Pratique du C Les directives au préprocesseur Types composé Définition de nouveaux types V-2 (21-03-2012) Pratique du C préprocesseur Types composés Définition de nouveaux types Les directives au V75 (21-03-2012) Pratique du C Les directives au préprocesseur Types composé Définition de nouveaux types Les directives au

Pratique du C Les directives au préprocesseur Types composés Définition de nouveaux types

Licence Informatique — Université Lille 1
Pour toutes remarques : Alexandre.Sedoglavic@univ-lille1.fr

Semestre 5 — 2012-2013

www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012)

ution de texte Pratique du C Les directives au

Substitution de texte

Le préprocesseur permet de définir des macros constantes et des fonctions sur la base de la substitution de chaîne de caractères.

- macros sans paramètres: #define A 20 (sans rien ajouter). Attention à l'usage du point virgule (;)
- ▶ on peut supprimer une macro par #undef A.

Remarques:

- manipulation purement syntaxique;
- toujours utile de parenthéser les paramètres;
- ▶ imbrication possible des macros;
- pas de blanc entre max et la parenthèse ouvrante;
- pas d'effet sur les chaînes de caractères constantes ;
- ▶ si la macro nécessite plusieurs lignes, utiliser le '\'.

www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012)

Directives conditionnelles

Il est possible de conditionner la compilation par :

► l'insertion optionnelle de code

#if expression_constante lignes à insérer si expression_constante vraie #endif

#ifdef identificateur lignes à insérer si identificateur est défini #endif

- un test de non définition : #ifndef;
- L'usage de l'alternative #if expression_constante lignes à insérer si expression_constante vraie #else lignes à insérer si expression_constante fausse

#endif

#ifdef identificateur lignes à insérer si identificateur est défini #else lignes à insérer si identificateur non défini #endif Pratique du C
Les directives au préprocesseur
Types composés
Définition de

Pratique du C Les directives au

préprocesseur

Types composé Définition de

nouveaux types

Les directives au

préprocesseur Types composés Définition de

nouveaux types

Les directives au

préprocesseur

préprocesseur

Les directives au préprocesseur

nouveaux types

Le type union

Définition de nouveaux types

Les champs de lettres binaires Le préprocesseur permet d'inclure dans le code source des fichiers texte complets.

Deux types d'inclusion de fichiers d'entête :

- 1. #include <file.h> : recherche du fichier file.h
 - dans les répertoires spécifiés par l'option -I du compilateur;
 - dans le répertoire de la librairie standard (/usr/include).
- 2. #include "file.h" : recherche du fichier file.h
 - dans le répertoire du fichier qui fait l'inclusion;
 - comme précédemment ensuite.

Ceci permet d'inclure des prototypes de fonctions, des macros, etc.

www.fil.univ-lille1.fr/~sedoglay/C/Cours04.pdf

Macro avec paramètres : attention aux effets latéraux

Considérons l'exemple classique : #define max(a,b) a>b?a:b.

Supposons que les paramètres soient des expressions incluant des opérateurs de priorité inférieur à > et ? (max(x=y , ++z) par exemple).

Le résultat est x = (y > ++z ? x=y : ++z) ce qui n'a pas grand rapport avec ce que l'on attendait. Ainsi, on a tout intérêt à définir la macro plus précisément : #define max(a,b) (((a)>(b))?(a):(b)).

Mais même dans ce cas, on doit bien remarquer que l'évaluation de cette macro implique une double incrémentation de z qui n'est pas explicite dans l'appel à cette macro.

www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf

Un petit exemple :

```
#ifdef ERREUR /* Attention \'a l'utilisation des */#define SQR(x) x * x /* param\'etres et aux effets */#else /* lat\'eraux */#define SQR(x) ((x) * (x)) #endif a=SQR(4+5); t[i]=SQR(t[i++]);
```

Une macro peut être déclarée depuis le shell lors de la compilation :

```
% gcc -D ERREUR fichiersource.c
```

et ainsi conditionner la compilation du code.

On peut aussi interrompre la compilation

```
#ifndef MAMACRO
#error "MAMACRO inconnue"
#endif
```

Pratique du C Les directives au Macro prédéfinie du préprocesseur préprocesseur Types composé Définition de Il existe un certain nombre de macro prédéfinies : nouveaux types ► __**FILE**__ correspond au nom du fichier source; Les directives au ► __FUNCTION__ correspond au nom de la fonction contenant la ligne courante dans le code; ► __LINE__ correspond à la ligne courante dans le code; ► __DATE__ correspond à la date du préprocessing ; ► __TIME__ correspond à l'heure du préprocessing ; Par exemple % nl preprocessing.c % gcc -E preprocessing.c 1 int main (void) { int main (void) { 2 int a = __LINE__ ; int a = 2; 3 return 0; return 0: 4 } ARKARIATE TOOR V75 (21-03-2012) www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012) Pratique du C La compilation et l'exécution du code suivant : préprocesseur Types composés Définition de #include<stdio.h> nouveaux types void foo(void) __attribute__ ((constructor)) ; void bar(void) __attribute__ ((destructor)) ; Les directives au int old(int) __attribute__ ((deprecated)) ; void foo(void) { printf("%s ",__FUNCTION__); return ; } void bar(void) { printf("%s ",__FUNCTION__); return ; } int toinline(int dummy) { return dummy+1 ; } int old(int dummy) { int res ; res = dummy++ ; return res ;} int main (void) { int tmp ; tmp = old(1) ; tmp = toinline(1) ; printf("%s ",__FUNCTION__) ; retur donnent: % gcc foo.c; ./a.out; echo \$? foo.c: In function main: foo.c:9: warning: old is deprecated (declared at foo.c:8) foo main bar 2 4 D > 4 B > 4 E > 4 E > 9 Q P V75 (21-03-2012) www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012) Pratique du C Les directives au Une structure est le regroupement de plusieurs variables de préprocesseur types différents dans une même entité. Types composés Définition de nouveaux types ▶ cet objet est composé d'une séquence de membres de types divers; chaque membre porte un nom interne à la structure; Le type structures ▶ le type des membres peut être quelconque (imbrication); les membres sont stockés de manière contiguë en mémoire ; déclaration : struct identificateur_du_modèle type liste_identificateur_de_membre; type liste_identificateur_de_membre; };

Attributs de fonctions

Pratique du C Les directives au

préprocesseur

Types compos Définition de

nouveaux types

Les directives au

Pratique du C

Les directives au

préprocesseur Types composés

Définition de

Les directives au

Pratique du C

Les directives au

préprocesseur

Types composés Définition de

nouveaux types

Le type structures

préprocesseur

nouveaux types

Le comportement vis à vis des fonctions du compilateur GCC peut être finement manipulé par le biais de la macro à paramètre __attribute__ ((param)). Par exemple, on peut utiliser comme paramètre :

- deprecated afin de lancer un avertissement si la fonction correspondante est utilisée ;
- constructor afin de lancer la fonction correspondante avant le main ;
- destructor afin de lancer la fonction correspondante après le main;
- ▶ always_inline afin d'inliner la fonction correspondante quelque soit l'option d'optimisation utilisée.

Ces fonctionnalités ne sont pas standardisées. Consultez la documentation pour plus d'informations.

```
www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf
```

Usage avancé (e.g. assert.h)

On peut convertir l'argument d'une macro en une chaîne de caractères en utilisant l'opérateur #.

```
Ceci permet des utilisations du type :
```

```
#include<stdio.h>
#define STRING(x) #x
#define ASSERT(x) if(!x) \
printf("%s not true in file: %s, line: %d, %function %s\n", \
STRING(x),__FILE__,__LINE__,__func__);
int main (void){
        int i = 3:
        ASSERT(i==0)
        return 0 ;
}
qui, si le source est essai.c produit, comme résultat :
% gcc essai.c ; ./a.out
% i==0 not true in file: essai.c, line: 8, function main
(à utiliser en phase de conception durant laquelle le message
est pertinent).
                                    <ロト 4個ト 4 差ト 4 差ト 差 め Q @
                                    www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf
```

- déclaration de variable d'un type structure : struct identificateur_de_modèle liste_identif_variable;
- définition et déclaration simultanées de variables : struct identificateur_de_modèle type liste_identificateur_de_membre type liste_identificateur_de_membre ; } liste_identificateur_de_variable ;
- le nommage de la structure est alors facultatif;
- > accès à un membre : opérateur . de sélection de champs
- identificateur_de_variable . identificateur_de_membre:

```
struct mastructure {
      char o :
      int six :
};
struct mastructure mavariable;
mavariable.o='o'; mavariable.six = 6;
                               ID + 48 + 48 + 3 - 990
```

```
Pratique du C
Les directives au
                Une spécificité du compilateur gcc
  préprocesseur
                 La norme ISO ne permet pas de faire de l'initialisation des
 Types composé
Définition de
                 structures lors de leurs déclarations.
 nouveaux types
                 Mais le compilateur gcc prévoit tout de même cette
                 possibilité :
Le type structures
                  struct complexe {
                    int re ;
                    int im :
                  } foo = {
                    .im = 2,
                     .re = 1
                 Plus canoniquement, l'initialisation peut se faire en donnant
                 la liste entre { } de constantes :
                    int re :
                    int im ;
V75 (21-03-2012)
 préprocesseur
Types composés
Définition de
Le type structures
                 struct adresse
                   int num = 36:
                };
V75 (21-03-2012)
  Pratique du C
 Les directives au
  préprocesseur
 Types composé
Définition de
                      différents des objets de type et de taille différents;
 nouveaux types
                   la manipulation de différents types de données dans un
                      même espace mémoire.
                 La manipulation des unions est semblable à celle des
Le type union
                 structures :
                   syntaxe similaire à celle des structures :
                            union identificateur_de_modèle_d'union
                            type liste_identificateur_de_champs ;
                            type liste_identificateur_de_champs ;
```

```
struct complexe {
  } foo = { 1, 2 } ; /* il faut respecter l'ordre */
                                         www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012)
Ne pas confondre C et ses héritiers
                                                                           préprocesseur
Types composés
                                                                           Définition de
                                                                           nouveaux types
 Attention : C n'est pas un langage orienté objet et donc, il
 n'y a pas de constructeur en C.
                                                                          Le type structures
 Il n'y a pas d'initialisation "générique" associée à un type.
 Le code suivant n'est pas du C valide :
   char rue[40] = "Quai des Orf\'evres";
   long int code = 75001;
   char ville[20] = "Paris" ;
                                          (D) (@) (E) (E) (D) (O)
                                         www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012)
 Un type union permet :
                                                                           Pratique du C
                                                                          Les directives au
                                                                           préprocesseur
   ▶ de définir une variable qui peut contenir à des moments
```

Exemple de représentation en mémoire d'une structure

Pratique du C Les directives au

préprocesseur Types composés Définition de

nouveaux types

Le type structures

Types composé Définition de

nouveaux types

Le type union

```
struct adresse {
 int num:
                            .globl bibi
 char rue[40];
 long int code;
                                    .align 32
 char ville[20];
                                    .type bibi,@object
                                    .size
                                           bibi,120
                           bibi:
struct personne {
                                    .string "Moi"
 char nom[20];
                                    .zero 16
 char prenom[25];
                                    .string "Idem'
 int age:
                                    .zero
 struct adresse adr;
                                    zero
                                           .3
} bibi = {
                                    .long
                                           100
  .nom = "Moi".
                                    .long 39
  .prenom = "Idem",
                                    .string "Publique"
  .age = 100,
                                    .zero 31
  .adr.num = 39.
                                    .long 59000
  .adr.rue = "Publique",
                                    .string "Lille"
  .adr.ville = "Lille",
                                    .zero 14
  .adr.code = 59000 };
                               www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf
```

Copie et affectation d'une structure comme un tout

Contrairement aux tableaux, l'affectation

```
#include "les_definitions_des_transparents_precedents"
struct personne bobo;
int main(void){
 bobo = bibi ;
 return 0 ;
```

est possible et provoque une copie physique des données de l'espace mémoire associé à bibi dans celui associé à bobo.

En conséquence, on peut :

- passer des structures en argument de fonction (copie);
- utiliser une structure comme valeur de retour de
- mais C étant un langage de bas niveau, les structures ne se comparent pas.

www.fil.univ-lille1.fr/~sedoglay/C/Cours04.pdf

Exemple de représentation en mémoire d'une union

Les champs potentiels sont stockés de manière superposé en mémoire.

```
union nombre{
                               .globl bar
                                        .data
 int entier;
                                        .align 4
                                        .type bar,@object
  struct complexe {
                                        .size bar,20
   float re ;
   float im ;
                                        .long
 } comp_var ;
                                        .zero
                                        .text
  char symbol[20] ;
} bar = { .entier = 999 } ;
```

accès à un champs :

liste_identificateur_de_variable;

identificateur_de_variable.identificateur_de_champs

```
Pratique du C
Les directives au
                                                                                        Pratique du C
Les directives au
                                                                                                           Syntaxe : type-énuméré :
               Exemple d'affectation en mémoire d'une
                                                                                                                ⇒ enum identificateur { liste-d-énumérateurs }
                                                                                         préprocesseur
Types composés
Définition de
  préprocesseur
 Types composés
Définition de
               union
                                                                                                             liste-d-énumérateurs :
 nouveaux types
                                                                                         nouveaux types
                                                                                                                     ⇒ liste-d-énumérateurs option énumérateur
                                                        .data
                 union nombre{
                                                        .size
                                                                 bar,20
                                                                                                             énumérateur :
                                                 bar:
                                                                                                                     ⇒ identificateur
                  int entier ;
                                                                   999
                                                          .long
                                                                                                                          identificateur = expression-constante
                                                           .zero
                                                                   16
Le type union
                  struct complexe {
                                                 foo:
                                                          .zero

    type dont les valeurs possibles font partie des

                                                                                        Type énuméré
                    float re ;
                                                           .text
                    float im ;
                                                 .globl main
                  } comp_var ;
                                                 main:
                                                                                                                identificateur dans énumérateur : constante entière ;
                                                          movl
                                                                   bar, %eax
                                                                                                                ▶ nom d'un identificateur : distinct d'une variable
                  char symbol[20];
                                                          movl
                                                                   %eax. foo
                                                          movl
                                                                   bar+4, %eax

    valeur entière nulle au départ et incrémentée pour

                } foo,bar={.entier=999};
                                                          movl
                                                                   %eax, foo+4
                                                                                                                  chaque nouvel identificateur;
                                                          movl
                                                                   bar+8, %eax
                                                                                                                ▶ spécifier une valeur (expression-constante).
                                                          movl
                 int main(void)
                                                                   %eax, foo+8
                                                          movl
                                                                   bar+12, %eax
                                                                                                        enum {VRAI, FAUX} test=FAUX; /* contraire aux
                                                          movl
                                                                   %eax, foo+12
                   foo=bar :
                                                                                                                                              conventions du C */
                                                          movl
                                                                   bar+16, %eax
                                                                                                        enum mois_m { jan=1, feb=2, mar, avr, may, jun, jul,
                                                          movl
                                                                                                                       aug, sep, oct, nov, dec};
                    return 0 ;
                                                                   %eax, foo+16
                                                                                                        enum mois_m mavariable ;
                                                       www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012)
V75 (21-03-2012)
                                                                                                                                               www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf
 Pratique du C
                                                                                                       Types définis par l'utilisateur
 préprocesseur
Types composés
                                                                                         préprocesseur
Types composés
 Définition de
                                                                                         Définition de
                                                                                         nouveaux types
 nouveaux types
                 enum {VRAI,FAUX} test=FAUX; .globl test
                                                                                                           ajoute un nom désignant un type existant;

    lisibilité : utilisé pour les structures complexes ;

                 int main(void){
                                                         .align 4
                  test = VRAI ;
                                                         .type test,@object
                                                                                                           portabilité : paramétrer un programme (size_t);
                  return 0 ;
                                                         .size
                                                                 test,4
                                                                                                           syntaxe : identique à celle d'une variable
Type énuméré
                                                         .long
                                                                                                                    typedef type identificateur_de_type
                                                         .text
                                                                                        Définition de
                                                                                                           ne crée pas un nouveau type, plutôt un synonyme.
                                                                                        nouveaux types
                                                .globl main
                                                         .type main, @function
                                                                                                        Par exemple, la déclaration du nouveaux type entier :
                                                main:
                                                                                                        typedef int entier;
                                                        movl
                                                                 $0. test
                                                                                                        permet les déclarations :
                                                                                                        entier i,j=2;
                                                        10 × 40 × 45 × 45 × 5 × 90 0
                                                                                                                                               4 D F 4 D F 4 E F 4 E F 9 9 9 9
V75 (21-03-2012)
                                                       www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012)
                                                                                                                                               www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf
                                                                                         Pratique du C
 Pratique du C
               Exemples de définition de type
 Les directives au
                                                                                        Les directives au
  préprocesseur
                                                                                        préprocesseur
Types composés
Définition de
 Types composés
Définition de
                                                                                                                                                    .file "typedef.c"
 nouveaux types
                                                                                         nouveaux types
                                                                                                         typedef struct point {
                                                                                                                                                 .globl carre_unite
                 On peut maintenant substituer des synonymes aux modèles
                                                                                                               int x;
                                                                                                                                                  .data
                                                                                                                int y;
                                                                                                                                                  .align 4
                 que l'on déclare :
                                                                                                         } point_t;
                                                                                                                                                 .type carre_unite,@object
                    typedef enum mois_m mois_t; /*d\'efinit pr\'ec\'edement*/
                                                                                                         typedef struct rectangle {
                                                                                                                                                 .size carre unite.16
                                                                                                               point_t P1;
                                                                                                                                                 carre_unite:
                    mois_t mois;
                                                                                                               point_t P2;
                                                                                                                                                    .long
                                                                                                                                                    .long
                                                                                                        } rectangle_t ;
Définition de
                                                                                        Définition de
                    typedef enum {FALSE, TRUE} bool_t; /* conforme \'a la
                                                                                                                                                    .long
                                                                                                        rectangle_t carre_unite = {{ 0, 0},
                                                                norme C
                                                                                                                                                    .long
                    bool_t b, btab[30];
                                                                                                                                    { 1, 1}};
                                                                                                                                                    .text
                                                                                                                                                 .globl main
                    typedef struct point { int x; int y;} point_t;
                                                                                                        int main(void)
                                                                                                                                                 .type main,@function
                    typedef struct rectangle {point_t P1, P2;} rectangle_t;
                                                                                                                                                main:
                    rectangle_t carre_unite = {{ 0, 0}, { 1, 1}};
                                                                                                          return 0 ;
```

```
Pratique du C
Les directives au
                     Modèle de tableaux
  préprocesseur
Types composés
Définition de
  nouveaux types
                       typedef int TableauDe10Entiers[10] ;
Définition de
                       TableauDe10Entiers a,b ;
 nouveaux types
                       cellules de type int.
V75 (21-03-2012)
  Pratique du C
  préprocesseur
Types composés
Définition de
                       binaires
  nouveaux types
                          exemple
                                   struct
Les champs de
lettres binaires
```

V75 (21-03-2012)

```
En utilisant typedef, il est possible de déclarer un type
(modèle) permettant de déclarer des tableaux. Par exemple :
```

ne déclare pas un tableau de 10 entiers mais permet de construire un synonyme (TableauDe10Entiers) utilisable dans la déclaration ultérieure de tableaux. Ainsi, la déclaration

correspond à la déclaration de 2 tableaux a et b de 10

La construction de ce genre de synonyme se base sur la déclaration classique d'une variable; l'usage de typedef permet de considérer que ce qui aurait été l'identificateur de variable (sans typedef) est un identificateur de type (cf. le cours sur les pointeurs de fonctions pour en autre exemple).

www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf V75 (21-03-2012)

Pratique du C Les directives au

préprocesseur Types composés Définition de

nouveaux types

Les champs de

lettres binaires

Pratique du C Les directives au

préprocesseur Types composés Définition de

nouveaux types

Les champs de

lettres binaires

Comme alternative, on peut utiliser des champs de lettres

- champ dans une structure dont on spécifie la longueur;
- considéré comme un petit entier;
- pas d'adresse pour les membres;

```
unsigned int est_mot_cle : 1;
  unsigned int est_externe : 1;
  unsigned int est_statique : 1;
} drapeaux:
/* Mise \'a 1 */
drapeaux.est_externe = drapeaux.est_statique = 1;
/* Test \'a 0 */
if (drapeaux.est_externe == 0 &&
    drapeaux.est_statique == 0 );
```

4 D > 4 B > 4 E > 4 E > 9 Q P www.fil.univ-lille1.fr/~sedoglay/C/Cours04.pdf V75 (21-03-2012) Réaliser un codage d'informations en utilisant une suite de

```
▶ Par exemple, identificateur dans une table des
  symboles : mot-clé, externe, statique :
     #define MOT_CLE 01
     #define EXTERNE 02
     #define STATIQUE 04
  ou bien :
     enum { MOT CLE = 01. EXTERNE = 02. STATIQUE = 04};
assignation
     int drapeaux:
     drapeaux |= EXTERNE | STATIQUE; /* Mise a 1 */
     drapeaux &= ~(EXTERNE | STATIQUE); /* Mise \'a 0 */
▶ test
     if ((drapeaux & (EXTERNE|STATIQUE))==0) /*Test \'a 0*/
```

Dans cet exemple, seuls 3 shannons sont utilisés sur 32.

```
4 D > 4 B > 4 B > 4 B > 9 4 P 9
www.fil.univ-lille1.fr/~sedoglav/C/Cours04.pdf
```

4 D > 4 B > 4 E > 4 E > 9 Q P

www.fil.univ-lille1.fr/~sedoglay/C/Cours04.pdf

```
.data
 unsigned int est_mot_cle : 1;
                                  .globl drapeaux
 unsigned int est_externe : 1;
                                  drapeaux:
  unsigned int est_statique : 1;
                                          .byte
} drapeaux = {
                                          .zero
/* Mise a 1 */
                                          .text
   .est_externe = 1,
                                  .globl main
   .est_statique = 1
};
                                  main:
                                    pushl
                                            %ebp
int
                                    movl
                                            %esp, %ebp
main
                                    subl
                                            $8. %esp
(void)
                                    andl
                                            $-16, %esp
                                    movl
                                            $0, %eax
   /* Test \'a 0 */
                                    subl
                                            %eax, %esp
   drapeaux.est_externe = 0 ;
                                    andb
                                            $-3, drapeaux
                                    movl
   return 0 :
                                            $0, %eax
                                    leave
                                    ret
```