Université de Reims
Champagne-Ardenne
U.F.R. de Sciences
Exactes et Naturelles

Licence
Informatique
Info 0403
2020/2021

Travaux Dirigés 4 & 5

Fichiers

Exercice 1

1) Écrire un programme permettant de copier un fichier source dans un fichier destination.

```
int main(int argc, char* argv[]){
   int src = open(argv[1],O_RDONLY);
   int dest = open(argv[2],O_CREAT | O_WRONLY,0644);
   int nbcharlu;
   char buf;

   while( (nbcharlu = read(src,&buf,sizeof(char)))!= 0 ){
      write(dest,&buf,nbcharlu);
   }
   close(src);
   close(dest);

   exit(0);
}
```

2) Écrire un programme qui crypte un fichier texte source dans un fichier destination, selon l'algorithme de César (décalage des lettres de x cases, x étant passé en paramètre au programme).

```
int main(int argc, char* argv[]){
   int src = open(argv[1], O_RDONLY);
   int dest = open(argv[2], O_WRONLY | O_CREAT, 0700);
   int dec = atoi(argv[3]);
   char buf;
   int lu;

   while((lu=read(src, &buf,sizeof(char)))!=0){
      buf = buf + dec;
      write(dest, &buf, lu);
   }

   close(src);
   close(dest);

   exit(0);
}
```

3) Écrire un programme permettant de créer un fichier à trous.

```
int main(int argc, char* argv[]){
    int dest = open(argv[1],O_CREAT | O_WRONLY,0644);
    int charlu;

    char buf ='c';

    write(dest,&buf,sizeof(char));
    lseek(dest, 400, SEEK_END);
    write(dest,&buf,sizeof(char));

    close(dest);

    exit(0);
}
```

4) Écrire un programme qui inverse un fichier source dans un fichier destination.

```
int main(int argc, char* argv[]){
    int src = open(argv[1],O_RDONLY);
    int dest = open(argv[2],O_CREAT | O_WRONLY,0644);
    int dep;
    char buf;
    int i=1;

    while( (dep = lseek(src, -i, SEEK_END))!=0){
        read(src,&buf,sizeof(char));
        write(dest,&buf,sizeof(char));
        i++;
    }

    read(src,&buf,sizeof(char));
    write(dest,&buf,sizeof(char));
    close(src);
    close(dest);

    exit(0);
}
```

Exercice 2

1) Écrire un programme qui affiche la taille d'un fichier passé en paramètre.

```
int main(int argc, char* argv[]){
    char *file = argv[1];
    struct stat buf;
    if(stat(file, &buf) != -1){
        printf("Taille du fichier: %ldo\n", buf.st_size);
    } else {
        printf("Erreur\n");
    }

    exit(0);
}
```

2) Écrire un programme qui affiche les droits d'un fichier passé en paramètre.

```
int main(int argc, char* argv[]){
   struct stat buf;

   printf("Mode : %lo (octal)\n",((unsigned long) buf.st_mode ^ S_IFMT));

   if (buf.st_mode & S_IRUSR){
      printf("r");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IWUSR){
      printf("w");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IXUSR){
      printf("x");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IRGRP){
      printf("r");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IWGRP){
      printf("w");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IXGRP){
      printf("x");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IROTH){
      printf("r");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IWOTH){
      printf("w");
   }else{
      printf("-");
   }

   if (buf.st_mode & S_IXOTH){
      printf("x");
   }else{
      printf("-");
   }
   exit(0);
}
```

Exercice 3

1) Écrire un programme qui liste les fichiers que contient le répertoire courant.

```
int main(int argc, char* argv[]){
   DIR * p = opendir("./");
   struct dirent * DE;
   while((DE = readdir(p))!=NULL){
      printf("%s\n",DE->d_name);
   }
   closedir(p);

   exit(0);
}
```

2) Écrire un programme qui affiche le contenu des fichiers contenant l'extension .txt dans le répertoire courant.

```
int main(int argc, char* argv[]){
   DIR * p = opendir("./");
   struct dirent * DE;
   int fic;
   int nbcharlu;
   char buf;

   while((DE = readdir(p))!=NULL){
      if (strstr(DE->d_name, ".txt") !=NULL){
         printf(" %s \n",DE->d_name);
         fic = open(DE->d_name,O_RDONLY);
         while((nbcharlu =read(fic,&buf,sizeof(char)))==1){
            printf ("%c",buf);
         }
         close(fic);
         printf("\n");
      }
   }
   closedir(p);

   exit(0);
}
```

3) Écrire un programme qui liste les fichiers que contient le répertoire courant avec leur taille et l'UID de son propriétaire.

```
int main(int argc, char* argv[]){
   DIR * p = opendir("./");
   struct dirent * DE;
   struct stat buf;
   while((DE = readdir(p))!=NULL)
   {
      stat(DE->d_name, &buf);
      printf("Nom: %s \t Taille: %ld \t User: %d \n", DE->d_name, buf.st_size, buf.st_uid);
   }
   closedir(p);

   exit(0);
}
```

4) Écrire un programme qui liste récursivement le contenu d'un répertoire.

```
void litRep(char *pathname){
   DIR *dirp;
   struct dirent *dp;
   struct stat s;
   chdir(pathname);
   printf("  **** Repertoire %s \n", pathname);
   dirp = opendir(".");
   while( (dp = readdir(dirp)) != NULL)
   {
         printf("%s\n", dp->d_name);
   }
   rewinddir(dirp);
   while( (dp = readdir(dirp)) != NULL)
   {
      stat(dp->d_name, &s);
      if((strcmp(dp->d_name, ".") != 0) && (strcmp(dp->d_name, "..") != 0) && (S_ISDIR(s.st_mode)))
      {
         litRep(dp->d_name);
         chdir("..");
      }
   }
   closedir(dirp);
}

int main(int argc, char *argv[]) {
   litRep(".");
   exit(0);
}
```