```
Pratique du C
Fonction –
tableau
                                                                                        Pratique du C
Fonction –
                                                                                                       Syntaxe ANSI: définition-de-fonction-ANSI:
                                                                                          tableau
                                                                                         compilation
séparée
                                                                                                                    type-retour
                                                                                                                    identificateur-de-fonction
                                                                                                                    ( liste-de-paramètres-typés<sub>option</sub> )
                                      Pratique du C
                                                                                       Définition d'une
                                                                                       fonction: ANSI
                                   Fonction - tableau
                                                                                                                    liste-de-déclarations-locales option
                                  compilation séparée
                                                                                                                    liste-d'instructions
                                                                                                       Sémantique :

    type-retour : type de la valeur retournée (quelconque),

                                Licence Informatique — Université Lille {\bf 1}
                         Pour toutes remarques : Alexandre.Sedoglavic@univ-lille1.fr
                                                                                                          liste-de-paramètres-typés<sub>option</sub>:
                                                                                                               liste des paramètres formels avec leur type;
                                    Semestre 5 — 2012-2013
                                                                                                          passage de paramètres uniquement par valeur;
                                                                                                          ► liste-de-déclarations-locales option :
                                                                                                               déclaration de variables locales à la fonction;
                                                                                                          liste-d'instructions : corps de la fonction.
V-2 (01-10-2010)
                                                       www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)
                                                                                                                                              www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
  Pratique du C
                Syntaxe ANSI: définition-de-fonction-ANSI:
                                                                                                      Définition à la Kernighan et Ritchie
   tableau
                                                                                          tableau
                                                                                        compilation
séparée
                             type-retour
                                                                                                         Syntaxe K&R: type-retour identificateur-de-fonction
                             identificateur-de-fonction
                                                                                                                            ( liste-d'identificateurs option )
                                 liste-de-paramètres-typés option )
                                                                                                                            liste-de-déclarations 1 option
Définition d'une fonction : ANSI
                                                                                       Définition d'une
                                                                                       fonction : ANSI
                             liste-de-déclarations-locales option
                                                                                                                            liste-de-déclarations<sub>2 option</sub>
                             liste-d'instructions
                                                                                                                            liste-d'instructions
                Une fonction retourne toujours une valeur :
                                                                                                       Sémantique : similaire à la norme ANSI
                  le corps doit contenir au moins une instruction :
                                                                                                          ► liste-d'identificateurs option : liste des paramètres formels
                                return expression;
                                                                                                            sans spécification de type;
                                               sinon le résultat est indéterminé;
                                                                                                          ► liste-de-déclarations<sub>1 option</sub> :

    expression qui doit être de type type-retour;

                                                                                                               déclaration des types des paramètres formels;
                                                                                                          les identificateurs doivent être identiques dans
                   ▶ cette instruction évalue expression qui sera la valeur de
                                                                                                               liste-d'identificateurs et liste-de-déclarations<sub>1</sub>;
                     retour et rend le contrôle d'exécution à l'appelant.
                                                                                                          ▶ si un paramètre est omis dans liste-de-déclarations 1 :
                                                                                                               son type par défaut est int.
                                                        V46 (01-10-2010)
                                                       www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)
                                                                                                                                              www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
 Pratique du C
                                                                                        Pratique du C
               Comparaison ANSI et K&R
                                                                                                      Remarques complémentaires
   tableau
                                                                                          tableau
                Exemple de définition de fonction : norme ANSI
                                                                                         compilation
séparée
                                                                                                          on ne peut pas définir des fonctions dans des fonctions;
                                                                                                          return est une instruction comme une autre :
                    int sum_square(int i, int j)
Définition d'une fonction : ANSI
                                                                                       Définition d'une fonction : ANSI
                                                                                                               ainsi, elle peut être utilisée plusieurs fois dans le corps
                   {
                                                                                                            d'une fonction
                      int resultat;
                      resultat = (i * i) + (j * j);
                      return resultat;
                                                                                                               max
                                                                                                                (int a, int b)
                                                                                                                  if (a > b) return (a); else return(b);
                Exemple de définition de fonction : norme K&R
                                                                                                          répétons que si la dernière instruction exécutée dans
                   int sum_square(i,j)
                                                                                                            une fonction n'est pas un return, le résultat retourné
                     int i,j;
                                                                                                            est indéterminé.
                      int resultat;
                                                                                                       Dans les transparents du cours, les accolades ouvrantes des
                                                                                                       bloc d'instructions ne sont pas sur une ligne indépendante
                      resultat = (i * i) + (j * j);
                      return(resultat);
                                                                                                       uniquement pour permettre la présentation. Ce n'est pas un
```

www.fil.univ-lille1.fr/~sedoglay/C/Cours03.pdf V46 (01-10-2010)

V46 (01-10-2010)

exemple à suivre.

www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf

Pratique du C Fonction – tableau Appel à une fonction compilation séparée Appel à une fonction V46 (01-10-2010) Pratique du C tableau compilation séparée } Passage de V46 (01-10-2010) Pratique du C tableau compilation séparée Les tableaux

```
► Syntaxe de l'appel à une fonction : expression-appel :
     ⇒ identificateur-de-fonction ( liste-d'expressions )
Sémantique :

    évaluation des expressions de liste-d'expressions;

    ► l'ordre d'évaluation n'est pas fixé par la norme;
     résultats passés en paramètres effectifs à la fonction ;
    ▶ le passage se fait par valeur;
```

identificateur-de-fonction; expression-appel : valeur retournée par la fonction ; Exemples :

contrôle d'exécution passé au début de

```
d = sum_square(a,b) / 2;
c = max(a,b);
                      ARKARIATE TOOR
```

www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)

```
En C, les paramètres sont des variables comme les autres.
Un passage d'information se fait par copie des paramètres.
void fct(int bar){
                        int main(void){
```

```
bar = 3;
                      int foo = 5;
return ;
                      fct(foo) ;
                      return foo ;
```

À chaque appel de fonction, de l'espace mémoire est créé pour les paramètres et les variables locales (et détruit après l'appel lors du retour à l'appelant).

```
int foo = 4 ; 4
```

4 D > 4 @ > 4 E > 4 E > E 994 P www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)

Points importants:

la taille d'un tableau est une constante qui doit être calculable à la compilation :

```
char tab[] = "123";
                          .globl tab
                                .data
int main(){
                                .type tab,@object
                                .size
                                       tab,4
        return 0 ;
                         tab:
                                .string "123"
```

les indices dans un tableau commencent en 0;

Les indices d'un tableau de taille N vont de 0 à N-1.

Pratique du C tableau

Pratique du C Fonction – tableau

compilation séparée

Appel à une

Pratique du C

tableau

compilation séparée

Les tableaux

fonction

compilation séparée

Les tableaux

Procédures : fonctions avec effet latéral

- C ne comporte pas de concept de procédures;
- Les fonctions peuvent réaliser tous les effets latéraux voulus:
- ▶ En C, une *procédure* est fonction qui ne retourne aucune valeur (plutôt une valeur indéterminée);
- "Valeur indéterminée" a un type de base, le type void;
- ▶ Il n'a pas de return dans le corps d'une fonction de type de retour void (pour faire cours, d'une procédure);

```
Exemple d'appel de procédure :
  #include<stdio.h>
  void testzero(int j) {
  if(j) return ; /* provoque la sortie */
  printf("test positif") ;
  int main(void) {
  testzero(0);
  return 0 ;
```

101 (8) (3) (3) (4) www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf

Les tableaux en C

En mémoire, un tableau est un bloc d'objets consécutifs de même type.

Sa déclaration est :

- similaire à une déclaration de variable;
- ▶ il faut indiquer le nombre d'éléments entre [].

Quelques exemples:

```
char s[22]; /* s tableau de 22 caract\'eres */
/* t1 tableau de 10 entiers longs et
   t2 tableau de 20 entiers longs */
long int t1[10], t2[20];
#define N 100
int tab[N/2];
```

(D) (A D) (E) (E) (O) www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf

Définition d'un tableau lors de sa déclaration

L'initialisation d'un tableau se fait :

- par des valeurs constantes placées entre {} séparées par des virgules (,);
- ▶ si il n'y a pas assez de valeurs : l'espace mémoire restant est soit indéterminé soit mis à 0;
- Par exemple : int t[4] = { 1, 2, 3, 4 };
- li n'y a pas de facteur de répétition.

Pratique du C Fonction – tableau compilation séparée Les tableaux V46 (01-10-2010) Pratique du C tableau compilation séparée { Tableaux passés d'une fonction V46 (01-10-2010) Pratique du C tableau

```
int
Exemple de
                    }-
programme:
d'Ératosthène
V46 (01-10-2010)
```

Manipulations élémentaires sur les tableaux

Accès à un élément de tableau par opérateur d'indexation;

- $expression \leftarrow identificateur-de-tableau [expression_1]$ Sémantique :
 - expression₁ délivre une valeur entière;
 - expression délivre l'élément d'indice expression1;
 - expression peut être une valeur de gauche comme dans l'exemple x = t[k]; t[i+j] = x;

L'identificateur t n'est pas une variable. Il est associé à une adresse constante correspondant au début de la mémoire allouée au tableau. En mémoire, on a les octets :

Comparer 2 identificateurs de tableau revient à comparer 2 adresses et non pas les objets stockés à ces adresses. De même, affecter quelque chose à cet identificateur t = ... n'a pas de sens.

www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)

Tableau bidimensionel

Bien que stockés linéairement, les tableaux peuvent être définis comme multidimensionel :

```
char tab[3][4]={"123","456","789"}; .file "tableau2d.c"
                                    .globl tab
                                     data
main
                                    .type tab,@object
(void)
                                    .size tab,12
                                    tab:
                                    .string "123"
  return 0 ;
                                    .string "456"
                                    .string "789"
```

La sémantique est la même que pour le cas monodimensionnel:

```
tab[3][0] = tab[3][0]++
```

4 D > 4 B > 4 E > 4 E > 9 Q P www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)

```
Pratique du C
  #include<stdio.h>
  #define IS_NON_PRIME 0
  #define IS_PRIME 1
  #define IS CANDIDATE 2
  #define N 100
  int prem[N];
void init (void)
    register int i;
    prem[0]=prem[1]=IS_NON_PRIME;
    for (i = 2; i < N; i = i + 1) prem[i] = IS_CANDIDATE;</pre>
int min_is_candidate (void)
    register int i = 0;
    while (prem[i] != IS_CANDIDATE) i = i + 1;
    return i:
```

tableau compilation séparée

Pratique du C Fonction – tableau

compilation séparée

Tableaux passés

en paramètre

d'une fonction

tableau

compilation

Tableaux passés

d'une fonction

fonction

Exemple de programme: d'Ératosthène

www.fil.univ-lille1.fr/~sedoglay/C/Cours03.pdf V46 (01-10-2010)

40) 40) 42) 42) 2 990

Puisque l'identificateur d'un tableau n'est pas une variable, quelle copie est faite lors du passage de paramètre suivant :

Passage d'un tableau en paramètre d'une

```
void fct(int tib[]){
                              int main(void){
                                  int tab[2] = { 0, 1};
  tib[0] = 1;
  return ;
                                  fct(tab) ;
                                 return tab[0];
```

C'est l'adresse qui est copiée. Ceci implique que la fonction principale retourne 1 dans notre exemple.

Dans fct, tib[0] fait référence à la première cellule mémoire définie dans le tableau local à la fonction principale.

Nous étendrons ce principe (passage de paramètre par adresse) aux autres types en utilisant la notion de pointeur.

```
www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
```

Un petit coup d'oeil du coté de l'assembleur

```
.file "tableau.c"
                                    char tab[] = "123";
.globl tab
                                    unsigned int i =0;
                                    int main(){
        .type tab,@object
        .size tab,4
       .string "123"
                                            return 0 :
.globl i
                                    }
        .align 4
        .type i,@object
        .size i,4 /* Ce code compile en lan\c{c}ant un
        .long 0
                         avertissement~:
                        warning: assignment makes integer
        .text
        .align 2
                        from pointer without a cast */
.globl main
        .type main,@function
main:
      . . . . . . .
               $tab, i /* Nous verrons pourquoi lors de
        movl
               $0, %eax l'\'etude des pointeurs
       movl
                                  4 D > 4 B > 4 E > 4 E > 9 Q P
                                  www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
void set_non_prime(int start)
{
    register int i = start + 1;
    for (:i < N: i = i + 1)
```

```
if (i % start == 0) prem[i]=IS_NON_PRIME;
     return ;
}
int main(void)
     register int next_prime = 1, i;
      init();
     while (next_prime * next_prime < N) {
     next_prime=min_is_candidate();
     prem[next_prime]=IS_PRIME;
     set_non_prime(next_prime);
     printf("Liste des nombres
             premiers inf\\'erieurs \\'a %d\n", N);
     for (i = 0; i < N; i = i + 1)
      if (prem[i] != IS_NON_PRIME) printf("%d ", i);
                                    www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
```

```
Pratique du C
Fonction –
tableau
                                                                                                    La fonction principale eratosMain.c
 Pratique du C
Fonction –
tableau
                                                                                                    (permet entre autre de déclarer les identificateurs) :
                Nous allons reprendre l'exemple du crible d'Ératosthène pour
  compilation
séparée
                                                                                      compilation
séparée
                illustrer la notion de compilation séparée et l'utilitaire de
                                                                                                    #include <stdio.h>
                gestion make associé à cette notion.
                                                                                                    #include "eratosthene.h"
                                                                                                    void init (void) ;
                                                                                                                                    /* le prototype des fonctions */
                Objectif: diviser un programme C en plusieurs fichiers afin
                                                                                                    int min_is_candidate(void) ; /* utilis\'ees doit \^etre */
                                                                                                    void set_non_prime(int) ; /* disponible
                d'en faciliter la maintenance.
                                                                                                    int prem[N]; /* la variable globale est d\'efinie ici */
                Il faut prendre garde à gérer correctement les dépendances
                                                                                                    int main(void) {
                entre les différents fichiers.
                                                                                                         register int next_prime = 1, i;
                                                                                                         init();
                Pour commencer, on peut regrouper les définitions de macro
                                                                                                         while (next_prime * next_prime < N) {</pre>
                dans un fichier eratosthene.h:
                                                                                                         next_prime=min_is_candidate();
                                                                                                         prem[next_prime]=IS_PRIME;
                   #define IS_NON_PRIME 0
                                                                                                         set_non_prime(next_prime);
                   #define IS PRIME 1
                   #define IS CANDIDATE 2
                                                                                                         printf("Liste des nombres
Compilation
                                                                                    Compilation
                   #define N 100
                                                                                                                  premiers inf\\'erieurs \\'a %d\n", N);
séparée et Make
                                                                                     séparée et Make
                                                                                                         for (i = 0; i < N; i = i + 1)
                Un programme doit contenir une fonction principale (main).
                                                                                                             if (prem[i] != IS_NON_PRIME) printf("%d ", i);
                                                                                                         return 0 :
                                                                                                                                          THE PROPERTY OF STREET
V46 (01-10-2010)
                                                                                                                                          www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
                                                      www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)
  Pratique du C
                                                                                      Pratique du C
               Fichiers composant notre programme
                                                                                                   Obtention d'un exécutable
   tableau
                                                                                       tableau
                                                                                      compilation
séparée
  compilation
séparée
                Il est possible d'obtenir un fichier objet associé à ce code :
                % gcc -c eratosMain.c
                                                                                                    Au final, on obtient
                % ls
                eratosMain.c eratosMain.o eratosthene.h
                                                                                                    % gcc -c eratosInit.c
                Puis, on peut par exemple faire un fichier par fonction :
                                                                                                    eratosInit.c eratosMain.c eratosMin.c eratosSet.c
                                                                                                    eratosInit.o eratosMain.o eratosMin.o eratosSet.o
                #include "eratosthene.h"
                                                                                                    eratosthene.h
                extern int prem [N]; /* prototype de la variable globale */
                                                                                                    Pour conclure, on fait l'édition de lien de ces fichiers objets :
                init
                                                                                                    % gcc -o executable eratos*.o
                (void)
                                                                                                    % executable
                { /* la d\'efinition de la fonction init */
                                                                                                    Liste des nombres premiers inf\'erieurs \'a 100
                   register int i:
                                                                                                    2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59
                                                                                    Compilation
Compilation
                   prem[0]=prem[1]=IS_NON_PRIME;
                                                                                                    61 67 71 73 79 83 89 97
séparée et Make
                                                                                     séparée et Make
                   for (i = 2; i < N; i = i + 1) prem[i] = IS_CANDIDATE;</pre>
                   return :
                                                      4 D > 4 @ > 4 E > 4 E > E 994 P
                                                                                                                                          4 D > 4 B > 4 E > 4 E > 9 Q P
V46 (01-10-2010)
                                                      www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)
                                                                                                                                          www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
  Pratique du C
                                                                                      Pratique du C
               Arbre de dépendances
                                                                                                   Utilitaire make : syntaxe
  Fonction –
tableau
                                                                                       tableau
                                                                                      compilation
séparée
                                                                                                    Pour les projets importants (le code source de Linux est
                Les opérations précédentes sont modélisées par l'arbre de
                                                                                                    constitué de 921 fichiers), il faut automatiser les tâches.
                dépendances :
                                                                                                    Automatisation de la compilation :
                                                                                                       ► Maintenance, mise à jour et régénération de fichiers
                                                                                                         dépendants;
                                                                                                       ▶ Sources → exécutables;
                                                                      eratosMin.o

    Recompilation quand nécessaire (dates);
```

Compilation séparée et Make

► Fichier de règles de dérivation (code l'arbre de

www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf

Makefile ou makefile.

dépendances)

Compilation séparée et Make

V46 (01-10-2010)

www.fil.univ-lille1.fr/~sedoglay/C/Cours03.pdf V46 (01-10-2010)

```
Format d'une règle : quoi, pourquoi, comment.
 Pratique du C
Fonction –
tableau
                                                                                   Pratique du C
Fonction –
                                                                                                Utilitaire make : notre exemple
                                     target : dependencies
                 syntaxe :
                                                                                     tableau
                                     (tabulation) commands
                                                                                                 Dans notre cas, on peut écrire le Makefile suivant :
                 quoi (target) objectif, généralement un fichier;
                                                                                                 OPTIONS = -Wall -ansi -pedantic
                 pourquoi (dependencies) liste des fichiers/cibles dont
                                                                                                 OBJETS = eratosMain.o eratosMin.o eratosSet.o eratosInit.o
                    dépend target;
                                                                                                 executable: $(OBJETS)
                 ▶ comment (commands) commandes à exécuter pour
                                                                                                         gcc $(OPTIONS) -o executable $(OBJETS)
                    réaliser target;
                                                                                                 eratosMain.o: eratosMain.c eratosthene.h
               On peut n'exécuter qu'une partie de l'arbre : %make target
                                                                                                         gcc $(OPTIONS) -c eratosMain.c
               Exemple (makefile pour un programme C)
                                                                                                 eratosMin.o: eratosMin.c eratosthene.h
                .PHONY:clean
                                                                                                         gcc $(OPTIONS) -c eratosMin.c
               executable: f1.o f2.o
                       gcc -o executable f1.o f2.o
                                                                                                 eratosSet.o: eratosSet.c eratosthene.h
               f1.o: f1.c fichier.h
Compilation
                                                                                                         gcc $(OPTIONS) -c eratosSet.c
                                                                                  Compilation
                      gcc -c f1.c
séparée et Make
                                                                                  séparée et Make
               f2.o: f2.c fichier.h
                                                                                                 eratosInit.o: eratosInit.c eratosthene.h
                       gcc -c f2.c
                                                                                                         gcc $(OPTIONS) -c eratosInit.c
               clean:
                       rm -f *~ *.o executable
                                                     COLUMN TERRITOR TO SO OF
                                                                                                                                      4 D > 4 B > 4 B > 4 B > 9 Q P
                                                                                                                                     www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
V46 (01-10-2010)
                                                    www.fil.univ-lille1.fr/~sedoglay/C/Cours03.pdf V46 (01-10-2010)
  Pratique du C
                                                                                   Pratique du C
              Algorithme et macros de make
                                                                                                Le dévermineur gdb
   tableau
                                                                                     tableau
  compilation
séparée
                                                                                   compilation
séparée
                 ▶ Pour chaque cible
                                                                                                 L'environnement gdb permet d'exécuter des programmes pas

    Vérifier les dépendances

                                                                                                 à pas et d'examiner la mémoire du processus en cours.
                           → Récursion
                                                                                                 Pour utiliser gdb, l'exécutable doit avoir été compilé avec
                           → Date des fichiers de base
                                                                                                 l'option -g.
                      Si modification
                                                                                                 On l'utilise dans un shell en indiquant le fichier à examiner :
                              alors → Lancer les commandes
                                                                                                 % gdb executable
                              sinon → Fichier à jour
                                                                                                 GNU gdb 5.3-22mdk (Mandrake Linux)
                                                                                                 ..... etc..........
                                                                                                 This GDB was configured as "i586-mandrake-linux-gnu"...
                $0 représente le nom complet de la cible courante;
                $? représente les dépendances plus récentes que la cible;
                                                                                                 Ce programme propose une aide en ligne :
                $< représente le nom de la première dépendance;
                                                                                                 (gdb) help help
Compilation
                $^ représente la liste de toutes les dépendances;
séparée et Make
                                                                                                 Print list of commands.
               On peut définir ses propres macros :
                                                                                                 (gdb) help quit
                                                                                  Exécution pas à
                                                                                  pas dans
l'environnement
                                                                                                 Exit gdb.
               REP = /etc/ /bin/ /usr/bin/
                                                                                  gnu debugger
                                                     (D) (A) (E) (E) (O)
                                                                                                                                      4 D F 4 D F 4 E F 4 E F 9 9 9 9
V46 (01-10-2010)
                                                    www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)
                                                                                                                                      www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf
  Pratique du C
                                                                                   Pratique du C
              Exécution et examen du code source
                                                                                                Placer des points d'arrêt
   tableau
                                                                                     tableau
               Le programme considéré peut être exécuté dans
                                                                                                 La commande break permet de placer un point d'arrêt sur
  compilation
séparée
                                                                                    compilation
séparée
               l'environnement gdb:
                                                                                                 une instruction du programme source de manière à ce qu'à
                                                                                                 la prochaîne exécution du programme dans gdb, l'invite du
                (gdb) run
                                                                                                 dévermineur soit disponible avant l'exécution de cette
               Starting program: /home/.../executable
               Liste des nombres premiers inf\'erieurs \'a 100
                                                                                                 instruction.
               2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59
                                                                                                 Une instruction du programme source peut être repérée par
               61 67 71 73 79 83 89 97
                                                                                                 le numéro de ligne correspondant ou par un identificateur :
               Program exited normally.
                                                                                                 (gdb) break 10
                                                                                                 Breakpoint 1 at 0x8048353: file eratosMain.c, line 10.
               Lorsque le code source de l'exécutable est disponible la
                                                                                                 (gdb) break min_is_candidate
               commande list permet d'afficher le code source avec
                                                                                                 Breakpoint 2 at 0x80483f2: file eratosMin.c, line 4.
               chacune de ces lignes numérotées. Dans notre cas :
                                                                                                 permet de placer deux points d'arrêts aux endroits spécifiés.
               (gdb) list
                                                                                                 la commande info fournit la liste des points d'arrêts :
               1
                        #include <stdio.h>
               2
                       #include "eratosthene.h'
                                                                                                 (gdb) info break
Exécution pas à
                                                                                  Exécution pas à
               3
                                                                                                 Num Type
                                                                                                                Disp Enb Address What
pas dans
                                                                                  pas dans
l'environnement
               4
                        void init (void) ;
                                                                                                 1 breakpoint keep y 0x08048353 in main at eratosMain.c:10
gnu debugger
```

V46 (01-10-2010)

gnu debugger

www.fil.univ-lille1.fr/~sedoglay/C/Cours03.pdf V46 (01-10-2010)

www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf

2 breakpoint keep y 0x080483f2 in min_is_candidate at ... 4 m >

```
Pratique du C
Fonction –
tableau
  Pratique du C
Fonction –
tableau
                Exécution pas à pas
                 Une fois ceci fait, exécutons notre programme dans gdb :
  compilation
séparée
                                                                                              compilation
séparée
                 Starting program: /home/.../executable
                 Breakpoint 1, main () at eratosMain.c:10
                                init();
                 Pour provoquer l'appel init(), utilisons la commande
                 next:
                 (gdb) next
                                 while (next_prime * next_prime < N) {
                 On peut exécuter les instructions associées
                 (gdb) step
                 init () at eratosInit.c:7
                             prem[0] = prem[1] = IS_PRIME;
                 Pour exécuter les instructions jusqu'au prochain point d'arrêt
Exécution pas à
                                                                                            Exécution pas à
                 (gdb) continue
pas dans
l'environnement
gnu debugger
                                                                                            pas dans
                 Continuing. Breakpoint 2, min_is_candidate () at eratosMin.c:4 | Tenvironnement gnu debugger
                                 register int i = 0;
                                                          www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf V46 (01-10-2010)
V46 (01-10-2010)
  Pratique du C
Fonction –
                Quelques remarques : gdb est un outils très
  tableau
compilation
séparée
                puissant
                 Remarquez qu'à l'entrée d'une fonction, les paramètres sont
                 indiqués :
                 (gdb) contenu
                 Continuing.
                 Breakpoint 1, set_non_prime (start=3) at eratosSet.c:5
                                 register int i = start + 1;
                 On peut modifier les valeurs des variables en cours
                 d'exécution :
                 (gdb) set variable start = 0xb
                 (gdb) print start
                 $15 = 11
                 Il est possible de tracer l'exécution, de l'interrompre lors
Exécution pas à
pas dans
l'environnement
                 d'événements prédéfinis, etc.
gnu debugger
                 Pour plus d'information, utilisez l'aide en ligne de gdb.
```

V46 (01-10-2010)

www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf

Affichage du contenu des variables et de la mémoire

Pour afficher le contenu d'une variable, il suffit d'utiliser print

```
(gdb) print prem

$3 = {0, 0, 1, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, ... etc...

0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2, 0, 2}

(gdb)
```

On peut provoquer l'affichage à chaque arrêt avec display et le formatter avec printf

```
(gdb) printf "%x\n",prem[1]
```

Plus généralement, on obtient l'affichage d'une zone mémoire grâce à la commande :

```
(gdb) x /4xw 0xbffff6a4
0xbffff6a4: 0x00000064 0xbffff6b8 0x0804836b 0x4014cf50
```

www.fil.univ-lille1.fr/~sedoglav/C/Cours03.pdf