**1:Write a program to copy a file into another using system calls.**

vi pro1.c

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

void main()

{

char buf;

int fd\_one, fd\_two;

fd\_one = open("first\_file.txt", O\_RDONLY);

if (fd\_one == -1)

{

printf("Error opening first\_file\n");

close(fd\_one);

return;

}

fd\_two = open("second\_file.txt",O\_WRONLY | O\_CREAT,S\_IRUSR | S\_IWUSR | S\_IRGRP | S\_IROTH);

while(read(fd\_one, &buf, 1))

{

write(fd\_two, &buf, 1);

}

printf("Successful copy");

close(fd\_one);

close(fd\_two);

}



**2:Write a c program using system call to create, open , write, close, stat,fstat, lseek**

vi pro2.c

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <fcntl.h>

#include <sys/stat.h>

char buf1[]="LAB ";

char buf2[]="OS Linux";

int main( void)

{

int fd;

if ((fd=creat("file.gol", 0666)) < 0)

{

perror("Creation error");

exit (1);

}

if (write(fd, buf1, sizeof(buf1)) < 0)

{

perror("Writing error");

exit(2);

}

if (lseek(fd, 4096, SEEK\_SET) < 0)

{

perror("Positioning error");

exit(3);

}

if (write(fd, buf2, sizeof(buf2)) < 0)

{

perror("Writing error");

exit(2);

}

}



**3:Write a program to create a child process and allow the parent to display parent and the child to display child on the screen**

vi pro3.c

#include<stdio.h>

#include<unistd.h>

#include<sys/types.h>

int main()

{

pid\_t p;

printf("before fork\n");

p=fork();

if(p==0)

{

printf("I am child having id %d\n",getpid());

printf("My parent's id is %d\n",getppid());

}

else

{

printf("My child's id is %d\n",p);

printf("I am parent having id %d\n",getpid());

}

printf("Common\n");

}

Output:



**4:write a program to create a zombie process**

vi pro4.c

#include<stdlib.h>

#include<stdio.h>

#include <sys/types.h>

#include <unistd.h>

int main()

{

pid\_t child\_pid = fork();

if(child\_pid > 0)

{

sleep(5);

}

else

{

exit(0);

}

return 0;

}

Output:



**5:write a pgm to implement inter process communication using pipes**

**vi pro5.c**

#include<stdio.h>

#include<unistd.h>

int main()

{

int pipefds[2];

int returnstatus;

char writemessages[2][20]={"Hi", "Hello"};

char readmessage[20];

returnstatus = pipe(pipefds);

if (returnstatus == -1)

{

printf("Unable to create pipe\n");

return 1;

}

printf("Writing to pipe - Message 1 is %s\n", writemessages[0]);

write(pipefds[1], writemessages[0], sizeof(writemessages[0]));

read(pipefds[0], readmessage, sizeof(readmessage));

printf("Reading from pipe – Message 1 is %s\n", readmessage);

printf("Writing to pipe - Message 2 is %s\n", writemessages[0]);

write(pipefds[1], writemessages[1], sizeof(writemessages[0]));

read(pipefds[0], readmessage, sizeof(readmessage));

printf("Reading from pipe – Message 2 is %s\n", readmessage);

return 0;

}

**Output:**

