

FR2I

Formation en Réseaux Internationaux d'Ingénieurs



















LINUX - LES ENTRÉES-SORTIES PROGRAMMATION DES ENTRÉES-SORTIES



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



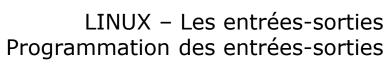
Entrée et sortie standard

L'écran ou le terminal

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

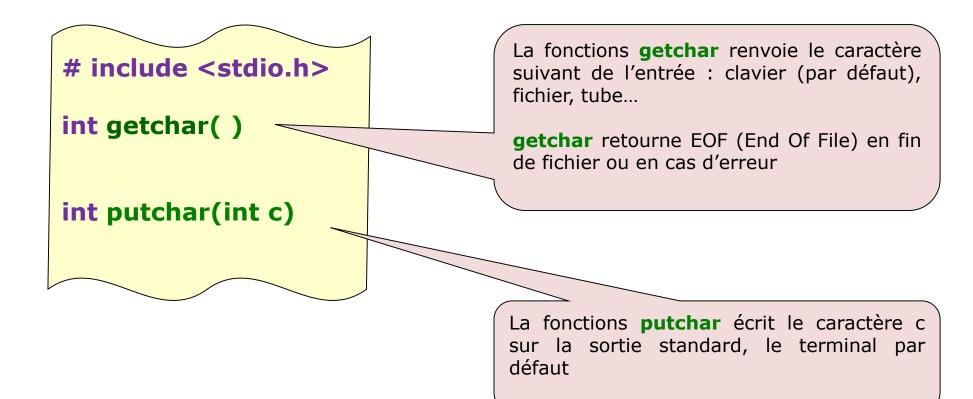
- Le clavier
- Redirection des entrées-sorties standards

```
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

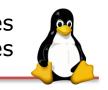












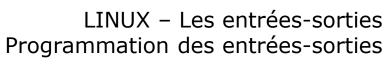
```
#include <stdio.h>
#include <ctype.h>
main() {
int c = 0;
while (c != EOF) {
 c = getchar();
 if (isascii(c) && (isprint(c) || c=='\n' || c=='\t' || c==' ')) {
 putchar(c);
```



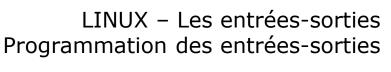














Tests de caractères dans ctype.h

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



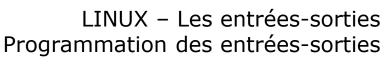
LINUX – Les entrées-sorties Programmation des entrées-sorties

Conversions dans ctype.h

toascii(c) convertir c en ASCII 7 bits

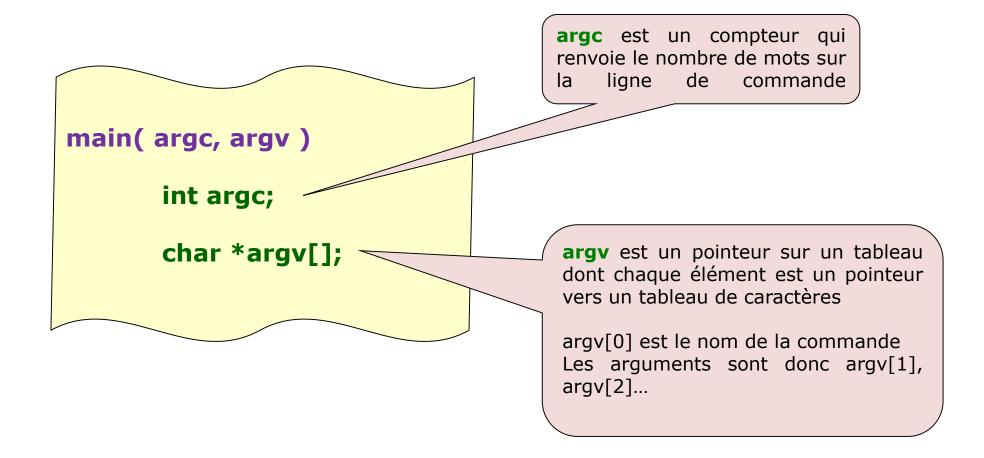
tolower(c) convertir c en minuscule

toupper(c) convertir c en majuscule





Les arguments d'un programme











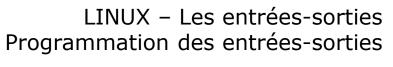
```
#include <stdio.h>
#include <ctype.h>
main(argc, argv)
int argc;
char *argv[];
 int c = 0, minus = 0;
 if (argc>1 && strcmp(argv[1], "-m")==0) minus = 1;
 while (c != EOF) {
  c = getchar();
  if (isascii(c) && (isprint(c) || c=='\n' || c=='\t' || c==' ' )) {
   if (minus) {
    putchar(tolower(c));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













LINUX – Les entrées-sorties Programmation des entrées-sorties



Quelques exemples de fonctions de traitement de chaînes

strcat(dest,src)

strncat(dest,src,n)

strcpy(to,from)

strncpy(to,from,n)

strcmp(src1,src2)

strncmp(src1,src2,n)

strlen(src)

strchr(src,c)

strrchr(src)

atoi(src)

atof(src)

malloc(n)

concatène la chaîne src à la suite de dest

concatène au plus n caractères de la chaîne src à la suite de dest

copie une chaîne de caractères d'une zone à une autre

copie au plus *n* caractères d'une chaîne d'une zone à une autre

compare deux chaînes numériquement

compare les *n* premiers octets au plus de deux chaînes

retourne la longueur d'une chaîne caractères

cherche un caractère c dans la chaîne src et renvoie un pointeur sur ce caractère, en cherchant

depuis le début

idem que strchr, recherche à partir de la fin

retourne la valeur entière de src

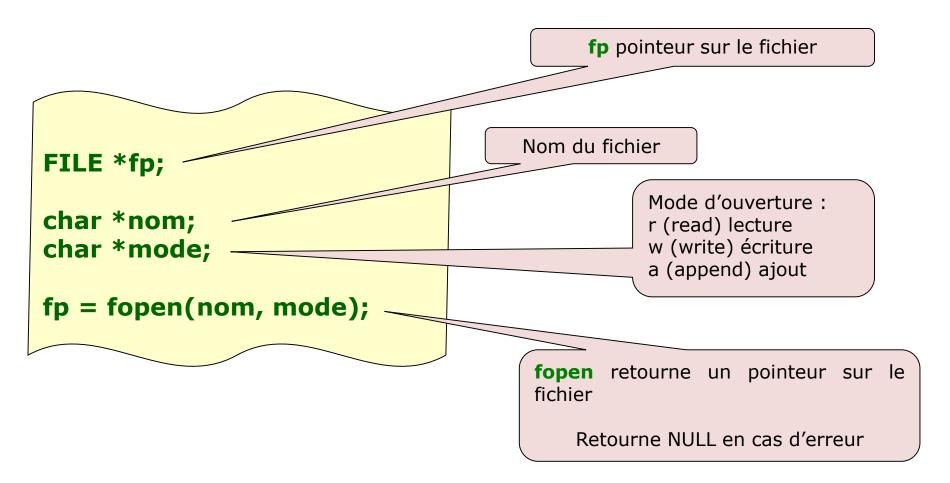
retourne la valeur réelle de src

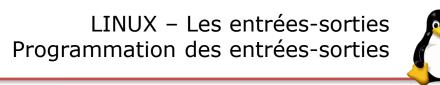
retourne un pointeur sur une zone de n octets, \0 sinon



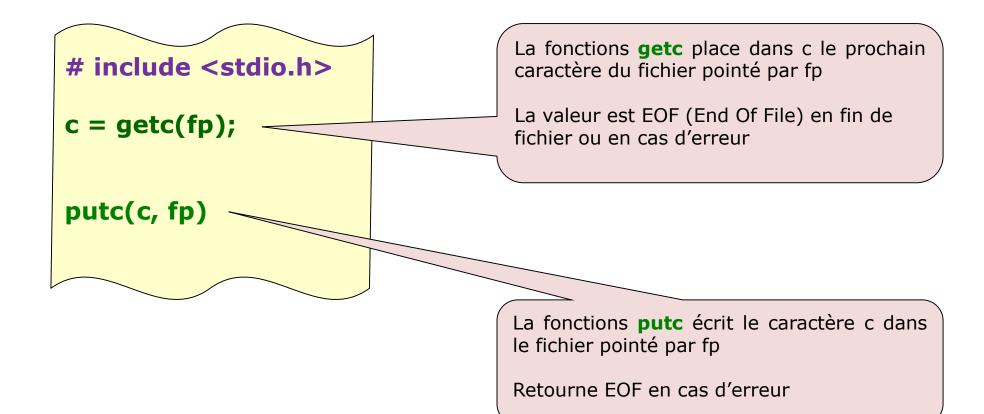
L'accès aux fichiers

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







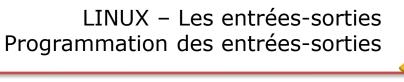












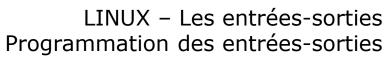
```
#include <stdio.h>
#include <ctype.h>
affiche(FILE *fp) {
 int c = 0;
 while (c != EOF) {
  c = getc(fp);
  if (isascii(c) && (isprint(c) || c=='\n' || c=='\t' || c==' '))
    putchar(c);
main(int argc, char *argv[]) {
 FILE *fp;
 if (argc==1) {
  printf("%s\n", "Il manque un argument : le nom du fichier !");
 else {
  fp = fopen(argv[1], "r");
  affiche(fp);
```



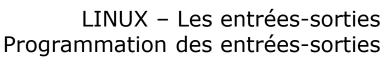














Quelques définitions de <stdio.h>

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)

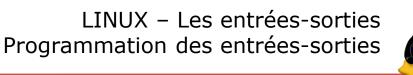
getc(fp) Retourne un caractère de fp

getchar() Identique à getc(stdin)

feof(fp) Différent de 0 si fin de fichier atteinte **ferror(fp)** Différent de 0 si erreur sur le fichier

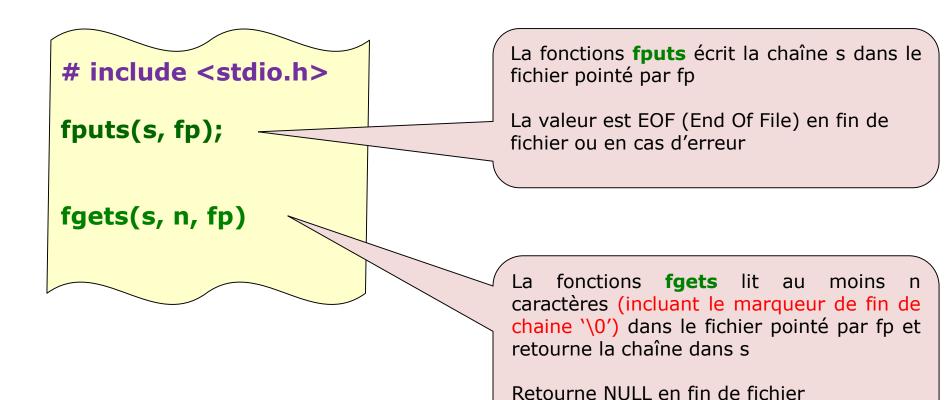
fileno(fp) Descripteur (de type entier) de l'argument fp

fclose(fp) Ferme le fichier fp





Après les caractères... les chaînes...

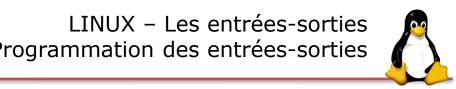












```
#include <stdio.h>
main() {
 FILE *fpIn, *fpOut;
 char chaine[11]; // ajouter un octet de plus pour le marqueur
                  // de fin de chaîne '\0'
 fpIn = fopen("dataIn.txt", "r");
 fgets(chaine, 11, fpIn); // lire 10 caractères dans le fichier
 fclose(fpIn);
 fpOut = fopen("dataOut.txt", "w");
 fputs(chaine, fpOut);
 fputs("\n", fpOut);
 fclose(fpOut);
```









