Pointeurs + fonctions + tableaux (pft)

Seance 8

UE « introduction à la programmation »

Bruno Bouzy

bruno.bouzy@parisdescartes.fr

PFT

- Exemple d'utilisation simultanée des 3 concepts vus aux 3 séances précédentes
  - **Pointeurs**, **Fonctions**, **Tableaux**
- Un **tableau** est utilisé
  - Accès par **pointeur** ou par valeur
- Une **fonction** par objectif du programme
  - Afficher, lire ou écrire dans les **tableaux**
- Une **fonction** avec sortie
  - Utilisation obligée de **pointeur**

Objectif du programme `codeInt.c` (1/5)

- Gérer des nombres entiers modulo N
  - Exemple: N = 5, E = { 0, 1, 2, 3, 4 }

- Coder ces nombres avec deux clés A et B
  - `code(x) = Ax + B (mod 5)`
  - Exemple: A=2, B=1,
  - code(0) = 1, code(1) = 3, code(2) = 0
  - code(3) = 2, code(4) = 4.
- Utiliser une fonction `codeN`

Objectif du programme `codeInt.c` (2/5)

- Mettre les résultats dans 4 tableaux
  - si: suite initiale de T nombres de E
    - Exemple: T = 6, `si = {0, 2, 4, 0, 3, 3 }`
  - sc: suite codée de si
    - Exemple: `sc = {1, 0, 4, 1, 2, 2 }`
  - tcode: tableau des codes
    - Exemple: `tcode = {1, 3, 0, 2, 4}`
  - tdecode: tableau des codes inverses
    - Exemple: `tdecode = { 2, 0, 3, 1, 4}`

Objectif du programme `codeInt.c` (3/5)

- Utiliser des fonctions
  - Une fonction `codeN` pour calculer le code d'un nombre
    - Version avec `return`
    - Version avec paramètre de sortie (`codeNbis`)
  - Une fonction `codeSuite` pour remplir un tableau avec les codes d'un autre tableau
  - Une fonction `afficheSuite` pour afficher un tableau
  - Une fonction `decodeN` pour calculer le code inverse d'un nombre

Objectif du programme `codeInt.c` (4/5)

- Le programme principal:
  - affiche `N` et `TAILLE`.
  - affiche les clés `A` et `B`.
  - initialise la suite `si` avec un tableau et l'affiche.
  - calcule la suite `sc` et l'affiche.
  - calcule le tableau `tcode` et l'affiche.
  - calcule le tableau `tdecode` et l'affiche.

Objectif du programme `codeInt.c` (5/5)

- Les constantes du programme:
  - Le nombre `N`.
  - La taille des suites `si` et `sc`: `TAILLE`.
  - La clé `A`: `CLE_A`.
  - La clé `B`: `CLE_B`.

Déclarer les constantes

```
#define NBL 13

#define TAILLE 10

#define CLE_A 5

#define CLE_B 3
```

La fonction `codeN`

```
// une fonction 'codeN'
// prenant un entier n et un entier a en entree,
// et retournant le reste de n*a divisé par NBL

int codeN(int n, int a, int b)
{
  return (n*a+b) % NBL;
}
```

La fonction `codeNbis`

```
// la meme sour forme de procedure avec
// parametre en sortie avec pointeur

void codeNbis(int n, int a, int b, int * r)
{
  *r = (n*a+b) % NBL;
}
```

La fonction `afficheSuite`

```
// fonction affichant un tableau (une suite) 's'
// de taille 'l'.

void afficheSuite(int * s, int l)
{
  int i;
  printf("{ ");
  for (i=0; i<l; i++) printf("%d ", *(s+i));
  printf("}\n");
}
```

La fonction `codeSuite`

```
// fonction transformant une suite de nombres s1
// en une suite de nombres s2 codée.
// s1: suite initiale de nombres
// s2: suite codee de nombres
// l: longueur de la suite
// a, b: cles de codage

void codeSuite(int * s1, int * s2, int l, int a, int b)
{
  int i;
  for (i=0; i<l; i++) *(s2+i) = codeN(*(s1+i), a, b);
  // for (i=0; i<l; i++) codeNbis(*(s1+i), a, b, s2+i);
}
```

La fonction `decodeN`

```
// une fonction 'decodeN' decodant un nombre n
// en essayant toutes les possibilites avec
// 'codeN'...

int decodeN(int n, int a, int b)
{
  int j=0;
  while ((j<NBL) && (codeN(j, a, b)!=n)) j++;
  return j;
}
```

La fonction `main` (1/2)

```
int main()
{
  printf("Bonjour.\n");
  printf("NBL = %d, TAILLE = %d\n", NBL, TAILLE);
  printf("CLE_A = %d, CLE_B = %d\n", CLE_A, CLE_B);

  // afficher une suite de nombres initiale
  int si[TAILLE] = {2, 13, 3, 4, 4, 12, 10, 7, 11, 9};
  printf("suite initiale = "); afficheSuite(si, TAILLE);

  // afficher la suite de nombres codee
  int sc[TAILLE];
  codeSuite(si, sc, TAILLE, CLE_A, CLE_B);
  printf("suite codee    = "); afficheSuite(sc, TAILLE);

  ...
```

La fonction `main` (2/2)

```
  ...
  // construire un tableau avec les valeurs de codeN
  // pour tous les nombres
  int tcode[NBL];
  int i; for(i=0; i<NBL; i++)
    tcode[i] = codeN(i, CLE_A, CLE_B);
  printf("tcode      = "); afficheSuite(tcode, NBL);

  // construire un tableau avec les valeurs inverses
  // de codeN pour tous les nombres.
  int tdecode[NBL];
  for(i=0; i<NBL; i++)
    tdecode[i] = decodeN(i, CLE_A, CLE_B);
  printf("tdecode    = "); afficheSuite(tdecode, NBL);

  printf("Au revoir.\n");
  return 0;
}
```

Une exécution...

```
ProgC/Seance8-pft/Cours/Src :-) gcc codeInt.c
ProgC/Seance8-pft/Cours/Src :-) ./a.out
Bonjour.
NBL = 13, TAILLE = 10
CLE_A = 5, CLE_B = 3
suite initiale = { 2 13 3 4 4 12 10 7 11 9 }
suite codee    = { 0 3 5 10 10 11 1 12 6 9 }
tcode      = { 3 8 0 5 10 2 7 12 4 9 1 6 11 }
tdecode    = { 2 10 5 0 8 3 11 6 1 9 4 12 7 }
Au revoir.
```

Exemple complet (1/6)

- Ecrire une fonction:

```
int deIntervalleANombre(int a, int b)
```

  - demandant à l'U un nombre entier dans [a, b],
  - retournant ce nombre,
  - (la demande est renouvelée tant que l'U n'a pas tapé une valeur dans [a, b])

Exemple complet (2/6)

```
int deIntervalleANombre(int a, int b)
{
  int x;

  do {
    printf("x ? (%d<=x<=%d) ", a, b);

    scanf("%d", &x);

  } while (x<a || x>b);

  return x;
}
```

Exemple complet (3/6)

- Ecrire une fonction:

```
void tabMaxMin()
```

  - prenant en **entrée** un tableau d'entiers `tab` et une longueur `l` de tableau,
  - donnant en **sortie** le maximum `max` et le minimum `min` des valeurs du tableau.

Exemple complet (4/6)

```
void tabMaxMin(int * tab, int l, int * max,
int * min)

{
   int i;
   *max = tab[0];
   *min = tab[0];

   for (i=0; i<l; i++) {
      if (*max<tab[i]) *max = tab[i];
      if (*min>tab[i]) *min = tab[i];
   }
}
```

Exemple complet (5/6)

- Ecrire un programme principal `main`:
  - remplissant un tableau d'entiers de taille `TAILLE` appartenant à `[0, 9]`,
  - appelant la fonction `deIntervalleANombre`,
  - affichant les valeurs du tableau,
  - appelant la fonction `tabMaxMin`,
  - affichant le maximum et le minimum des valeurs.

Exemple complet (6/6)

```
#include <stdio.h>
#define TAILLE 3
// ici les definitions des fonctions
int main() {
  int t[TAILLE], i, max, min;
  for (i=0; i<TAILLE; i++) {
    printf("t[%d] : ", i);
    t[i] = deIntervalleANombre(0,9);
  }
  for (i=0; i<TAILLE; i++)
    printf("t[%d] = %d, ", i, t[i]);
  printf("\n");
  tabMaxMin(t, TAILLE, &max, &min);
  printf("max = %d, min = %d\n", max, min);
  return 0;
}
```

Résumé

- Exemple de 2 programmes C

- Utilisation simultanée de :
  - Pointeurs et fonctions,
  - Fonctions et tableaux,
  - Tableaux et pointeurs.
- A faire:
  - améliorer la fonction `decode`,
  - utiliser des tableaux de `char` et `N=26`.