TP 0 : Introduction

Gilles Peskine

4 octobre 2002

Les sujets, corrigés, etc., de TD et de TP seront placés au fur et à mesure sur ma page web : `http://para.inria.fr/~peskine/enseignement/deug-2002`.

1 Utilisation des ordinateurs

1.1 Introduction

Les ordinateurs de la salle de TP tournent sous FreeBSD, une variante d'Unix. Vous aurez essentiellement affaire à l'interface KDE.

Avant de pouvoir travailler sur une machine, il faut vous identifier. La machine demande un nom d'utilisateur (`login:`) et un mot de passe (`password:`). Dans la salle de TP, il y a un compte commun appelé `me` sans mot de passe.

Le principal logiciel à connaître est l'éditeur de textes `kwrite`, qu'il est vivement recommandé d'utiliser pour écrire les programmes.

Vous aurez aussi besoin de taper des commandes Unix. Pour cela, lancer une fenêtre *shell* (sous KDE, cliquer sur l'icône représentant un coquillage devant un écran d'ordinateur).

1.2 Utilisation de `kwrite`

Regarder dans le menu les commandes proposées. Les plus utiles sont dans le menu « Fichier » (« Nouveau » pour créer un fichier, « Ouvrir » pour éditer un fichier existant, « Enregistrer » pour écrire le fichier modifié sur le disque dur).

`kwrite` peut afficher les programmes C(−−) en couleurs. Pour cela, utiliser l'entrée « Activer la surbrillance » dans le menu option, choisir « C » ou « C++ ».

1.3 Le shell Unix

C'est un programme qui permet de taper des commandes qui sont exécutées dès qu'on appuie sur la touche « entrée ». Voici quelques commandes utiles.

`ls -l`	afficher la liste des fichiers dans le répertoire courant
`cp` *`fichier-original nom-de-la-copie`*	faire une copie d'un fichier
`mv` *`ancien-nom nouveau-nom`*	renommer un fichier
`rm` *`fichier`*	supprimer définitivement un fichier
`mkdir` *`nom-de-répertoire`*	créer un nouveau répertoire dans le répertoire courant
`cd` *`répertoire`*	changer de répertoire courant
`cd ..`	aller au répertoire parent du répertoire courant
`rmdir` *`répertoire`*	supprimer un répertoire vide
`grep` *`mot fichier1 fichier2 ...`*	chercher un mot dans un ou plusieurs fichiers

1.4 Utilisation des disquettes

Les disques des salles de TP sont régulièrement effacés. Aussi il n'est pas possible de conserver vos fichiers après la fin du TP. Si vous souhaitez garder une trace de votre travail, vous pouvez **apporter des disquettes**.

Bref guide d'utilisation des disquettes :

`mdir a:`	affiche la liste des fichiers sur la disquette
`mcopy toto a:`	copie le fichier `toto` du disque dur vers la disquette
`mcopy a:toto .`	copie le fichier `toto` de la disquette vers le disque dur
`mcopy toto a:`	copie le fichier `toto` du disque dur vers la disquette
`mdel a:toto`	efface le fichier `toto` de la disquette

1.5 Développement C−−

L'objectif des TP sera d'écrire des programmes en C−−, un sous-ensemble de C++. Une fois le texte du programme écrit et **enregistré** dans un fichier `toto.C`, le cycle de production du programme sera le suivant :

1. **Vérifier** que le fichier est un programme C−− correct. Ceci se fait en tapant la commande suivante dans une fenêtre shell :

   ```
   g-- <toto.C
   ```

2. Si `g--` n'a pas signalé d'erreur, **compiler** le programme, c'est-à-dire produire du code exécutable à partir du code source :

   ```
   g++ -O -Wall -W -o toto toto.C
   ```

3. Si `g++` n'a pas signalé d'erreur, **exécuter** le programme en tapant la commande suivante :

   ```
   ./toto
   ```

Attention à bien faire la différence entre majuscules et minuscules.

S'il y a des erreurs signalées, utiliser `kwrite` pour les corriger, et recommencer le cycle vérification-compilation-exécution. Astuce : dans le shell, on peut utiliser la flèche vers le haut pour rappeler les commandes précédentes.

2 Exercice : afficher « Bonjour ! »

Taper ce programme dans un fichier nommé `bonjour.C`. Lancer `g--` pour le vérifier, puis `g++` pour le compiler, et enfin l'exécuter.

```
#include <iostream.h>
int main () {
    cout << "Bonjour !\n";
}
```

3 Exercice : puissance entière

```
#include <iostream.h>
// Calcul de la puissance entière d'un réel
float puissance (float r, int n) {
    int i;
    float p = 1;
    for (i=1; i<=n; i++) p = p * r;
    return p;
}

int main () {
    float a;
    int n;

    cout << "Élévation à une puissance entière.\n";
```

```
    cout << "Donnez un nombre réel : "; cin >> a;
    cout << "Donnez la puissance : "; cin >> n;

    if (n<0) cout << "La puissance doit être positive.\n";
    else cout << a << " puissance " << n << " = " << puissance (a,n) << endl;
}
```

4 Exercice : soustraction de deux nombres en base 10

```
#include <iostream.h>

int main () {
    int a, b, c, d; /* les entrees */
    int e, f;       /* les sorties */
    cout << "Programme qui effectue l'operation ab - cd\n avec a, b, c, d des chiffres \n";
    cout << "Introduisez le deux nombres, chiffre par chiffre :\n";

    /* lecture de a */
    do {
        cout << "a (0..9) = ";
        cin >> a;
    } while ((a < 0) || (a > 9)); /* fin do */
    /* lecture de b */
    do {
        cout << "b (0..9) = ";
        cin >> b;
    } while ((b < 0) || (b > 9)); /* fin do */
    /* lecture de c */
    do {
        cout << "c (0..9) = ";
        cin >> c;
    } while ((c < 0) || (c > 9)); /* fin do */
    /* lecture de d */
    do {
        cout << "d (0..9) = ";
        cin >> d;
    } while ((d < 0) || (d > 9)); /* fin do */

   /* algorithme de soustraction */
    cout << "Le resultat de " << a << b << " - " << c << d << " = ";
    if ((a > c) || ((a==c) && (b >= d))) {
        if (b >= d) {
            f = b - d;
            e = a - c;
        } else {
            f = (10 + b) - d;
            e = a - 1 - c;
        } /* fin if */
    } else {
        if (d >= b) {
            f = d - b;
            e = c - a;
        } else {
            f = (10 + d) - b;
            e = c - 1 - a;
```

```
        } /* fin if */
        cout << "-";
    } /* fin if */
    cout << e << f << "\n";
}
```

5 Ressources

Sur l'utilisation d'Unix :

- `http://www.eleves.ens.fr/tuteurs/` Unix très bien expliqué ; recommandé même si certaines parties sont spécifiques à l'ENS. Lire notamment le glossaire.
- `http://www.grbb.polymtl.ca/syst_info/cours_unix.html` Bien, mais un peu vieux.
- `http://www.kde.org/fr/index.php` Le site francophone de KDE.

En outre, Unix dispose d'une documentation en ligne assez fournie. La commande `man` affiche le manuel d'une commande, par exemple « `man ls` » L'aide en ligne de KDE est accessible par le menu « Aide ».