Université Toulouse III – Paul sabatier
L2 Informatique

Systèmes 2 — TD

Semestre 4

Toulouse, le 31 janvier 2013

Table des matières

Chapitre 1

Introduction

1.1 Exercice 1

```c
#include <stdio.h>

int main(int argc, char** argv) {
  int i;
  for(i=1; i < argc; ++i) {
    printf("%s\n", argv[i]);
  }

  return 0;
}
```

Listing 1.1 – Exercice 1 – Version portable

```c
#define _POSIF_C_SOURCE 1

#include <stdio.h>

int main(int argc, char** argv, char** envp) {
  int i;
  printf("Argument :\n");
  for(i=1; i < argc ; ++i) {
    printf("argv[%d]=%s\n", i, argv[i]);
  }
  i=0;
  while(envp[i] != NULL) {
    printf("%s\n", envp[i++]);
  }

  return 0;
}
```

Listing 1.2 – Exercice 1 – Version Unix

1.2 Exercice 2

```c
#define _POSIX_C_SOURCE 1

#include <stdio.h>
#include <string.h>

int main(int argc, char** argv) {
  if(argv != 1 || !(strlen(argv[1]))) {
    return 1;
  }

  printf("%s = %s\n", argv[1], getenv(argv[1]));
```

```
}
```

Listing 1.3 – Exercice 2

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char** argv) {
  char *valeur;

  if(argc != 2) {
    fprintf(stderr, "Usage : %s variable\n", argv[0]);
    return (1);
  }

  valeur = getenv(argv[1]);

  if(valeur == NULL) {
    fprintf(stderr, "Variable %s inconnue \n", argv[1]);
    return (2);
  }

  printf("%s=%s", argv[1], valeur);

  return 0;
}
```

Listing 1.4 – Exercice 2 – Correction

Chapitre 2

Processus

2.1 Exercice 3

```c
#define _POSIX_C_SOURCE 1
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <sys/types.h>

int main(int argc, char** argv) {
  pid_t pid;
  switch(pid = fork()) {
    case -1:
      perror("fork");
      exit(1);
    case 0: //fils
      printf("Executé par le fils\n");
      printf("PID du père: %d\n", (int)getppid());
      printf("PID du fils %d\n\n", (int)getpid());
      break;
    default: //père
      printf("Executé par le père\n");
      printf("PID du père: %d\n", (int)getpid());
      break;
  }

  return 0;
}
```

Listing 2.1 – Exercice 3

2.2 Exercice 4

```c
execlp("date", "date", NULL);
```

Le programme ci-dessus va exécuter le programme `./date` avec l'argument *date*.

2.3 Exercice 5

2.4 Exercice 6

2.5 Exercice 7

Chapitre 3

Gestion de la mémoire : mémoire virtuelle et allocation non contiguë

Chapitre 4

Structure interne du système de fichier d'Unix

Chapitre 5

Primitives Unix (POSIX.1) de manipulation des fichiers