ECUE «Introduction à la programmation » - Session 2

Corrigé

Introduction

Cet énoncé correspond à l'écriture d'un programme C appelé `deuxTroisCinq.c`.

On considère l'ensemble $DTC = \{ n > 0 \mid n = 2^d3^t5^c \text{ avec } d \geq 0, \ t \geq 0, \ c \geq 0 \}$.

$2=2^1 3^0 5^0$, $6=2^1 3^1 5^0$, $10=2^1 3^0 5^1$, $30=2^1 3^1 5^1$, $75=2^0 3^1 5^2$ font partie de `DTC`.

$7=2^0 3^0 5^0 7^1$, $11=2^0 3^0 5^0 11^1$, $22=2^1 3^0 5^0 11^1$, $28=2^2 3^0 5^0 7^1$ ne font pas partie de `DTC`.

Question 1

a) Ecrire une fonction `dePetEaN` prenant un entier `p` et un entier `e` en entrée, et retournant p^e. On utilisera une boucle `for`.

```
int dePetEaN(int p, int e)
{
  int n=1;
  int i;
  for (i=1;i<=e;i++) n *= p;
  return n;
}
```

b) Ecrire une fonction `procedure1` demandant au clavier un entier `p`, un exposant entier `e`, appelant la fonction `dePetEaN` et affichant le résultat sous la forme `x^y = z`.

```
procedure1: lire un nombre p et un exposant e, afficher p puissance e.
p ? 3
e ? 4
3^4 = 81
```

```
void procedure1()
{
  printf("\nprocedure1: ");
  printf("lecture d'un nombre p et d'un exposant e et affichage de p puissance
e.\n");
  int n, p, e;
  printf("p ? "); scanf("%d", &p);
  printf("e ? "); scanf("%d", &e);
  n = dePetEaN(p, e);
  printf("%d^%d = %d\n", p, e, n);
}
```

Question 2

a) Ecrire une fonction `deNetPaE` prenant un entier `n` et un entier `p` en entrée, et retournant l'exposant `e` du nombre `p` dans l'écriture de `n` en puissance de `p`. On utilisera une boucle `while`.

```
int deNetPaE(int n, int p)
{
  int e=0;
  while (n % p==0) { n /= p; e++; }
  return e;
}
```

b) Ecrire une fonction `procedure2` demandant au clavier un entier `n` un nombre entier `p`, appelant la fonction `deNetPaE` et affichant le résultat sous la forme `z est divisible x fois par y`.

```
procedure2: lire un nombre n et un diviseur p, afficher l'exposant .
n ? 440
p ? 2
440 est divisible 3 fois par 2.
```

```
void procedure2()
{
  printf("\nprocedure2: ");
    printf("lecture d'un nombre n et d'un diviseur p et affichage de
l'exposant\n");
  int n, p, e;
  printf("n ? "); scanf("%d", &n);
  printf("p ? "); scanf("%d", &p);
  e = deNetPaE(n, p);
  printf("%d est divisible %d fois par %d.\n", n, e, p);
}
```

Question 3

a) Ecrire une fonction `lireDTC` demandant au clavier 3 entiers valorisant 3 paramètres en sortie `d`, `t` et `c`.

```
void lireDTC(int * d, int * t, int * c)
{
  printf("d ? "); scanf("%d", d);
  printf("t ? "); scanf("%d", t);
  printf("c ? "); scanf("%d", c);
}
```

b) Ecrire une fonction `deDTCaN` avec 3 paramètres entiers `d`, `t` et `c`, en entrée et retournant le nombre `n = 2^d 3^t 5^c`. On utilisera des appels à la fonction `dePetEaN`.

```
int deDTCaN(int d, int t, int c)
{
  int n=1;
  n *= dePetEaN(2, d);
  n *= dePetEaN(3, t);
  n *= dePetEaN(5, c);
}
```

c) Ecrire une fonction `afficheDTC` avec 3 paramètres entiers `d`, `t` et `c` en entrée et affichant `2^d 3^t 5^c`.

```
void afficheDTC(int d, int t, int c)
{
  printf("2^%d 3^%d 5^%d ", d, t, c);
}
```

d) Ecrire une fonction `procedure3` appelant les 3 fonctions `lireDTC`, `deDTCaN` et `afficheDTC`.

```
procedure3: lire 3 exposants d, t, c et afficher le nombre correspondant.
d ? 3
t ? 2
c ? 2
2^3 3^2 5^2 = 1800
```

```
void procedure3()
{
  printf("\nprocedure3: ");
  printf("lecture de 3 exposants, d, t et c de 2, 3 et 5 et affichage du nombre
correspondant.\n");
  int d, t, c;
  lireDTC(&d, &t, &c);
  int n = deDTCaN(d, t, c);
  afficheDTC(d, t, c);
  printf("= %d \n", n);
}
```

Question 4

a) Ecrire une fonction `deNaDTC` prenant un entier `n` en entrée et 3 entiers `d`, `t`, `c` en sortie, exposants de `2`, `3` et `5` dans la décomposition de `n` en facteurs premiers. Cette fonction effectuera des appels à `deNetPaE`.

```
void deNaDTC(int n, int * d, int * t, int * c)
{
  *d = deNetPaE(n, 2);
  *t = deNetPaE(n, 3);
  *c = deNetPaE(n, 5);
}
```

b) Ecrire une fonction `estUnDTC` prenant un entier `n` en entrée et retournant `1` si `n` est de la forme DTC, `0` sinon. Premièrement, pour connaître les nombres `d`, `t` et `c` de `n`, cette fonction appellera `deNaDTC`. Deuxièmement, cette fonction calculera `m` résultat de `deDTCaN` sur `d`, `t` et `c`. `n` est de la forme DTC si et seulement si `m` est égal à `n`.

```
int estUnDTC(int n)
{
  int d, t, c;
  deNaDTC(n, &d, &t, &c);
  if (n==deDTCaN(d, t, c)) {
    // afficheDTC(d, t, c);
    return 1;
  }
  else return 0;
}
```

c) Ecrire une fonction `procedure4` demandant un nombre `n` au clavier, appelant `estUnDTC` et affichant si le nombre `n` est de la forme DTC ou pas. Dans tous les cas, la fonction affichera la forme DTC de `n`. Si `n` n'est pas un DTC, elle affichera en plus `n` divisé par sa forme DTC.

```
procedure4: test si un nombre est de la forme DTC.
n ? 28
28 n'est pas de la forme DTC
2^2 3^0 5^0 7 = 28
```

```
void procedure4()
{
  printf("\nprocedure4: ");
  printf("test si un nombre est de la forme DTC.\n");
  int n, d, t, c;
  printf("n ? "); scanf("%d", &n);
  if (estUnDTC(n)) {
    printf("%d est de la forme DTC.\n", n);
    deNaDTC(n, &d, &t, &c);
    afficheDTC(d, t, c);
  }
  else {
    printf("%d n'est pas de la forme DTC.\n", n);
    deNaDTC(n, &d, &t, &c);
    int m = deDTCaN(d, t, c);
    int q = n / m;
    printf("%d ", q);
    afficheDTC(d, t, c);
    printf("= %d \n", n);
  }
}
```

Question 5

Ecrire une fonction `procedure5` affichant tous les nombres de forme DTC compris entre `1` et `50`.

```
procedure5: affichage des nombres DTC compris entre 1 et 50.
1 2 3 4 5 6 8 9 10 12 15 16 18 20 24 25 27 30 32 36 40 45 48 50
```

```
void procedure5()
{
  printf("\nprocedure5: ");
  printf("affichage des nombres de la forme DTC compris entre 1 et 50.\n");
  int n;
  for (n=1; n<=50; n++) {
    if (estUnDTC(n)==1) printf("%d ", n);
  }
  printf("\n");
}
```

Question 6

a) Ecrire une fonction `pgcdDTC` retournant le pgcd de deux nombres de la forme DTC.

```
int pgcdDTC(int m, int n)
{
  int dm, tm, cm; deNaDTC(m, &dm, &tm, &cm);
  int dn, tn, cn; deNaDTC(n, &dn, &tn, &cn);
  int dd = (dm<dn) ? dm : dn;
  int td = (tm<tn) ? tm : tn;
  int cd = (cm<cn) ? cm : cn;
  return deDTCaN(dd, td, cd);
}
```

b) Ecrire une fonction `ppcmDTC` retournant le ppcm de deux nombres de la forme DTC.

```
int ppcmDTC(int m, int n)
{
  int dm, tm, cm; deNaDTC(m, &dm, &tm, &cm);
  int dn, tn, cn; deNaDTC(n, &dn, &tn, &cn);
  int dd = (dm>dn) ? dm : dn;
  int td = (tm>tn) ? tm : tn;
  int cd = (cm>cn) ? cm : cn;
  return deDTCaN(dd, td, cd);
}
```

c) Ecrire une fonction `procedure6` demandant deux nombres `m` et `n` au clavier, et vérifiant qu'ils sont de la forme DTC et affichant leur pgcd et leur ppcm en appelant `pgcdDTC` et `ppcmDTC`.

« pgcd » signifie Plus Grand Commun Diviseur et « ppcm » signifie Plus Petit Commun Multiple. Si `n1` = $2^{d1}3^{t1}5^{c1}$ et `n2` = $2^{d2}3^{t2}5^{c2}$, alors `pgcd(n1, n2)` = $2^{min(d1,d2)}3^{min(t1,t2)}5^{min(c1,c2)}$ et `ppcm(n1, n2)` = $2^{max(d1,d2)}3^{max(t1,t2)}5^{max(c1,c2)}$

```
procedure6: calcul du pgcd et du ppcm de 2 nombres m et n de forme DTC.
m ? 360
n ? 450
pgcd(360, 450) = 90
ppcm(360, 450) = 1800
```

```
void procedure6()
{
  printf("\nprocedure6: ");
  printf("calcul du pgcd et du ppcm de 2 nombres m et n de forme DTC.\n");
  int m; printf("m ? "); scanf("%d", &m);
  if (estUnDTC(m)==0) {
    printf("desole, %d n'est pas de la forme DTC.\n", m);
    return;
  }
  int n; printf("n ? "); scanf("%d", &n);
  if (estUnDTC(n)==0) {
    printf("desole, %d n'est pas de la forme DTC.\n", n);
    return;
  }
  printf("pgcd(%d, %d) = %d\n", m, n, pgcdDTC(m, n));
  printf("ppcm(%d, %d) = %d\n", m, n, ppcmDTC(m, n));
}
```

Question 7

Ecrire le programme `main` appelant les fonctions `procedure*` des questions 1 à 6.

```
int main()
{
  procedure1();
  procedure2();
  procedure3();
  procedure4();
  procedure5();
  procedure6();
  return 0;
}
```