Initiation à la programmation en C

Correction du TP n°3

Antoine Miné

1er mars 2007

Site du cours: `http://www.di.ens.fr/~mine/enseignement/prog2006/`

Exercice 1. Dates.

```
int jour_mois[12] = { 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31 };

int date(int jour, int mois, int an)
{
  int i;

  if (mois<1 || mois>12) {
    /* on quitte tout de suite pour éviter un accès invalide dans jour_mois */
    printf("mois invalide!\n");
    return 0;
  }

  for ( i=0; i < mois-1; i++ )
    /* pour chaque mois complètement écoulé */
    jour += jour_mois[i];

  if (mois>=3 && !(an%4) && ((an%100) || !(an%400))) {
    /* année bissextile, à partir de mars */
    jour ++;
  }
  return jour;
}
```

Exercice 2. Crible d'Ératosthène.

```
#include <stdio.h>

/* on cherche tous les nombre premiers inférieurs à N */
#define N 10000000

/* alloue un seul octet par nombre à tester */
char tab[N];

int main()
{
  int i,j;

  /* initialisation: tout le monde est premier */
```

```
  for (i=0;i<N;i++) tab[i] = 1;

  /* biffage */
  for ( i=2; i<N; i++ ) {
    if (tab[i]) {
      for ( j=2; j*i<N; j++)
        /* on biffe les multiples stricts de i < à N */
        tab[i*j] = 0;
    }
  }

  /* affichage */
  for ( i=2; i<N; i++ )
    if (tab[i]) printf("%i est premier\n",i);

  return 0;
}
```

Exercice 3. Tri de tableaux.

```
#include <stdio.h>
#include <stdlib.h> /* pour lrand48 */

#define N 10000

int tab[N];

void tri_bulle()
{
  int fin; /* sert à savoir quand s'arréter */
  do {
    int i;

    fin = 1; /* armement du drapeau de fin */

    for ( i=0; i<N-1; i++ )
      if (tab[i]>tab[i+1]) {

        /* échange de tab[i] et tab[i+1]
           il faut passer par une variable temporaire
         */
        int temp = tab[i];
        tab[i] = tab[i+1];
        tab[i+1] = temp;

        fin = 0; /* la tableau n'était pas trié */
      }
  }
  while (!fin);
}

/* vérifie que tab est bien trié */
void verifie()
{
  int i;
  for ( i=0; i<N-1; i++ )
    if (tab[i]>tab[i+1]) {
      printf("tableau mal trié!\n");
      return;
    }
```

```
}

/* test sur un tableau déja trié */
void test1()
{
  int i;
  for ( i=0; i<N; i++ ) tab[i] = i;
  tri_bulle();
  verifie();
}

/* test sur un tableau à l'envers */
void test2()
{
  int i;
  for ( i=0; i<N; i++ ) tab[i] = N-i;
  tri_bulle();
  verifie();
}

/* test sur un tableau aléatoire */
void test3()
{
  int i;
  srand48(time(NULL)); /* initialisation par la date courante */
  for ( i=0; i<N; i++ ) tab[i] = lrand48();
  tri_bulle();
  verifie();
}

int main()
{
  test1();
  test2();
  test3();
  return 0;
}
```

Exercice 4. Traduction de boucles.

for.c

```
x=10;
while (x<100) {
  printf("%i\n",x);
  x++;
}
```

dowhile.c

```
if (x%2) x++;
else {
  printf("%i\n",x);
  x += 2;
}
while (x<y) {
  if (x%2) { x++; continue; }
  printf("%i\n",x);
  x += 2;
}
```

Exercice 5. Recherche par dichotomie dans un tableau trié.

```
int dichotomie(int elem)
{
  int first = 0;
  int last = N-1;

  while (1) {
    /* cherche elem entre les indices first et last (inclus) */

    int mid = (first+last) / 2;   /* milieu du tableau */

    if (first>last) return 0;     /* absent */

    if (elem==tab[mid]) return 1; /* trouvé */

    if (elem>tab[mid]) first = mid + 1; /* elem dans la moitié haute */
    else               last  = mid - 1; /* elem dans la moitié basse */
  }
}
```