Ex No:3(i) Implementation of Unix system calls: OPENDIR, READDIR.

AIM

To write a program for implementing the following directory system calls of UNIX operating system: opendir, readdir.

ALGORITHM

1.Open the directory specified in command line input.
2. Display the directory contents.

PROGRAM

```
#include<stdio.h>
#include<dirent.h>
#include<stdlib.h>
#include<sys/types.h>
int main(int argc,char *argv[])
{
        DIR *dname;
        struct dirent *dptr;
        printf("\nListening the directory content");
        if(argc!=2){
                printf("Usage :./a.out <dirname>\n");
                exit(-1);
        }

        if((dname=opendir(argv[1]))==NULL){
                perror(argv[1]);
                exit(-1);
        }

        while(dptr=readdir(dname)){
                printf("%s\n",dptr->d_name);
        }


        closedir(dname);

}
```

Ex No:3(ii) Implementation of Unix system calls : STAT.

AIM:

To write the C program to implement the system call stat( ).

ALGORITHM:

Step 1: Include the necessary header files to execute the system call stat( ).
Step 2: Input file path of source and destination file.
Step 3: Open, source file in r (read) and destination file in w (write) mode.
Step 4: Read character from source file and write it to destination file .Repeat step 3 till source file has reached end.
Step 5: Close both source and destination file.

SYSTEM CALLS USED:

1. creat()

The creat() function either creates a new file or prepares to rewrite an existing file named by the path name pointed to by pathname.

2. open( )

The open() system call opens the file specified by pathname. The return value of open() is a file descriptor, a small nonnegative integer that is an index to an entry in the process's table of open file descriptors.

3. fstat( )

fstat() is a system call that is used to determine information about a file based on its file descriptor.

4. stat()

The stat() function obtains information about the file pointed to by path. Read, write, or execute permission of the named file is not required, but all directories listed in the path name leading to the file must be searchable.

PROGRAM CODING:

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<fcntl.h>
main()
{
int fd1,fd2,n;
char source[30],ch[5];
struct stat s,t,w;
fd1=creat("text.txt",0644);
printf("Enter the file to be copied\n");
scanf("%s",source);
fd2=open(source,0);
if(fd2==-1)
{
perror("file doesnot exist");
exit(0);
}
while((n=read(fd2,ch,1))>0)
write(fd1,ch,n);
close(fd2);
stat(source,&s);
printf("Source file size=%d\n",s.st_size);
fstat(fd1,&t);
printf("Destination file size =%d\n",t.st_size);
close(fd1);
}
```

Ex No: 4 SIMULATION OF UNIX COMMANDS LS, CP, GREP

AIM

To write a C program to simulate UNIX commands like ls, cp, grep.

ALGORITHM

1. Create a file using cat command.
2. Simulate ls command to list all the files in current working directory
3. Open the file in read mode and copy the contents of the file.
4. Input the word to be searched in the file.
5. Display the word if searching is successful.

PROGRAM

i)ls

```
#include<stdio.h>
#include<dirent.h>
int main()
{
        struct dirent **namelist;
        int n,i;
        char pathname[100];
        printf("ENTER THE PATH : ");
        scanf("%s",pathname);
        //getcwd(pathname);
        n=scandir(pathname,&namelist,0,alphasort);
        if(n<0){
                printf("No such file or directory");
        }else{
                for(i=0;i<n;i++){
                        printf("%s\n",namelist[i]->d_name);
                }
        }
}
```

ii)grep

PROGRAM:

```
#include <stdio.h>
#include <string.h>
int main()
{
   char fn[30], pat[30], temp[200];
   FILE *fp;
   printf("Enter file name : ");
   scanf("%s", fn);
   printf("Enter pattern to be searched : ");
   scanf("%s", pat);
   fp = fopen(fn, "r");
   if(fp==NULL){
          printf("File not fount");
          return 0;
    }
    while (!feof(fp))
    {
        fgets(temp, 1000, fp);
        if (strstr(temp, pat) != NULL)
            printf("%s", temp);
    }
    fclose(fp);
}
```

iii)cp

PROGRAM:

```
#include <sys/stat.h>
#include <sys/types.h>
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
int main(int argc, char *argv[])
{
    int i, fd1, fd2;
    char *file1, *file2, buf[2];
    file1 = argv[1];
    file2 = argv[2];
    fd1 = open(file1, O_RDONLY, 0777);
    if (fd1 == -1)
    {
        printf("%s File not fount",file1);
        return 0;
    }
    fd2 = creat(file2, 0777);
    while (i = read(fd1, buf, 1) > 0)
        write(fd2, buf, 1);
    printf("Copied Successfully\n");
     close(fd1);
    close(fd2);
}
```