Université de Nice-Sophia Antipolis — avril 2016
Polytech/Xidian — Durée: 1h30

CPP
Travaux Pratiques – Séance n° 8

Le préprocesseur s'applique dans une phase qui précède celle de la compilation proprement dite, passe sur tout le texte source du programme et traite toutes les directives qui lui sont destinées. Ces directives sont introduites par un dièse (#). Nous avons déjà vu les directives #**define**, pour définir une constante, et #**include**, pour inclure le contenu d'un fichier.

Pour mettre en évidence le fonctionnement réel, nous allons jouer avec les options `--save-temps` et `-P` de `gcc`.

1) Reprenez un fichier de la séance précédente qui contient un #**define** et un #**include**. Compilez-le avec les deux options précisées ci-dessus. Décrivez les fichiers produits.

2) À quoi sert l'option `-P` ? Quelle information enlève cette option ?

3) Décrivez le comportement exact du #**define** et du #**include**. Quelles informations sont perdues après le passage du préprocesseur ?

4) Écrivez, compilez, exécutez et expliquez le comportement du programme suivant `prog.c` :

```
#include <stdlib.h>
#include <stdio.h>
int main(void) {
  printf("%s %d\n", __FILE__, MAX);
  return EXIT_SUCCESS;
}
```

```
gcc -DMAX=300 -o prog prog.c && ./prog
```

5) Que se passe-t-il si `MAX` est aussi défini dans le programme ? Vérifiez.

6) Écrivez un programme qui affiche sur la sortie standard les valeurs des macros prédéfinies `__DATE__`, `__TIME__`, `__FILE__`, `__TIMESTAMP__`, `__LINE__`, et `__func__`.

7) Écrivez la *macro* `MAX2(x,y)` qui donne le maximum de `x` et `y`, puis écrivez la *fonction* `max2(int x, int y)` qui donne le maximum de `x` et `y`.

8) Écrivez la *macro* `MAX3(x,y,z)` qui donne le maximum de `x y` et `z`, puis écrivez la *fonction* `max3(int x, int y, int z)` qui donne le maximum de `x`, `y`, et `z`.

9) Écrivez un programme qui teste vos macros et fonctions. Testez le code suivant avec `x=3`, `y=2` et `z=1`, et expliquez les résultats :

```
printf("max3(%d,%d,%d)=%d\n", x, y, z, max3(x++,y,z));
printf("MAX3(%d,%d,%d)=%d\n", x, y, z, MAX3(x++,y,z));
```

10) Écrivez un programme qui affiche *C ansi* si le compilateur est ansi, qui affiche `C ansi/ISO C99` s'il est iso c99, et `C non standard` dans les autres cas. Vous devrez utiliser les directives de pro-

grammation conditionnelle et les macros prédéfinies `__STRICT_ANSI__` et `__STDC__`. Vous compilerez votre programme successivement avec les options de compilation `-ansi -pedantic`.

11) Écrivez un programme qui déclare un tableau `tab` de `SIZE` réel **double** si la constante `SIZE` est supérieure ou égale à 1024. Cette constante sera définie dans le fichier d'en-tête `param.h`. D'autre part, si la constante `DEBUG` est définie, votre programme affichera sur la sortie standard la valeur de `SIZE`.