LINUX - LES ENTRÉES-SORTIES
PROGRAMMATION DES ENTRÉES-SORTIES

MINISTÈRE DE L'ÉDUCATION NATIONALE, DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE

Entrée et sortie standard

- L'écran ou le terminal

```
$ cat fichier.txt
$ abcdefgf
$
```

- Le clavier

- Redirection des entrées-sorties standards

Redirection de l'entrée standard :
sort **<** listing.txt
Redirection de la sortie standard :
ls -al **>** listing.txt
Redirection de la sortie d'erreur :
cp **2>** error.txt

Les fonctions les plus simples

```
# include <stdio.h>

int getchar( )


int putchar(int c)
```

La fonctions **getchar** renvoie le caractère suivant de l'entrée : clavier (par défaut), fichier, tube…

getchar retourne EOF (End Of File) en fin de fichier ou en cas d'erreur

La fonctions **putchar** écrit le caractère c sur la sortie standard, le terminal par défaut

Exemple d'utilisation

```
#include <stdio.h>
#include <ctype.h>

main() {
 int c = 0;
 while (c != EOF) {
  c = getchar();
  if (isascii(c) && (isprint(c) || c=='\n' || c=='\t' || c==' ')) {
   putchar(c);
  }
 }
}
```

Exemple d'utilisation

Tests de caractères dans ctype.h

isalpha(c)	Alphabétique a-z A-Z
isupper(c)	Majuscule A-Z
islower(c)	Minuscule a-z
isdigit(c)	Chiffre 0-9
isxdigit(c)	Hexadécimal 0-9 a-f A-F
isalnum(c)	Aphanumérique = Alphabétique ou Chiffre
isspace(c)	Caractère d'espacement, tabulation…
ispunct(c)	Caractère de ponctuation
iscntrl(c)	Caractère de contrôle
isprint(c)	Imprimable
isascii(c)	Caractère ASCII

Conversions dans ctype.h

toascii(c)	convertir c en ASCII 7 bits
tolower(c)	convertir c en minuscule
toupper(c)	convertir c en majuscule

Les arguments d'un programme

```
main( argc, argv )

        int argc;

        char *argv[];
```

argc est un compteur qui renvoie le nombre de mots sur la ligne de commande

argv est un pointeur sur un tableau dont chaque élément est un pointeur vers un tableau de caractères

argv[0] est le nom de la commande
Les arguments sont donc argv[1], argv[2]…

Exemple d'utilisation

```
#include <stdio.h>
#include <ctype.h>

main(argc, argv)
  int argc;
  char *argv[];
{
  int c = 0, minus = 0;
  if (argc>1 && strcmp(argv[1], "-m")==0) minus = 1;
  while (c != EOF) {
    c = getchar();
    if (isascii(c) && (isprint(c) || c=='\n' || c=='\t' || c==' ' )) {
      if (minus) {
        putchar(tolower(c));
      }
      else {
        putchar(c);
      }
    }
  }
}
```

Exemple d'utilisation

Quelques exemples de fonctions de traitement de chaînes

strcat(dest,src)	concatène la chaîne src à la suite de dest
strncat(dest,src,n)	concatène au plus n caractères de la chaîne src à la suite de dest
strcpy(to,from)	copie une chaîne de caractères d'une zone à une autre
strncpy(to,from,n)	copie au plus *n* caractères d'une chaîne d'une zone à une autre
strcmp(src1,src2)	compare deux chaînes numériquement
strncmp(src1,src2,n)	compare les *n* premiers octets au plus de deux chaînes
strlen(src)	retourne la longueur d'une chaîne caractères
strchr(src,c)	cherche un caractère c dans la chaîne src et renvoie un pointeur sur ce caractère, en cherchant depuis le début
strrchr(src)	idem que strchr, recherche à partir de la fin
atoi(src)	retourne la valeur entière de src
atof(src)	retourne la valeur réelle de src
malloc(n)	retourne un pointeur sur une zone de n octets, \0 sinon

L'accès aux fichiers

Pour pouvoir être lu, un fichier doit être ouvert avec la fonction fopen

```
FILE *fp;

char *nom;
char *mode;

fp = fopen(nom, mode);
```

fp pointeur sur le fichier

Nom du fichier

Mode d'ouverture :
r (read) lecture
w (write) écriture
a (append) ajout

fopen retourne un pointeur sur le fichier

Retourne NULL en cas d'erreur

Les fonctions les plus simples

```
# include <stdio.h>

c = getc(fp);


putc(c, fp)
```

La fonctions **getc** place dans c le prochain caractère du fichier pointé par fp

La valeur est EOF (End Of File) en fin de fichier ou en cas d'erreur

La fonctions **putc** écrit le caractère c dans le fichier pointé par fp

Retourne EOF en cas d'erreur

Exemple d'utilisation

```
#include <stdio.h>
#include <ctype.h>

affiche(FILE *fp) {
  int c = 0;
  while (c != EOF) {
    c = getc(fp);
    if (isascii(c) && (isprint(c) || c=='\n' || c=='\t' || c==' ' ))
        putchar(c);
  }
}

main(int argc, char *argv[]) {
  FILE *fp;
  if (argc==1) {
    printf("%s\n", "Il manque un argument : le nom du fichier !");
  }
  else {
    fp = fopen(argv[1], "r");
    affiche(fp);
  }
}
```

Exemple d'utilisation

Quelques définitions de <stdio.h>

stdin	Entrée standard
stdout	Sortie standard
stderr	Sortie d'erreur standard
EOF	Fin de fichier (souvent -1)
NULL	Pointeur invalide (souvent 0)
FILE	Pour déclarer les pointeurs de fichiers
BUFSIZE	Taille des tampons d'entrée–sortie (souvent 512 ou 1024 octets)
getc(fp)	Retourne un caractère de fp
getchar()	Identique à getc(stdin)
putc(c,fp)	Ecrit le caractère c dans fp
putchar(c)	Identique à putc(c,stdout)
feof(fp)	Différent de 0 si fin de fichier atteinte
ferror(fp)	Différent de 0 si erreur sur le fichier
fileno(fp)	Descripteur (de type entier) de l'argument fp
fclose(fp)	Ferme le fichier fp

Après les caractères… les chaînes…

```
# include <stdio.h>

fputs(s, fp);


fgets(s, n, fp)
```

La fonctions **fputs** écrit la chaîne s dans le fichier pointé par fp

La valeur est EOF (End Of File) en fin de fichier ou en cas d'erreur

La fonctions **fgets** lit au moins n caractères (incluant le marqueur de fin de chaine '\0') dans le fichier pointé par fp et retourne la chaîne dans s

Retourne NULL en fin de fichier

Exemple d'utilisation

```
#include <stdio.h>

main() {
  FILE *fpIn, *fpOut;
  char chaine[11]; // ajouter un octet de plus pour le marqueur
                   // de fin de chaîne '\0'

  fpIn = fopen("dataIn.txt", "r");
  fgets(chaine, 11, fpIn); // lire 10 caractères dans le fichier
  fclose(fpIn);

  fpOut = fopen("dataOut.txt", "w");
  fputs(chaine, fpOut);
  fputs("\n", fpOut);
  fclose(fpOut);
}
```

Exemple d'utilisation