mult2.c`.
- **(4)**Modifier `mult2.c` pour qu'il affiche toutes les tables de multiplications de 0 à 9.

Solutions :

mult1.c

```
#include <stdio.h>

int main()
{
    int i;
    int a = 2;

    printf("----- table de %d -----\n",a);
    for(i=0 ; i<10 ; i=i+1)
    {
        printf("%d x %d = %d\n", a, i, a*i);
    }
    return 1;
}
```

mult2.c

```
#include <stdio.h>

int main()
{
    int i;
    int a;

    for(a=0; a<10; a=a+1)
    {
        printf("----- table de %d -----\n",a);
        for(i=0 ; i<10 ; i=i+1)
        {
            printf("%d x %d = %d\n", a, i, a*i);
        }
    }
    return 1;
}
```

Question bonus(+2 points)

Proposer une commande pour enregistrer le résultat du programme `mult2.c` dans un fichier `multiplication.txt`.

Solutions :

./mult2 > multiplication.txt

FIN (2 points)

Créer une archive compressée `login.tgz` avec vos fichiers `login.txt, prog1.c, mult1.c, mult2.c, multplication.txt`.
Copier cette archive dans le répertoire `/home/shared/InfoGene/AGRAL/soumission1`.
Solutions :

```
tar czvf login.tgz login.txt prog1.c mult1.c mult2.c multplication.txt
cp login.tgz /home/shared/InfoGene/AGRAL/soumission1
```