**Experiment 2**

Jyothiradithyan K

S4 CSE, 34.

**I. Implementation of fork() system call**

1)

#include <stdio.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

void main(){

int a,b;

a = fork();

if(a==0){

b = fork();

if(b==0){

printf("hello from proc C2\n");

}

else{

wait(NULL);

printf("hello from proc C1\n");

}

}

else{

b = fork();

wait(NULL);

if(b==0){

printf("hello from proc C3\n");

}

else{

wait(NULL);

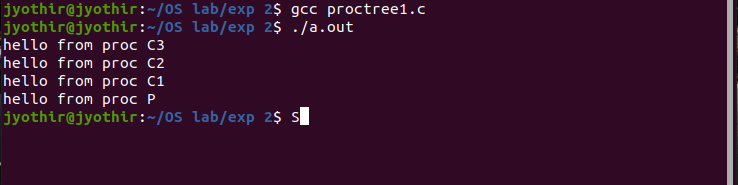
printf("hello from proc P\n");

}

}

}

**Output**



2)

#include <stdio.h>

#include <sys/types.h>

#include <sys/wait.h>

#include <unistd.h>

void main(){

int a,b,c;

a = fork();

b = fork();

c = fork();

if(a>0 && b>0 && c>0){

wait(NULL);

wait(NULL);

wait(NULL);

printf("\nHello from Process P\n");

}

else if(a==0 && b>0 && c>0){

wait(NULL);

wait(NULL);

printf("\nHello from Process C1\n");

}

else if(a==0 && b==0 && c>0){

wait(NULL);

printf("\nHello from Process C2\n");

}

else if(a==0 && b==0 && c==0){

sleep(1);

printf("\nHello from Process C3\n");

}

else if(a==0 && b>0 && c==0){

wait(NULL);

sleep(1);

printf("\nHello from Process C4\n");

}

else if(a>0 && b==0 && c>0){

wait(NULL);

wait(NULL);

printf("\nHello from Process C5\n");

}

else if(a>0 && b==0 && c==0){

wait(NULL);

printf("\nHello from Process C6\n");

}

else if(a>0 && b>0 && c==0){

wait(NULL);

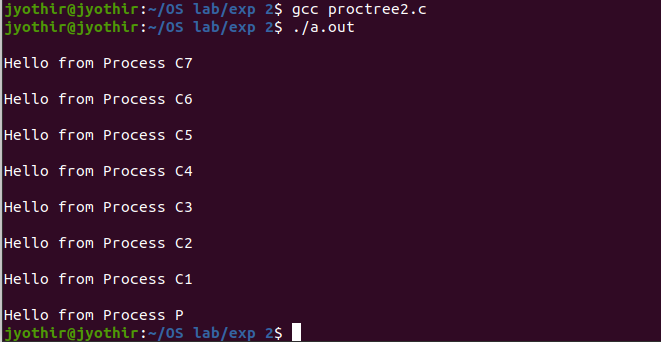
wait(NULL);

printf("\nHello from Process C7\n");

}

}

**Output**

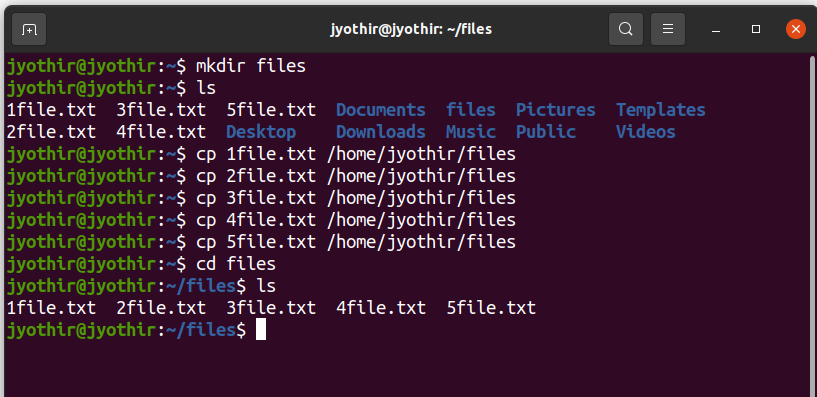


**II. Implementation of Input-Output system calls**

**1)**

**2)**

**Output**

****

**3)**

#include <stdio.h>

#include <dirent.h>

void main(){

struct dirent \*de;

DIR \*dr = opendir(".");

if(dr == NULL){

printf("Could not open current directory" );

}

while((de = readdir(dr))!= NULL){

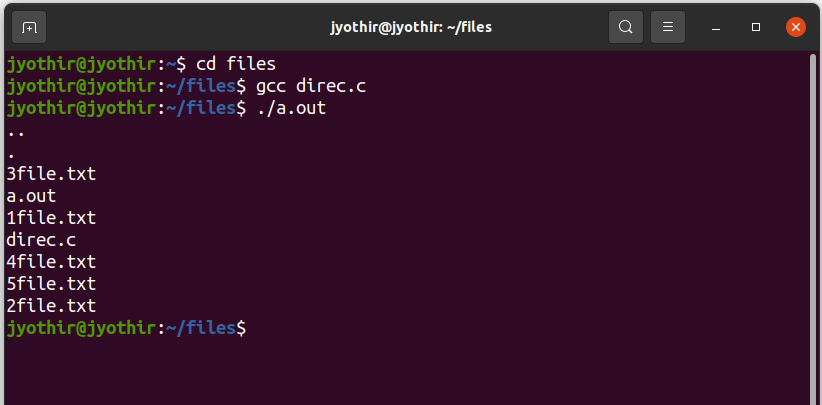
printf("%s\n",de->d\_name);

}

closedir(dr);

}

**Output**



4)

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

#include <string.h>

void main(){

int fd,a,k;

char buffer[80];

char t[10];

for(k = 1;k<6;k++){

sprintf(t,"%d",k);

fd = open(strcat(t,"file.txt"), O\_RDONLY);

if(fd != -1){

a = read(fd,buffer,sizeof(buffer));

}

close(fd);

fd = open("newfile.txt", O\_WRONLY);

lseek(fd,0,SEEK\_END);

if(fd != -1){

printf("merging file %d\n",k);

write(fd,buffer,(a-1));

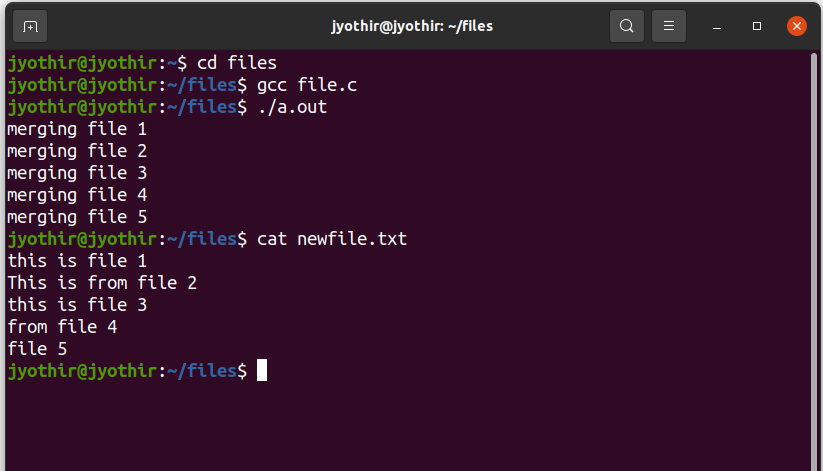
}

close(fd);

}

}

**Output**

****