

Systemes 2 — TD

Semestre 4

Table des matières

1	Introduction	4
1.1	Exercice 1	4
1.2	Exercice 2	4
2	Processus	6
2.1	Exercice 3	6
2.2	Exercice 4	6
2.3	Exercice 5	6
2.4	Exercice 6	6
2.5	Exercice 7	6
3	Gestion de la mémoire : mémoire virtuelle et allocation non contiguë	7
4	Structure interne du système de fichier d'Unix	8
5	Primitives Unix (POSIX.1) de manipulation des fichiers	9

Introduction

1.1 Exercice 1

```
1 #include <stdio.h>
2
3 int main(int argc, char** argv) {
4     int i;
5     for(i=1; i < argc; ++i) {
6         printf("%s\n", argv[i]);
7     }
8
9     return 0;
10 }
```

Listing 1.1 – Exercice 1 – Version portable

```
1 #define _POSIX_C_SOURCE 1
2
3 #include <stdio.h>
4
5 int main(int argc, char** argv, char** envp) {
6     int i;
7     printf("Argument :\n");
8     for(i=1; i < argc ; ++i) {
9         printf("argv [%d]=%s\n", i, argv[i]);
10    }
11    i=0;
12    while(envp[i] != NULL) {
13        printf("%s\n", envp[i++]);
14    }
15
16    return 0;
17 }
```

Listing 1.2 – Exercice 1 – Version Unix

1.2 Exercice 2

```
1 #define _POSIX_C_SOURCE 1
2
3 #include <stdio.h>
4 #include <string.h>
5
6 int main(int argc, char** argv) {
7     if(argv != 1 || !(strlen(argv[1]))) {
8         return 1;
9     }
10
11     printf("%s = %s\n", argv[1], getenv(argv[1]));
```

12 | }

Listing 1.3 – Exercice 2

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main(int argc, char** argv) {
5      char *valeur;
6
7      if(argc != 2) {
8          fprintf(stderr, "Usage : %s variable\n", argv[0]);
9          return (1);
10     }
11
12     valeur = getenv(argv[1]);
13
14     if(valeur == NULL) {
15         fprintf(stderr, "Variable %s inconnue \n", argv[1]);
16         return (2);
17     }
18
19     printf("%s=%s", argv[1], valeur);
20
21     return 0;
22 }
```

Listing 1.4 – Exercice 2 – Correction

Processus

2.1 Exercice 3

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(int argc, char** argv) {
5     int pid;
6
7     switch(pid=fork()) {
8         case -1:
9             exit(1);
10            break;
11         case 0: //fils
12             printf("Exécuté par le fils -- Père: %d\nFils: %d\n", getppid(), getpid());
13             break;
14         default: //père
15             printf("Exécuté par le père -- père: %d\n", getpid());
16
17     }
18 }
```

Listing 2.1 – Exercice 3

2.2 Exercice 4

2.3 Exercice 5

2.4 Exercice 6

2.5 Exercice 7

Gestion de la mémoire : mémoire virtuelle et allocation non contiguë

CHAPITRE

3

Structure interne du système de fichier d'Unix

4

Primitives Unix (POSIX.1) de manipulation des fichiers
