C Programs to simulate UNIX Commands like cp , ls , grep

Aim

To write C programs to simulate UNIX commands like cp ls and grep

1 Program for simulation of cp UNIX Commands

Algorithm

```
1. Start
2. Declare variables : ch , *fp , sc = 0
3. Open the file in read mode
4. Get the character
5. If ch == " " then increment sc value by one
6. Print the number of spaces
7. Close the file
```

Program

```
# include < fcntl.h>
# include <unistd.h>
# include <stdio.h>
main (int argc , char *argv[]) {
        FILE *fp;
        char ch;
```

```
    int sc = 0;

    fp = fopen (argv[1], "r");

    if ( fp == NULL) {

        printf("Unable to open a file", argv[1]);

    }
    else {

        while (! feof( fp)) {

            ch = fgetc ( fp);
            if (ch == '') {

                sc++;
            }
        }

    printf(" no. of spaces = %d\n", sc);
    fclose (fp);

    }

}
```

2) Program for Simulation of LS UNIX Command

Algorithm

1. Start the program
2. Open the directory with directory object dp
3. Read the directory content and print it
4. Close the directory.

Program

```
# include <stdio.h>
# include < dirent.h>
main (int argc, char **argv){
        DIR * dp;
        struct dirent * link;
        dp = opendir (argv[1]);
        printf ("Contents of the directory %s are \n", argv[1]).
        while ((link = readdir (dp))! = 0){
                printf ("%s", link-> d_name);
         }
         closedir (dp);
}
```

3) Program for Simulation of grep UNIX Commands

Algorithm

1. Start the program
2. Declare the variables fline [max], count=0, occurrences = 0 and pointers *fp, * newline
3. Open the file in read mode
4. In while loop check fgets (fline, max, fp)! = NULL
5. Increment count value
6. Check newline = strchr (fline, '\n')
7. print the count, fline value and increment the occurrence value.
8. Stop the program.

Program

```
# include <stdio.h>
# include <string.h>
# define max 1024
void usage() {
        printf(" usage: \t /a.out file name word \n");
}
```

```
int main (int argc, char* argv[]){

    FILE *fp;
    Char fline[max];
    Char * newline;
    int count=0;
    int occurrences =0;
    if (argc != 3) {

        usage();
        exit (1);
    }

    if (!(fp= fopen (argv[1], "r"))) {
        printf(" grep : could not open file : %s \n", argv[1]);
        exit(1);
    }
    while( fgets ( fline, max, fp)! = NULL) {

        Count ++;
        if (newline - strchr( fline, '\n')) {
            *newline = '10';
        }
        if (strstr( fline, argv[2]) ! = NULL) {
            printf ("%s : %d %s \n", argv[1], count, fline);
            occurrences ++;

        }
    }
}
```

2) Program for fork , getpid , exit

Algorithm

1. Start the program
2. Declare the variables pid , pid1 , pid2
3. Call the fork() system call to create process
4. if pid == -1 , exit()
5. if pid ! == -1 get the process id using get pid ()
6. print the process id
7. Stop the program

2) Write a shell program to check the given [illegible] is

or not

<u>Algorithm</u>

1. Start the Program
2. read the value of n
3. Loop from 2 to n/2, i as loop variable
4. if n % 2 == 0 then print "Not prime" and flag = 1
5. if flag != 1 then print "Prime"
6. Stop

3) Write a shell program to find fibonacci series

Algorithm

```
1. Start the Program
2. Declare variables i, a, b, show
3. initilize the variables a=0, b=1 and show=0
4. Read n
5. Use a loop from i=0 to n for the following steps
       - show = a+b
       - a = b
       - b = show
       - print show
       - i++
6  Stop
```