CSci 4061 Introduction to Operating Systems

Recitation - 5
Files and Directory Handling

9th Oct 2017 – TA Manu Khandelwal

Agenda

- File and Directory related system calls
  - Does file exists
  - Renaming a file
  - Permission on a file
  - Size of a file
- Creating a symbolic link
- String manipulations in C

Programs we will discuss

- Go to Moodle and under recitation download
  Recitation5-resources.tar.gz
  tar –xvf Recitation5-resources.tar.gz

- programs
  - file_exists.c
  - is_dir.c
  - rename.c
  - symlink.c
  - size.c
  - permission.c
  - string.c
  - string_parsing.

- template
  - file_exists_sol.c
  - is_dir_sol.c
  - rename_sol.c
  - size_sol.c
  - string_sol.c

file_exists.c

Checks if the file exists or not

```
gcc -o file_exists file_exists.c
./file_exists <filename>
```

file_exists.c

```
int main(int argc, char *argv[])
{
     struct stat  st;
     if (argc != 2) {
        printf("Usage: %s <file_name>\n", argv[0]);
        exit(1);
    }
    if (!stat(argv[1], &st)) {
        printf("File Exists!\n");
     } else {
        perror("File does not exist\n");
     }
}
```

Exercise Problem 1

- Problem Statement:
  Write a program to check if a file is executable

- Resources: file_exists_sol.c

Hint: User st_mode property of stat struct and S_IXUSR

is_dir.c

Checks to see if the directory exists

```
gcc -o is_dir is_dir.c
./is_dir <dirname>
```

is_dir.c

```
int main(int argc, char *argv[])
{
if (argc != 2) {
      printf("Usage: %s <file_name>\n", argv[0]);
      exit(1);
}
DIR *dip;      /* points of the directory named filename */
dip = opendir(argv[1]);
if (dip != NULL) {
      printf("Directory exists\n");
} else {
      perror("Directory does not exists\n");
}
}
```

What is wrong with this approach !!

Is the directory there? Or is it permission problem?

Create directory mkdir d1
chmod u-r d1

Run again!

Exercise Problem 2

- Problem Statement:
  Write a program to check if a directory exists

- Resources: is_dir_sol.c

Hint: Use stat system call and explore S_ISDIR() macro

rename.c

Rename a file

```
gcc -o rename rename.c
./rename <current_name> <new_name>
```

rename.c

```
int main(int argc, char *argv[])
{
if (argc != 3) {
	printf("Usage: %s <srcfile> <dstfile>\n", argv[0]);
	return 1;
}
if (rename(argv[1], argv[2]))
	perror("rename");
return 0;
}
```

How to avoid overwriting?

Exercise Problem 3

- Problem Statement:

  Write a program to rename a file.

  If destination file already exist, first rename it to <dest_filename>.bak and then rename source file

- Resources: rename_sol.c

Hint: You will need string function strcat() to create <dest_filename>.bak filename

symlink.c

Create a symbolic link and read the link property

```
gcc -o symlink symlink.c
./ symlink
```

symlink.c

```
int main(int argc, char *argv[])
{
    int status;
    if (argc < 3)    /*  If the user does not enter any directory name*/
    {
        printf("Usage: %s ExistingFilePath NewLinkFilePath \n", argv[0]);
        exit(1);
    }
    status = symlink( argv[1], argv[2] );
    if ( status == -1 ) {
        perror (" Failed to create symbolic link");
        exit(2);
    }
```

size.c

Find the size of a file

```
gcc -o size size.c
./size <filename>
```

size.c

```
int
main(int argc, char *argv[])
{
if (argc != 2) {
        printf("Usage: %s <file_name>\n", argv[0]);
        exit(1);
}
int  size;

/* either do this */
FILE *f = fopen(argv[1], "r");
fseek(f, 0, SEEK_END);
size = ftell(f);
fseek(f, 0, SEEK_SET);
printf("%d\n", size);
```

size.c (Cont)

```
/* or do this */
struct stat st;
stat(argv[1], &st);
size = st.st_size;
printf("%d\n", size);
```

Exercise Problem 4

- Problem Statement:

  Write a program to find sum of all the files (including symbolic link) in an immediate directory. Ignore size of other directories.

- Resources: size_sol.c

permission.c

Get the permissions on a file

```
gcc -o permission permission.c
./ permission <filename>
```

permission.c

```
int main(int argc, char *argv[])
{
        if (argc != 2) {
          printf("Usage: %s <file_name>\n", argv[0]);
          exit(1);
        }
        mode_t perm;

        struct stat st;
        stat(argv[1], &st);
        perm = st.st_mode;
        printf("%o\n", perm);
}
```

string.c

Perform some string manipulations

```
gcc -o string string.c
./string
```

string.c

```
int main()
{
   char example[100];
   char example2[100];

   //string copy and concatenation
   strcpy (example,"Phone ");



   strcat (example,"number ");
   strcat (example,"is ");
   strcat (example,"10 ");
   printf("final string == %s\n\n",example);
```

string.c (Cont)

```
    //string compare
    strcpy (example,"Phone");
    strcpy (example2,"Phone");

    int out = strcmp(example,example2);
    if (out == 0){
        printf("Example and Example 2 are equal\n\n");
    } else {
        printf("Example and Example 2 are different\n\n");
    }

    //string length
    int size = strlen (example);
    printf("string lenght == %d\n\n",size);

    return 0;
}
```

string_parsing.c

Perform some string manipulations

```
gcc -o string_parsing string_parsing.c
./string_parsing
```

string_parsing.c

```
char str[] = "now # is the time for all # good men # aid of their country";
char delims[] = "#";
char *result = NULL;
result = strtok( str, delims );
while( result != NULL ) {
    printf( "result is \"%s\"\n", result );
    result = strtok( NULL, delims );
}
```

Exercise Problem 5

- Problem Statement:

  Write a program to count vowels, consonants, digits and whitespaces in string.

- Resources: string_sol.c