C pointeurs et autres concepts choisis

I Types : en C des données ont un type

(constantes : découle normalement de la valeur : 3.14 : double, 4 : int, 'a' : char
variables, fonctions : doit être déclaré)

- types élémentaires : Entier (int), Caractère (char), Enum.° (enum)
  Flottant (float, double, long double), type vide (void)

- types composés (à partir de TE $T_1, T_2, \ldots T_n$) :
  - pointeur vers T_1
  - tableau de T_1
  - structure de $T_1, \ldots T_n$
  - fonction à valeur de retour de type T_1

— pointeurs : pour chaque type T on peut produire un type pointeur « pointeur vers T »
déclaration : T *
∃ valeur spéciale : pointeur vide (NULL)

2 opérateurs utiles :
- op. adresse &
- op. déréférenciation *

exemple `int i; int *p; p = &i; i = *p;`

— tableaux : collections d'objet de même type
ex : `int vec[100]`

— structures
`struct nom { composants };`
exemple
```
struct complex { double re;
                 double im; };
struct complex x1, x2, *p;
x1.re = 5;
x1.im = 2;
p = &x2;
p -> re = 3;
p -> im = 1;
```

— fonctions
`float square(float x) { return x*x; }`

II Visibilité variables

classes
respectivement (auto, register) variables existent et jusqu'à fin
static, extern " visibles bloc {}.
gardent leur valeur

III mémoire

définition : place réservée automatiquement.

mais attention : place pour quoi ?

int i; → pour un entier
int i[10] 10 " "
int *pi pointeur vers "
↝ *pi = 3 cata !!!