COMSATS UNIVERSITY ISLAMABAD
ATTOCK CAMPUS

Lab Report 5 : Operating System

Submitted to : Sir Fayyaz Ali

Group Members	Muaaz Shoaib FA20-BCS-074 Shahzeb Shaheen FA20-BCS-040

Rubrics Assessment Sheet for Operating System

Lab #:	**Lab no 5**
Lab Title:	**Linux System calls**
Submitted by:	

Names	**Registration**
Muaaz Shoaib	**FA20-BCS-074**
Shahzeb Shaheen	**FA20-BCS-040**

Rubrics name & number		Marks	
		ln-Lab	Post lab
Engineering Knowledge	*R2:* Use of Engineering Knowledge and follow Experiment Procedures: *Ability to follow experimental procedure,control variables,and record Procedural steps on lab report.*		
Problem Analysis	*R6: Experimental Data Analysis :* *Ability to interept findings,compare them to values in the literature,identify weaknesses and limitations*		
Design	*RS: Best Coding Staudards:* *Ability lo follow the* coding standards and programming practices		
Modem Tools Usage	*R9: Undcrsta11d Tools: Ability to describe and explain the principles behind applicability of engineering tools*.		
Individual and Teamwork	*R9:Management of Team Work:* *Ability to appreciate, understand and work multidisciplinary team members*		

Rubrics #	R2	R6	RS	R9	Rl3
Jn -Lab					
Post- Lab					

Description :

System calls :

It is a way in which computer request a service from Kernel of Operating System.

Program coding

Example 1 :

```c
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
void main(int argc,char *arg[])
{
printf("%d",argc);
argv[0]="/bin/ls";
int pid;
pid=fork();
if(pid<0)
{
printf("fork failed");
exit(1);
}
else if(pid==0)
{
 execve( argv[0],argv,NULL);
}
else
{
printf("\n Process id is -%d\n",getpid());
wait(NULL);
exit(0);
} }

```

Example 2 :

```
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
int main( )
{
int pid;
pid=fork( );
if(pid== -1)
{
perror("fork failed");
exit(0);
}
if(pid==0)
{ printf("\n Child process is under execution");
printf("\n Process id of the child process is %d", getpid());
printf("\n Process id of the parent process is %d", getppid());
}
else
{
printf("\n Parent process is under execution");
printf("\n Process id of the parent process is %d", getpid());
printf("\n Process id of the child process in parent is %d", pid());
printf("\n Process id of the parent of parent is %d", getppid());
}
return(0);
}
```

2. PROGRAM USING SYSTEM CALLS opendir( ) readdir( ) closedir()

```
#include <stdio.h>
#include <fcntl.h>
#include <sys/types.h>
#include <unistd.h>

int main(int argc, char **argv)
{
  int i;
  int fd;
  off_t size;

  for (i = 1; i < argc; i++) {
    fd = open(argv[i], O_RDONLY);
    if (fd < 0) {
      printf("Couldn't open %s\n", argv[i]);
    } else {
      size = lseek(fd, (off_t) 0, SEEK_END);
      printf("%10lld %s\n", size, argv[i]);
      close(fd);
    }
  }
  return 0;

}
```

Example 1 :

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/dir.h>
void main(int age,char *argv[])
{
DIR *dir;
struct dirent *rddir;
printf("\n Listing the directory content\n");
dir=opendir(argv[1]);
while((rddir=readdir(dir))!=NULL)
{
printf("%s\t\n",rddir->d_name);
}
closedir(dir);
}

```

3. PROGRAM USING SYSTEM CALL stat( ), creat(),open( ), stat(),fstat( ),gets() and lseek()

Example 1 :

```
#include<stdio.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<unistd.h>
#include<fcntl.h>
void 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,O_RDONLY);
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);
}
```

Example 2 :

```
1 #include<stdio.h>
2 #include<unistd.h>
3 #include<string.h>
4 #include<fcntl.h>
5 int main( )
6 {
7 int fd[2];
8 char buf1[25]= ”just a test\n”;
9 char buf2[50];
10 fd[0]=open(“file1”, O_RDWR);
11 fd[1]=open(“file2”, O_RDWR);
12 write(fd[0], buf1, strlen(buf1));
13 printf(“\n Enter the text now….”);
14 gets(buf1);
15 write(fd[0], buf1, strlen(buf1));
16 lseek(fd[0], SEEK_SET, 0);
17 read(fd[0], buf2, sizeof(buf1));
18 write(fd[1], buf2, sizeof(buf2));
19 close(fd[0]);
20 close(fd[1]);
21 printf(“\n”);
22 return0;
23 }
24
```