Licence d'Informatique

TD d'IPR : Feuille 4

Pointeurs

Exercice 1 Que font les programmes `pointeur1.c` et `pointeur2.c`?

```
/* pointeur1.c */
#include <stdio.h>
#include <stdlib.h>

int
main(void) {
  int i, j;
  int *a, *b, **c;

  i = 1;
  j = 2;
  printf("valeur de i, j : %d %d \n", i, j);

  a = &i;
  printf("valeur de *a apres a=&i : %d\n", *a);

  *a = *a + 5;
  printf("valeur de i apres *a=*a+5 : %d\n", i);

  i = i + 5;
  printf("valeur de *a apres i=i+5 : %d\n", *a);

  b = a;
  printf("valeur de *b apres b=a : %d\n", *b);

  *b = *b + 5;
  printf("valeur de i, *a et *b apres *b=*b+5 : %d %d %d\n",
         i, *a, *b);

  b = &j;
  c = &b;
  printf("valeur de j, *b, **c apres b=&j et c= &b : %d %d %d\n",
         j, *b, **c);

  printf("valeur de b, *b, c, *c, **c : %p %d %p %p %d\n",
         b, *b, c, *c, **c);

  return EXIT_SUCCESS;
}
```

```
/* pointeur2.c */
#include <stdio.h>
#include <stdlib.h>

int
main(void) {
  int tab[5] = {1, 2, 3, 4, 5};
  int* t = tab;
  int i;
  char mot[] = "arithmetique des pointeurs";
  char* s;

  for (i=0; i<5; i++)
    printf("% d, %d \n", tab[i], *t++);

  for (s = mot; *s; s++)
    printf("%s\n", s);

  return EXIT_SUCCESS;
}
```

Exercice 2 Que font les fonctions suivantes.

```
void
echanger(int x, int y) {
  int temp;
  temp = x;
  x=y;
  y=temp;
}
void
echanger(int *px, int *py) {
  int temp;
  temp = *px;
  *px=*py;
  *py=temp;
}
```

Exercice 3 Ecrire les fonctions suivantes. Pour chacune d'elles rédiger deux versions : l'une avec notation tableau et l'autre avec notation pointeur.

- `char* prem_char(char *s, char c)`

  qui retourne un pointeur vers la première occurrence du caractère c dans la chaine s'il est présent, le pointeur NULL sinon.

- `char* dern_char(char *s, char c)`

  retourne un pointeur vers la dernière occurrence du caractère c dans la chaine s'il est présent, le pointeur NULL sinon.

- `int souschn(char * s, char *t)`

  qui retourne la position de t dans s, -1 si n'existe pas.

Utiliser ces fonctions pour écrire la commande d'usage :

```
aff_chaines caractere chaine1 chaine2
```

Cette commande affiche :

- la partie de chaine1 se trouvant après la 1ère occurrence du caractère (la chaine entière si le caractère n'est pas présent).
- la partie de chaine2 se trouvant après la dernière occurrence du caractère (la chaine entière si le caractère n'est pas présent).
- la partie de chaine1 se trouvant à partir du motif chaine2 (le message "motif non present" si ce motif n'apparait pas dans la chaine).

Exemple d'exécution :

```
> aff_chaines a abbabcccannn abcccan
bbabcccannn
n
abcccannn

> aff_chaines a abbabcccannn abccanm
bbabcccannn
nm
motif non present
```

Exercice 4 Que fait le programme suivant?

```
/* ini_chn_vct.c */
#include <stdlib.h>
#include <stdio.h>

char v1[] = { '1', '2', '3', '4', '5', '6'};
char v2[] = "123456";
char v3[10] ={ '1', '2', '3', '4', '5', '6'};
char v4[10] ="123456";
char *p="123456";

int
main(void)
{
  printf("Taille de v1 : %d\n", sizeof v1);
  printf("Taille de v2 : %d\n", sizeof v2);
  printf("Taille de v3 : %d\n", sizeof v3);
  printf("Taille de v4 : %d\n", sizeof v4);
  printf("Taille de p  : %d\n", sizeof p);

  printf("Modification de v2[0]\n");
  v2[0]='A';
  printf("OK\nModification de *p\n");
  *p='A';
  printf("OK\n");
  return EXIT_SUCCESS;
}
```

Exercice 5

1-Que font les fonctions suivantes?

```
static char tabjour[2][13] = {
  {0,31,28,31,30,31,30,31,31,30,31,30,31},
  {0,31,29,31,30,31,30,31,31,30,31,30,31}
};

int
jour_annee(int annee, int mois, int jour) {
  int i,bis;
  bis = annee%4 ==0 && annee%100 != 0 || annee%400 ==0;
  for (i = 1; i < mois; i++)
    jour += tabjour[bis][i];
  return jour;
}

int
mois_jour(int annee, int jourannee, int *pmois, int *pjour) {
  int i,bis;
  bis = annee%4 ==0 && annee%100 != 0 || annee%400 ==0;
  for (i = 1; jourannee > tabjour[bis][i]; i++)
    jourannee -= tabjour[bis][i];
  *pmois = i;
  *pjour = jourannee;
}
```