Algorithmique - TP1

IUT 1ère Année

16 septembre 2012

1 Environnement de Compilation

Nous travaillerons sous le système d'exploitation Linux. Pour la programmation C++, il sera nécessaire de :

- créer un répertoire appelé algorithmique (dans votre répertoire documents), dans lequel vous construirez plusieurs répertoires (TP1, TP2, etc.), chacun correspondant à une feuille de TP.
- utiliser un éditeur de code source, comme gedit ou emacs (le dernier existe dans la plupart des environnements).
- utiliser une fenêtre terminal ouverte afin de lancer des commandes en ligne.

Lorsque vous aurez terminé une feuille de TPi, vous archiverez le répertoire TPi avec vos codes sources, et enverrez le fichier compressé sur votre compte Moodle.

Note : tous les documents (cours, TD et TP) sont accessibles à la fois sur Moodle et Google documents. Pour ce dernier, l'adresse d'accès est :

`https://docs.google.com/folder/d/0BzrjfYzjlac-NXRzdHphZm1vVE0/edit`

1.1 Ecriture d'un programme C++

Le code source d'un programme C++ est sauvegardé dans un fichier dont l'extension est .cc ou .cpp. Pour vous familiariser avec votre éditeur, commencez par écrire le programme (un grand classique) suivant :

Listing 1 – helloWorld.cpp

```
#include <iostream>
using namespace std;

int main()
{
cout << "Hello world" << endl;
}
```

1.2 Compilation d'un programme C++

La compilation d'un programme C++ inclut plusieurs étapes

1. **prétraitement** : le pré-processeur transforme le code source (ascii) en un code prétraité (toujours ascii) selon les directives qui commencent par #. Dans le cas du fichier helloWorld.cpp, le pré-processeur remplace la commande #include <iostream> par le contenu du fichier iostream installé sur le système. Ce fichier iostream contient les déclarations des opérations d'affichage et de lecture.
2. **compilation** : le compilateur transforme le code prétraité (ascii) en code objet (binaire), en utilisant plusieurs analyses (lexicale, syntaxique et sémantique) et en terminant (pour les compilateurs modernes) par une phase d'optimisation (selon le type de processeur utilisé par la machine).
3. **édition de lien** : l'éditeur de lien intègre au code objet toutes les bibliothèques (libraries) contenant le code objet des fonctions utilisées par le programme (par exemple l'opérateur ≪) et retourne ainsi un code exécutable du programme.

Pour nos programmes, nous utiliserons le compilateur **g++**. Notre programme helloWorld sera compilé par la ligne de commande suivante :

g++ -o helloWorld helloWorld.cpp

L'exécutable final est lancé en tapant ./helloWorld.

2 Exercices du TP1

Exercice 1. Ecrire un programme perimetreCercle permettant d'afficher le périmètre d'un cercle dont le rayon est donné par l'utilisateur.

Exercice 2. Ecrire un programme heuresMinutesSecondes permettant de convertir un nombre entier (représentant une quantité de secondes) et heures, minutes et secondes.

Exercice 3. Ecrire un programme distanceEntrePoints permettant de calculer la distance entre deux points dont l'abscisse et l'ordonnée sont entrées par l'utilisateur.

Exercice 4. Ecrire un programme sontEnCollision permettant de déterminer si l'intersection de deux cercles sont en collision (intersection non vide). Pour chaque cercle, le rayon et l'abscisse et l'ordonnée du centre sont entrés par l'utilisateur.

Exercice 5. Ecrire un programme sontParalleles permettant de déterminer si deux segments définis par leurs extrémités sont parallèles. Pour chaque segment, l'abscisse et l'ordonnée de chaque extrémité sont entrés par l'utilisateur.

Exercice 6. Ecrire un programme calculette permettant de simuler une calculette pour les nombres entiers : le programme lit en entrée deux nombres entiers et un opérateur arithmétique, et affiche en sortie le calcul de l'opération. Les opérateurs sont $+, -, *, /, \%$.

Exercice 7. Construire un programme factorielle permettant de calculer la factorielle d'un nombre entier.

Exercice 8. Construire un programme conversionBinaire permettant de convertir un nombre en base 2.

Exercice 9. Construire un programme PGCD permettant de calculer le PGCD de deux nombres entiers.

Exercice 10. Construire un programme solveur permettant de résoudre une équation du second degré :
- Données : les coefficients a, b et c de l'équation $ax^2 + bx + c = 0$
- Résultat : le nombre et la valeur des solutions réelles de l'équation