ED 1 Système Informatique B
Système = Environnement

I Environnement d'un programme

Soit le programme "c" suivant (calcul_somme.c):

```
1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <math.h>
4.
5. int TAILLE_ALLOC = 1024*1024*1024;
6.
7. // ======================A=========================
8. int calcul_trigo( int n)
9. {
10.     int i, res = 0;
11.     for(i=0; i<=n; i++)
12.         res+=3* sin(i);
13.     return res;
14.}
15.
16.// ======================B=========================
17.int calcul_carre( int n)
18.{
19.     int i, res = 0;
20.     for(i=0; i<=n; i++)
21.         res+=i*i;
22.     return res;
23.}
24.
25.// ======================C=========================
26.int main(int argn, char *argv[], char *env[])
27.{
28.  int nb_pas;
29.  int i;
30.  char *zone;
31.
32.  for (i=0; env[i] != NULL; i++)
33.    //for (i=0; i < 10; i++)
34.    printf("VARIABLE ENVIRONNEMENT:%s\n", env[i]);
35.
36.  //sleep(10);
37.  zone = (char *) malloc(TAILLE_ALLOC);
38.
39.  if (argn != 2){
40.    printf("il faut 1 argument : le nb de pas de calcul\n");
41.    exit (-2);
42.  }
43.
44.  sscanf(argv[1], "%d", &nb_pas);
45.
46.  printf("\n\n\n\nnombre de pas = %d \n", nb_pas);
47.  printf("somme trigo = %d\n", calcul_trigo(nb_pas));
48.  printf("somme carre = %d\n", calcul_carre(nb_pas));
49.
50.  printf("Adresse de   main         = %09lx\n", main);
51.  printf("Adresse de   TAILLE_ALLOC = %09lx\n", &TAILLE_ALLOC);
52.  printf("Adresse de   nb_pas       = %09lx\n", &nb_pas);
53.  printf("Adresse de   zone         = %09lx\n", zone);
54.  printf("Adresse de   argv[1]      = %09lx\n", argv[1]);
55.
56.  // ======================D=========================
57.  sleep(5);
58.  exit(nb_pas);
59.}
```

Q1.a) *A quoi correspondent la fonction main et ses arguments (int argn, char *argv[], char *env[]) ?*

Q1.b) *Définir les différentes composantes de ce programme?*
Voici le début d'une exécution de ce programme. (calcul_somme 100)

```
VARIABLE ENVIRONNEMENT:HOST=bacchus
VARIABLE ENVIRONNEMENT:TERM=xterm
VARIABLE ENVIRONNEMENT:SHELL=/bin/bash
VARIABLE ENVIRONNEMENT:USER=root
VARIABLE_ENVIRONNEMENT:PATH=/usr/local/bin:/usr/bin:/usr/X11R6/bin:/bin
...
nombre de pas = 100
somme trigo = -2
somme carre = 338350
Adresse de   main         = 0080485be
Adresse de   TAILLE_ALLOC = 00804a030
Adresse de   nb_pas       = 0bf9d4048
Adresse de   zone         = 077e31008
Adresse de   argv[1]      = 0bf9d612f
```

Q1.c) *A quoi correspondent ces variables d'environnement et à quoi servent-elles ?*
Q1.d) *Qui initialise ces variables ?*

II Compilation d'un programme

Q2.a) *Expliquer la chaine de production d'un programme.*

On exécute les commandes suivantes :

```
#gcc -c calcul_somme.c
#ls -l calcul_somme.o
     -rw-r--r-- 1 root root 2132 Feb 21 15:55 calcul_somme.o
```

Q2.b) *Rappeler les différentes composantes d'un ficher "objet".*
Q2.c) *Expliquer le résultat de la commande suivante:*

```
#nm calcul_somme.o
00000000  D TAILLE_ALLOC
00000059  T calcul_carre
00000000  T calcul_trigo
          U exit
0000008a  T main
          U malloc
          U printf
          U sin
          U sleep
          U sscanf
```

III Edition de liens

III.1Edition de liens statique

On exécute les commandes suivantes :

```
#gcc -static calcul_somme.c -lm -o calcul_somme.st
#ls -l calcul_somme.st
-rw-rwxr-xr-x 1 root root 546835 Feb 22 14:01 calcul_somme.st
```

Q3.a) *Expliquer la différence de taille avec fichier objet obtenu à la Q2.a ?*
Q3.b) *Expliquer le resultat des commandes suivantes :*

```
#ldd calcul_somme.st
      not a dynamic executable
#nm  calcul_somme.st| egrep 'printf|sscanf|exit|calcul|malloc|sin'
...
080482a1 T calcul_carre
08048248 T calcul_trigo
08048bc0 T exit
08048e40 T printf
08048e70 T sscanf
0804efe0 T malloc
08048440 T sin
```

III.2Edition de liens dynamique

On refait les même manipulations et on obtient;

```
#gcc calcul_somme.c -lm -o calcul_somme
#ls -l calcul_somme
   -rwxr-xr-x 1 root root 9819 Feb 21 15:37 calcul_somme

#nm calcul_somme| egrep 'printf|scanf|exit|calcul'
   0804858d T calcul_carre
   08048534 T calcul_trigo
            U exit@@GLIBC_2.0
            U printf@@GLIBC_2.0
            U sscanf@@GLIBC_2.0

#ldd calcul_somme
     libm.so.6 => /lib/libm.so.6 (0xb7eb1000)
     libc.so.6 => /lib/libc.so.6 (0xb7d83000)
     /lib/ld-linux.so.2 (0xb7ef6000)
```

Q3.c) *Expliquer les différences entre les deux éditions de liens ?*

On s'interesse à la librairie partagée "libc.so.6 ". On effectue donc la commande suivante :

```
#nm /lib/libc.so.6 | egrep 'printf|sscanf|exit|malloc'
...
0002baf0 T exit
00044540 T printf
00053510 T sscanf
00068040 T malloc
...
```

Q3.d) *Pourquoi dit on que cette librairie est partagée et expliquer comment le système peut utiliser cette librairie ?*

IV Lancement du programme

On se propose d'étudier le démarrage de notre programme dans le système. Pour cela on exécute les deux commandes suivantes :

```
#strace calcul_somme.st 100
                    = ?
    1. execve("./calcul_somme.st", ["./calcul_somme.st", "100"], [/* 85 vars */]) = 0
    2. fstat64(1, {st_mode=S_IFREG|0644, st_size=3742, ...}) = 0      --> Sortie Out
    3. mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
       0xb7f7c000                                  ----->Plages buffers out
    4. write(1, "VARIABLE ENVIRONNEMENT:LESSKEY=/"..., 4096
    5.
    6. ...
    7. mmap2(NULL, 1073745920, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
       0x77f7b000
    8. write(1, "\n\n\n\n", 4
    9. ...
    10.nanosleep({5, 0}, {5, 0})            = 0
    11.exit_group(100)                        = ?

#strace calcul_somme 100
    1. execve("./calcul_somme", ["./calcul_somme", "100"], [/* 85 vars */]) = 0
    2. mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
       0xb7fa4000                                                -> plage temp0
    3. fstat64(3, {st_mode=S_IFREG|0644, st_size=117013, ...}) = 0
    4. mmap2(NULL, 117013, PROT_READ, MAP_PRIVATE, 3, 0) = 0xb7f87000 --> plage temp
    5. close(3)                                 = 0
    6. open("/lib/libm.so.6", O_RDONLY)         = 3
    7. read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0`5\0\000"..., 512) = 512
    8. fstat64(3, {st_mode=S_IFREG|0755, st_size=190963, ...}) = 0
    9. mmap2(NULL, 151680, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
       0xb7f61000                                                ------>Plages libm
    10.mmap2(0xb7f85000, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3,
       0x23) = 0xb7f85000                                        ------>ajout Plages libm
    11.close(3)                                 = 0
    12.open("/lib/libc.so.6", O_RDONLY)         = 3
    13.read(3, "\177ELF\1\1\1\0\0\0\0\0\0\0\0\0\3\0\3\0\1\0\0\0\340`\1"..., 512) = 512
    14.fstat64(3, {st_mode=S_IFREG|0755, st_size=1491141, ...}) = 0
    15.mmap2(NULL, 1234372, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) =
```

```
   0xb7e33000                                          ----->Plages libc
16.mmap2(0xb7f5b000, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE,
   3, 0x128) = 0xb7f5b000                              ----->ajout Plages libm
17.mmap2(0xb7f5e000, 9668, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1
   , 0) = 0xb7f5e000                                    ----->ajout Plages libm
18.close(3)                                 = 0
19.mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
   0xb7e32000
20.munmap(0xb7f87000, 117013)               = 0    --> restitution plage temp
21.fstat64(1, {st_mode=S_IFREG|0644, st_size=8083, ...}) = 0   ---> Sortie Out
22.mmap2(NULL, 4096, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
   0xb7fa3000                                         ----->Plages buffers out
23.write(1, "VARIABLE ENVIRONNEMENT:LESSKEY=/"..., 4096
24....
25.mmap2(NULL, 1073745920, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) =
   0x77e31000                                                ----->Plages Tas
26.write(1, "\n\n\n\n", 4
27....
28.nanosleep({5, 0}, {5, 0})              = 0
29.exit_group(100)                          = ?
```

Q4.a) *Rappeler le rôle de la primitive execve; Qui l'exécute et comment ?*

Q4.b) *Détailler les différentes étapes mises en évidence par la commande "strace".*

V Exécution

On s'intéresse à l'exécution des deux versions de notre exécutable (édition de liens statique ou dynamique). Un outils nous permet de retrouver les zones mémoires allouées par le système à chacun des processus. De plus on connait les adresses des symboles suivants :

symbole	version statique	pour dynamique
main	0080482d2	0080485be
nb_pas (local main)	0bf8534f8	0bf9d4048
TAILLE_ALLOC (global)	0080bb00c	00804a030
argv[1]	0bf85412c	0bf9d612f
zone	077f7b008	077e31008

Version statique----------># cat /proc/4602/maps

```
Plages d'adresse        Taille          Droits  Offset  Device  Inode   Fichier
08048000-080ba000  456k         r-xp    00000000   08:02  699122   calcul_somme.st
080ba000-080bc000  8k           rw-p    00071000   08:02  699122   calcul_somme.st
080bc000-080e0000  144k         rw-p    080bc000   00:00  0        [heap]
77f7b000-b7f7d000  1048584k     rw-p    77f7b000   00:00  0
b7f7d000-b7f7e000  4k           r-xp    b7f7d000   00:00  0        [vdso]
bf840000-bf855000  84k          rw-p    bf840000   00:00  0        [stack]
```

Version dynamique----------># cat /proc/4802/maps

```
Plages d'adresse        Taille       Droits  Offset  Device  Inode     Fichier
08048000-08049000 4k         r-xp 00000000 08:02   699172    calcul_somme
08049000-0804a000 4k         r--p  00000000 08:02   699172    calcul_somme
0804a000-0804b000 4k         rw-p 00001000 08:02   699172    calcul_somme
77e31000-b7e33000 1048584k rw-p 77e31000  00:00   0
b7e33000-b7f5b000 1184k      r-xp   00000000  08:02 1154195   /lib/libc-2.5.so
b7f5b000-b7f5c000  4k        r--p   00128000 08:02   1154195   /lib/libc-2.5.so
b7f5c000-b7f5e000  8k        rw-p  00129000 08:02  1154195   /lib/libc-2.5.so
b7f5e000-b7f61000  12k       rw-p b7f5e000 00:00   0
b7f61000-b7f85000  144k      r-xp  00000000 08:02  1154203   /lib/libm-2.5.so
b7f85000-b7f87000  8k        rw-p 00023000 08:02  1154203   /lib/libm-2.5.so
b7fa3000-b7fa5000  8k        rw-p  b7fa3000 00:00   0
b7fa6000-b7fc1000  108k      r-xp  00000000 08:02 1154188   /lib/ld-2.5.so
b7fc1000-b7fc3000  8k        rw-p  0001a000 08:02 1154188   /lib/ld-2.5.so
bf9c2000-bf9d8000  88k       rw-p bf9c2000 00:00 0           [stack]
```

Q5.a) *Comparer les deux versions (taille, nombre de zones, permissions) et faites le lien avec le résultat de la commande "strace".*

Q5.b) *Quel est la taille des programmes en mémoire pour chacune des versions (l'unité des tailles données plus haut est de 1Ko). Expliquer la différence.*