TD 1 : PDS

1 TD 1 : Opérations de base sur un fichier

1.1 La commande “cat”

On propose ici d’implémenter la commande Unix `cat` pour afficher le contenu d’un fichier. Notre commande `mcat` prendra en entrée sur la ligne de commande un nom de fichier, ouvrira le fichier, lira son contenu et l’affichera sur la sortie standard.

Question 1 *Utiliser les appels de la librairie standard.* Dans ce première exercice, on utilisera la fonction `fopen()` pour ouvrir un fichier, la fonction `fgetc()` pour lire un caractère, la fonction `printf()` pour l’afficher sur la sortie standard, et la fonction `fclose()` pour fermer le fichier.

Pour obtenir les paramètres de la ligne de commande, et donc le nom du fichier, on utilise les paramètres `argc` et `argv[]` de la fonction `main()`.

N’oubliez pas de vérifier que le fichier existe et qu’il est lisible. ■

```
int mcat(const char * filename) {
    FILE* f = 0; char c;
    if ((f = fopen(filename, "r")) == NULL) {
        printf("impossible d'ouvrir %s", filename);
        return -1;
    }
    while ((c=fgetc(f)) != EOF) {
        printf("%c", c);
    }
    fclose(f);
    return 0;
}
int main(int argc, char * argv[]) {
    if (argc < 2)
        return -1;
    return mcat(fd);
}
```

Question 2 *Utilises les appels POSIX* Maintenant, on utilisera les appels système `open()`, `close()`, `read()` et `write()`.

L’appel système `read()` demande l’adresse d’un tampon mémoire et la taille de ce tampon. Nous allons utiliser un tampon de taille 16 octets :

```
#define TAILLE 16
char tampon[TAILLE];
```

La taille peut varier en fonction de nos besoin : plus grand est le tampon, plus d’octets on lit avec un seul appel et donc on nécessitera de moins d’appels.

On discutera de l’importance de la taille du tampon dans le prochain TD. ■

```
int mcat_nostd(char * filename){
    char tampon[TAILLE]; int sz;
   int fd = open(filename, O_RDONLY);
   if (fd<0){
        perror("open"); exit(-1);
  }
  while ((sz = open(fd, tampon, TAILLE) <= 0){
        write(1, tampon, sz);
  }
  close(fd);
  Return 0;
}
```

Question 3 *Écrire un tableau d'entiers en binaire* Étant donnée un tableau d'entiers de taille `n`, écrire ce tableau sur un fichier en format binaire.

```c
#define TAILLE 256
int tableau[TAILLE];
int n;

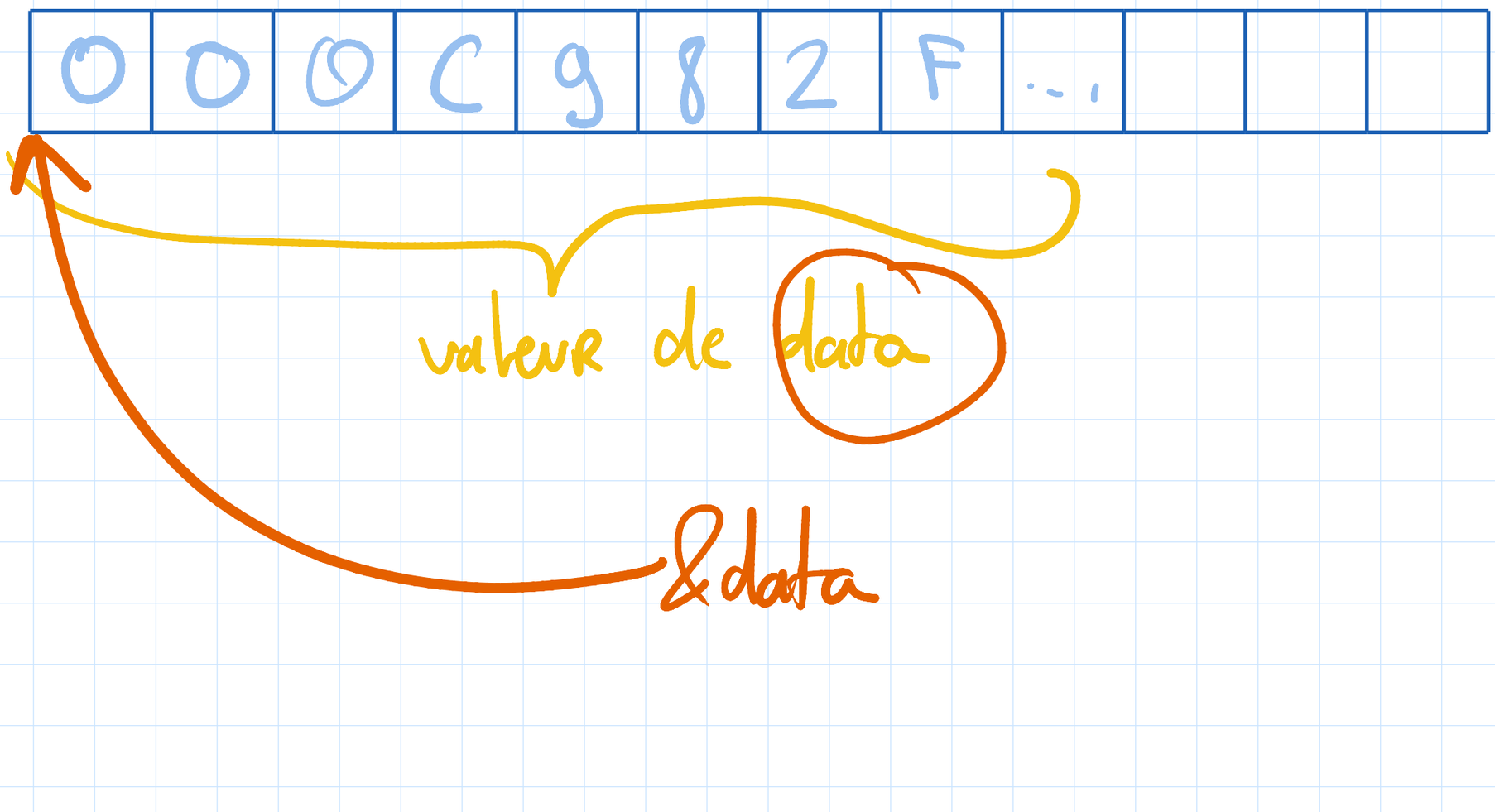
// remplissage tableau ...
// n contient le nombre d'entiers présents dans le tableau
// n <= TAILLE

int f = open("tableau", O_WRONLY | O_CREAT, 0x777);
// TODO: sauvegarder le tableau sur le fichier ...
close(f);
```

0o777.

■

0x4267...



```
char * a = (char *) &data;
```