Langage compilé
langa pré té

JM Routoure r l'ingénieur
Universit lormandie

commentaire pour l'année prochaine : approche bcq trop rapide : passer au moins deux séances sur cette partie : programme plus ambitieux en C

Pourquoi programmer

- Un ordinateur seul avec son système d'exploitation ne sait rien faire => nécessité de faire tourner des programmes pour
  - faire des taches bien particulières : traitement de texte, dessin, CAO...
  - automatiser des taches
  - ...

- Nécessité de programme qui peuvent être de 2 types : compilés ou interprétés

programme compilé

Compilation
= traduction

Fichier texte

ne dépend pas du système d'exploitation

Fichier binaire contenant Code binaire compréhensible par le microprocesseur

dépend du système d'exploitation et du matériel

Exemple de langage de programmation basé sur un compilateur : C, C++, java,
Intérêt : rapidité et optimisation du code
Désavantage : complexité

programme interprété

Code source

Fichier texte

ne dépend pas du système d'exploitation

Interprétation = chaque ligne correspond à une commande qui est exécuté en "lisant" le fichier texte

Exemple de langage de programmation basé sur un interpréteur : shell, BASIC, langage de macro
Intérêt : simplicité
Désavantage : lenteur

Travail à effectuer

- “hello world” en C et en shell (*)

  Réaliser un programme qui affiche hello world et s’arrête !

- Le même programme mais avec une une fenêtre et un bouton en C et shell

(*) le shell est l’interpréteur de commande utilisé pour lancer les différents programme à partir du terminal.

Structure d'un programme source C

```
#include <bibliotheque>

int main(){
 ;
return (0);
}
```

insertion des bibliothèques

partie principale

programme à écrire dans un éditeur de texte

compilation : `gcc -o mon_executable fichier_source.c`

exécution : `./mon_executable`

“hello world” en C

```
#include <stdio.h>
#include <stdlib.h>

int main(){
 printf(“Hello world\n”);
return (0);
}
```

insertion des bibliothèques

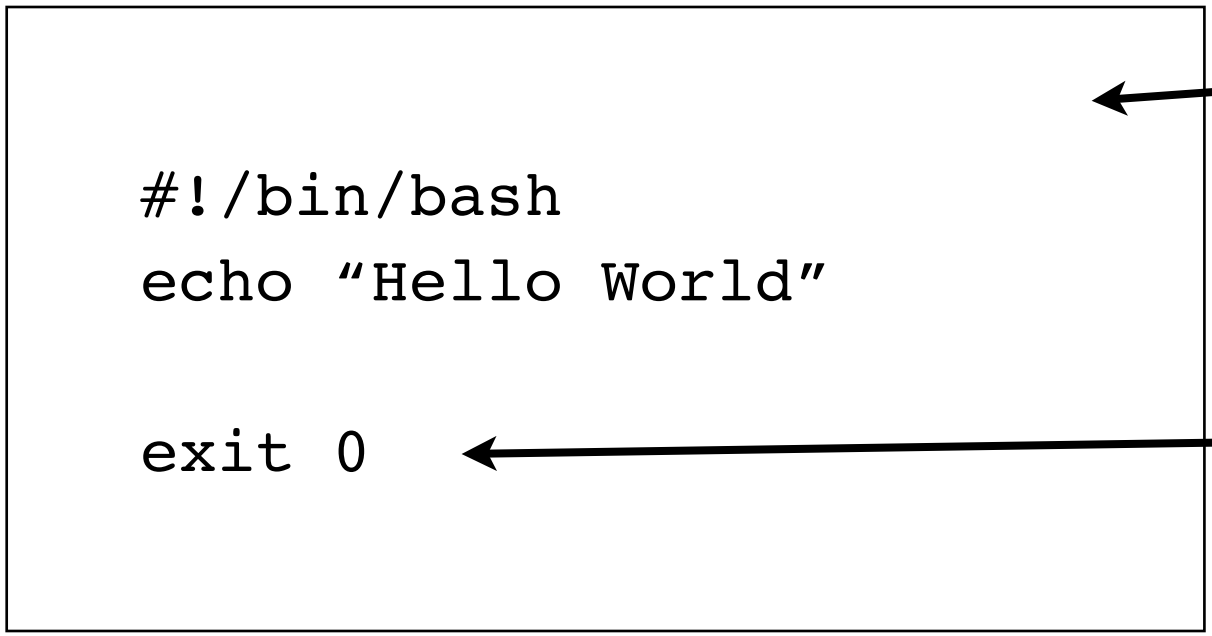
partie principale

programme à écrire dans un éditeur de texte

compilation : `gcc -o mon_executable fichier_source.c`

exécution : `./mon_executable`

Structure d'un programme shell



lancement d'interpréteur de comma,de

partie principale

programme à écrire dans un éditeur de texte

rendre exécutable le script : `chmod uoa+x mon_script.sh`
ou `chmod 755 mon_script.sh`
exécution : `./mon_script`

Un peu plus compliqué

- programme C : mettre dans un fichier deux colonnes X Y pour générer sin(x)/x entre -10 et 10
- création d'un fichier gnuplot permettant de representer les données créées. Générer un postscript du graphique
- création d'un fichier latex intégrant la figure et des commentaire
- création d'un script lançant le programme C, gnuplot, latex, qui transforme le dvi en postscript et qui affiche le résulat.
-

générer sin(X)/X en C

- Dans le corps du programme

```
#include <stdio.h>
#include <stdlib.h>

int main(){
int i;
float x;
float y;
for (i=0;i<35;i++){
     x= (i*20.0/34-10);
     y=sin(x)/x;
     if (x==0) y=1;
     printf("%f %f  \n",x,y);
     }
return (0);
```

compil : gcc -o exe -lm toto.c

Mettre le résultat dans un fichier et cree le fichier gnuplot

- On execute le programme et on envoit le résultat dans un fichier en utilisant la redirection unix :“>”

```
./exe > results.txt
nedit results.txt
```

- Création du fichier gnuplot : `nedit result.plt &`

```
plot “results.txt” using 1:2 with lines
set term postscript lw 3
set output “results.ps”
replot
```

- Création du fichier postscript : `gnuplot result.plt ; ls; gv result.ps`

Creation du fichier latex; compilation latex, postscript final

```
\documentclass{article}
\usepackage{graphicx}
\title {essai de creation automatique}
\author{L1}
\begin{document}
\maketitle
\includegraphics[width=7cm, angle=-90]{results.ps}
\end{document}
```

Compilation dans le shell et postscript final :

```
latex mon_fichier.tex
xdvi mon_fichier.dvi
dvips -o final.ps mon_fichier.dvi
```

script complet : complet.sh

```
#!/bin/bash
./mon_executable > results.txt
gnuplot result.plt
latex mon_fichier.tex
dvips mon_fichier.dvi -o final.ps
gv final.ps
```

Modification et éxecution du script

```
chmod 755 complet.sh
./complet.sh
```


Université de Caen
Basse - Normandie

Démonstration de script plus compliqué Compilation d'un logiciel libre à montrer !