WEEK 8

1.write a c program that illustrates how an orphan is created

Program

```
#include <stdio.h>
#include <unistd.h>
 main() {
int pid;
printf("I'm the original process with PID %d and
PPID %d.\n", getpid(), getppid());
pid = fork();
if (pid != 0)
{
    printf("I'm the parent with PID %d and PPID
%d.\n", getpid(), getppid());
    printf("My child's PID is %d\n", pid);
}
else
{
    sleep(4);
    printf("I'm the child with PID %d and PPID
%d.\n", getpid(), getppid());
}
printf("PID %d terminates.\n", getpid());
return 0; /

}
```

Output

```
I am the original process with PID2242 and PPID1677.
I am the parent with PID2242 and PPID1677
 My child's PID is 2243 PID2243 terminates.
$ I am the child with PID2243 and PPID1.
PID2243 termanates.
```

2.Write a program that illustrates how to execute two commands concurrently with a command pipe.

Program

```
#include <stdio.h> #include <unistd.h> #include <sys/types.h> #include <stdlib.h>
int main() {
int pfds[2];
char buf[80];  // Corrected the size of the buffer
if (pipe(pfds) == -1)
{
   perror("pipe failed");
   exit(1);
}
if (!fork())
{
   close(1);
   dup(pfds[1]);  // Corrected the syntax error
   close(pfds[0]);  // Close unused read end in the child process
   system("ls -l");
}
else
{
   close(pfds[1]);  // Close unused write end in the parent process
   printf("parent reading from pipe \n");
   while (read(pfds[0], buf, sizeof(buf)) > 0)  // Corrected the size argument in read
      printf("%s \n", buf);
}
return 0;  // Added a return statement at the end of the main function

}
```

Output

Parent reading from pipe Total 24

3.Write a C programs that illustrate communication between two unrelated processes using named pipe

Program

```
#include<stdio.h>
 #include<stdlib.h>
#include<errno.h>
#include<unistd.h>
int main() {
int pfds[2];
char buf[30];
if(pipe(pfds)==-1)
{
   perror("pipe");
   exit(1);
}
 printf("writing to file descriptor #%d\n", pfds[1]);
write(pfds[1],"test",5);
printf("reading from file descriptor #%d\n ", pfds[0]);
read(pfds[0],buf,5);
printf("read\"%s\"\n" ,buf);

}
```

Output

```
writing to file descriptor #4 reading from file
descriptor #3 read"test"
```

THANK YOU