Algorithmique et Programmation – Examen (durée 3 heures)

CORRECTION de la première session du 9 juin 2021

Remarque : Ce document ne présente que des éléments de correction. Les variantes possibles sont nombreuses.

Exercice 1 : Inclusion mutuelle d'objets communicants

a)

```
1  char inclusion_cercles (int c1x, int c1y, int c1r, int c2x, int c2y, int c2r) {
2    double dist = sqrt((c1x - c2x) * (c1x - c2x) + (c1y - c2y) * (c1y - c2y)) ;
3    char res = 0 ;
4
5    if (dist < c1r && dist < c2r) {
6      res = 3 ;
7    }
8    else if (dist < c1r || dist < c2r) {
9      res = 2 ;
10   }
11   else if (dist - c1r - c2r > 0) {
12     res = 1 ;
13   }
14
15   return res ;
16 }
```

ou encore :

```
1  char inclusion_cercles (int c1x, int c1y, int c1r, int c2x, int c2y, int c2r) {
2    double dist = sqrt((c1x - c2x) * (c1x - c2x) + (c1y - c2y) * (c1y - c2y)) ;
3
4    if (dist < c1r && dist < c2r)
5      return 3 ;
6
7    if (dist < c1r || dist < c2r)
8      return 2 ;
9
10   if (dist - c1r - c2r > 0)
11     return 1 ;
12
13   return 0 ;
14 }
```

b)

```
1  int main () {
2    int d1, d2 ;
3
4    printf ("Donnez␣le␣premier␣rayon␣:␣") ;
5    scanf ("%d", &d1) ;
6
7    printf ("Donnez␣le␣deuxieme␣rayon␣:␣") ;
8    scanf ("%d", &d2) ;
9
10   if (inclusion_cercles (-2, 3, d1, 2, 3, d2) >= 2) {
11     printf ("Les␣objets␣peuvent␣communiquer.\n") ;
12   }
13   else {
14     printf ("Les␣objets␣ne␣peuvent␣pas␣communiquer.\n") ;
15   }
16
17   return 0 ;
18 }
```

ou encore :

```c
int main () {
  int d1, d2 ;
  char communicate ;

  printf ("Donnez le premier rayon : ") ;
  scanf ("%d", &d1) ;

  printf ("Donnez le deuxieme rayon : ") ;
  scanf ("%d", &d2) ;

  communicate = inclusion_cercles (-2, 3, d1, 2, 3, d2) ;

  if (communicate == 3 || communicate == 2) {
    printf ("Les objets peuvent communiquer.\n") ;
  }
  else {
    printf ("Les objets ne peuvent pas communiquer.\n") ;
  }

  return 0 ;
}
```

c)

```c
int main () {
  int bx[8] = { -2, 3, -4,  5, -5, 1,  0, 2 } ;
  int by[8] = {  5, 5, -2, -2,  5, 3, -1, 2 } ;
  int br[8] = {  2, 1,  1,  1,  4, 2,  2, 4 } ;
  int x, y, r, i, cpt2=0, cpt3=0 ;
  char etat ;

  printf ("Donnez l'abscisse : ") ;
  scanf ("%d", &x) ;

  printf ("Donnez l'ordonnee : ") ;
  scanf ("%d", &y) ;

  printf ("Donnez le rayon : ") ;
  scanf ("%d", &r) ;

  for (i=0 ; i<8 ; i++) {
    etat = inclusion_cercles(bx[i], by[i], br[i], x, y, r) ;

    if (etat == 3) {
      cpt3++ ;
    }
    else if (etat == 2) {
      cpt2++ ;
    }
  }

  printf ("Cet objet peut communiquer :\n");
  printf ("en bi-directionnel avec %d objets.\n", cpt3) ;
  printf ("en uni-directionnel avec %d objets.\n", cpt2) ;

  return 0 ;
}
```

E Être capable d'écrire des fonctions et des programmes simples

Exercice 2 : Paquetage d'objets communicants

a)

```
13 char objet_read (TObjet *b, FILE *desc) {
14   int x, y, r ;
15   char res = 0 ;
16
17   if (!feof(desc) && fscanf (desc, "%d␣%d␣%d", &x, &y, &r) == 3) {
18     b->x = x ;
19     b->y = y ;
20     b->r = r ;
21     res = 1 ;
22   }
23
24   return res ;
25 }
```

ou encore :

```
13 char objet_read (TObjet *b, FILE *desc) {
14   char res = 0 ;
15
16   if (!feof(desc) && fscanf (desc, "%d␣%d␣%d", &(b->x), &(b->y), &(b->r)) == 3) {
17     res = 1 ;
18   }
19
20   return res ;
21 }
```

ou encore :

```
13 char objet_read (TObjet *b, FILE *desc) {
14   return !feof(desc) && fscanf (desc, "%d␣%d␣%d", &(b->x), &(b->y), &(b->r)) == 3) ;
15 }
```

b)

```
27 char objet_communicate (TObjet *b1, TObjet *b2) {
28   return inclusion_cercles(b1->x, b1->y, b1->r, b2->x, b2->y, b2->r) ;
29 }
```

D **Être capable de lire dans des fichiers texte et de manipuler une structure de données**

Exercice 3 : Programme d'analyse des communications entre objets communicants

```
1  /* File: main.c */
2  #include <stdio.h>
3  #include <stdlib.h>
4
5  #include "TObjet.h"
6
7  #define FOPEN(_d,_f,_m) {\
8    if (!(_d = fopen (_f, _m))) {\
9    perror("Error:␣cannot␣open␣file.");\
10   exit (1) ;\
11   }}
12
13 #define MALLOC(_p,_t,_n) {\
14   if (!(_p = malloc (_n * sizeof (_t)))) {\
15   perror("Error:␣memory␣allocation␣failed.");\
16   exit (1) ;\
17   }}
18
```

```
19 int main () {
20   TObjet *objets ;
21   int *communique ;
22   int N, i ;
23   char filename[256] ;
24   FILE *desc ;
25
26   int j, cpt=0 ;
27   char etat ;
28
29   printf ("Donnez␣le␣nom␣du␣fichier␣de␣previsions␣:␣") ;
30   scanf ("%s", filename) ;
31
32   FOPEN (desc, filename, "r") ;
33
34   fscanf (desc, "%d", &N) ;
35
36   MALLOC (objets, TObjet, N) ;
37   MALLOC (communique, int, N) ;
38
39   i = 0 ;
40   while (!feof(desc)) {
41     objet_read (objets+i, desc) ;
42     communique[i] = 0 ;
43     i++ ;
44   }
45
46   fclose (desc) ;
47
48   for (i=0 ; i<N ; i++) {
49     for (j=i+1 ; j<N ; j++) {
50       etat = objet_communicate (objets+i, objets+j) ;
51
52       if (etat == 3) {
53         printf ("Ces␣deux␣objets␣peuvent␣communiquer␣en␣bi-directionnel␣:\n") ;
54         objet_print (objets+i) ;
55         objet_print (objets+j) ;
56         communique[i] = 1 ;
57         communique[j] = 1 ;
58       }
59       else if (etat == 2) {
60         printf ("Ces␣deux␣objets␣peuvent␣communiquer␣en␣uni-directionnel␣:\n") ;
61         objet_print (objets+i) ;
62         objet_print (objets+j) ;
63         communique[i] = 1 ;
64         communique[j] = 1 ;
65       }
66     }
67   }
68
69   for (i=0 ; i<N ; i++) {
70     cpt += !communique[i]  ;
71   }
72
73   printf ("%d␣objet%s␣hors␣atteinte␣:\n", cpt, cpt==1 ? "␣est" : "s␣sont") ;
74
75   for (i=0 ; i<N ; i++) {
76     if (!communique[i]) {
77       objet_print (objets+i) ;
78     }
79   }
80
81   return 0 ;
82 }
```

ou encore :

```
19 int main () {
20   TObjet *objets ;
21   int *communique ;
22   int N, i ;
23   char filename[256] ;
24   FILE *desc ;
25
26   int j, cpt=0 ;
27   char etat ;
28
29   printf ("Donnez␣le␣nom␣du␣fichier␣de␣previsions␣:␣") ;
30   scanf ("%s", filename) ;
31
32   FOPEN (desc, filename, "r") ;
33
34   fscanf (desc, "%d", &N) ;
35
36   MALLOC (objets, TObjet, N) ;
37   MALLOC (communique, int, N) ;
38
39   i = 0 ;
40   while (!feof(desc)) {
41     objet_read (&(objets[i]), desc) ;
42     communique[i] = 0 ;
43     i++ ;
44   }
45
46   fclose (desc) ;
47
48   for (i=0 ; i<N ; i++) {
49     for (j=i+1 ; j<N ; j++) {
50       etat = objet_communicate (&(objets[i]), &(objets[j])) ;
51
52       if (etat == 3) {
53         printf ("Ces␣deux␣objets␣peuvent␣communiquer␣en␣bi-directionnel␣:\n") ;
54         objet_print (&(objets[i])) ;
55         objet_print (&(objets[j])) ;
56         communique[i] = 1 ;
57         communique[j] = 1 ;
58       }
59       else if (etat == 2) {
60         printf ("Ces␣deux␣objets␣peuvent␣communiquer␣en␣uni-directionnel␣:\n") ;
61         objet_print (&(objets[i])) ;
62         objet_print (&(objets[j])) ;
63         communique[i] = 1 ;
64         communique[j] = 1 ;
65       }
66     }
67   }
68
69   for (i=0 ; i<N ; i++) {
70     cpt += !communique[i]  ;
71   }
72
73   printf ("%d␣objet%s␣hors␣atteinte␣:\n", cpt, cpt==1 ? "␣est" : "s␣sont") ;
74
75   for (i=0 ; i<N ; i++) {
76     if (!communique[i]) {
77       objet_print (&(objets[i])) ;
78     }
79   }
80
81   return 0 ;
82 }
```

C Être capable d'utiliser des pointeurs et des structures de données