Expt. No. ..... Date .....

Page No. 22

SIGNALS

(1) WAP to create a child process. The parent catches a SIGCHILD signal from the child process when the child terminates. (the child sends SIGCHLD either when it is interrupted or it resumes after being interrupted).

Ans)

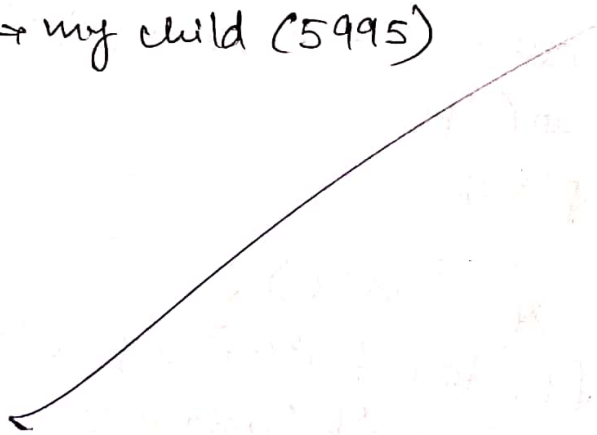
```
#include<stdio.h>
#include<sys/types.h>
#include<sys/wait.h>
#include<signal.h>
  pid_t childpid, pid;
  void func(int signum){
printf("child(%d) exists with SIGCHLD -> %d\n", childpid, signum);
}
   void main(){
   signal(SIGCHLD, func);
    if((childpid=fork())==0){
          pid=getpid();
printf("child(%d) with parent(%d)\n", getpid(), getppid());
    system("who am i");
    exit(0);
    }
    else if(childpid>0){
          sleep(2);
    printf("Parent(%d) -> my child(%d)\n", getpid(), childpid);
    }
}
```

Teacher's Signature : _____

Scanned by CamScanner

OUTPUT

Child (5995) with parent (5994)
2yrgr2 pts/0 2019 09 25 9:21 (:0.0)
child(0) exists with SIGCHLD → 17
child(5995) exists with SIGCHLD → 17
Parent (5994) → my child (5995)



Scanned by CamScanner

Date 16/10/19

Expt. No. DAY 8 Page No. 23

SIGNALS

2,3) WAP to print the default message of SIGINT and also print the user and a program for a process which cannot be killed by pressing ctrl + C and again restore the default status of it

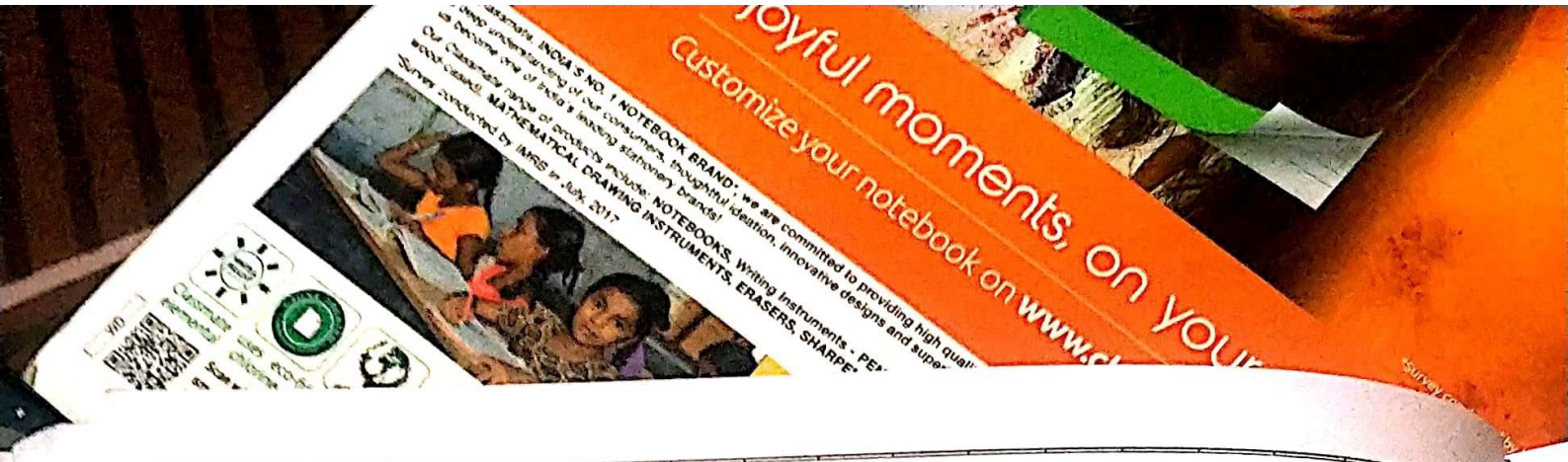
Ans)

```
#include <stdio.h>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <signal.h>
void func (int signum)
{
    signal (SIGINT, SIG_DFL);
    printf ("\nSignal SIGINT received = %d, trying to stop program\n", signum);
}

void main()
{
    signal (SIGINT, func);
    printf ("process (%d) is running\n", getpid());
    sleep(2)
    while (1);
    {}
}
```

Teacher's Signature : _____

Scanned by CamScanner



OUTPUT

```
[2yrgr2] $ gcc signal23.c -o toy
[2yrgr2] $ ./toy
         process (8934) is running
         ^C
         signal SIGINT received=2, trying to stop the program
         ^C
[2yrgr2] $
```

Scanned by CamScanner

File Edit View Search Tools Documents Help

Open Save Undo

signal1.c signal23.c signal4.c signal5.c

```c
#include<unistd.h>
#include<signal.h>
pid_t pida,pidb;
void sigusr1(int signum)
{
    sleep(2);
    printf("\nsignal %d received for SIGUSR1",signum);
    printf("\n Process A(%d) is awake\n",pida);
}
void sigusr2(int signum)
{
    sleep(2);
    printf("\nsignal %d received for SIGUSR2",signum);
    printf("\n Process B(%d) is awake\n",pidb);
}
void func(int signum)
{
    printf("signal sigchild received %d to stop\n",signum);
}
void main()
{
    signal(SIGUSR1,sigusr1);
    signal(SIGUSR2,sigusr2);
    signal(SIGCHLD,func);
    pidb=getpid();
    if((pida=fork())==0)
    {
        printf("child(%d):sending SIGUSR1 to parent(%d)\n",getpid(),pidb);
        kill(pidb,SIGUSR1);
    }
    else if(pida>0)
    {
        printf("parent(%d):sending SIGUSR2 to child(%d)\n",getpid(),pida);
```

2vrqr2 CSE_2_042 signal4.c (~/CSE_2_04... [pipes]

Scanned by CamScanner

File Edit View Search Tools Documents Help

Open Save Undo

signal1.c signal23.c signal4.c signal5.c

```
 9 {
10     sleep(2);
11     printf("\nsignal %d received for SIGUSR1",signum);
12     printf("\n Process A(%d) is awake\n",pida);
13 }
14 void sigusr2(int signum)
15 {
16     sleep(2);
17     printf("\nsignal %d received for SIGUSR2",signum);
18     printf("\n Process B(%d) is awake\n",pidb);
19 }
20 void func(int signum)
21 {
22     printf("signal sigchild received %d to stop\n",signum);
23 }
24 void main()
25 {
26     signal(SIGUSR1,sigusr1);
27     signal(SIGUSR2,sigusr2);
28     signal(SIGCHLD,func);
29     pidb=getpid();
30     if((pida=fork())==0)
31     {
32         printf("child(%d):sending SIGUSR1 to parent(%d)\n",getpid(),pidb);
33         kill(pidb,SIGUSR1);
34     }
35     else if(pida>0)
36     {
37         printf("parent(%d):sending SIGUSR2 to child(%d)\n",getpid(),pida);
38         kill(pida,SIGUSR2);
39         wait(NULL);
40     }
41 }
```

2vrar2 CSE_2_042 signal4.c (~/CSE_2_04... [pipes]

Scanned by CamScanner

E_2_042) - gedit

2yrgr2@localhost:~/CSE_2_042

File Edit View Search Terminal Help

```
signal4.c:28: error: (Each undeclared identifier is reported o
signal4.c:28: error: for each function it appears in.)
[2yrgr2@localhost CSE_2_042]$ gcc signal4.c -o try
[2yrgr2@localhost CSE_2_042]$ ./try
parent(4105):sending SIGUSR2 to child(4106)

signal 12 received for SIGUSR2
 Process B(4105) is awake
child(4106):sending SIGUSR1 to parent(4105)
signal sigchild received 17 to stop

signal 10 received for SIGUSR1
 Process A(4106) is awake
[2yrgr2@localhost CSE_2_042]$ signal5.c
bash: signal5.c: command not found
[2yrgr2@localhost CSE_2_042]$ gedit signal5.c
[2yrgr2@localhost CSE_2_042]$ clear

[2yrgr2@localhost CSE_2_042]$ gcc signal5.c -o try
signal5.c: In function 'call1':
signal5.c:14: error: 'delay' undeclared (first use in this functio
signal5.c:14: error: (Each undeclared identifier is reported only
signal5.c:14: error: for each function it appears in.)
signal5.c: In function 'main':
```

Scanned by CamScanner

Applications Places System

signal5.c (~

File Edit View Search Tools Documents Help

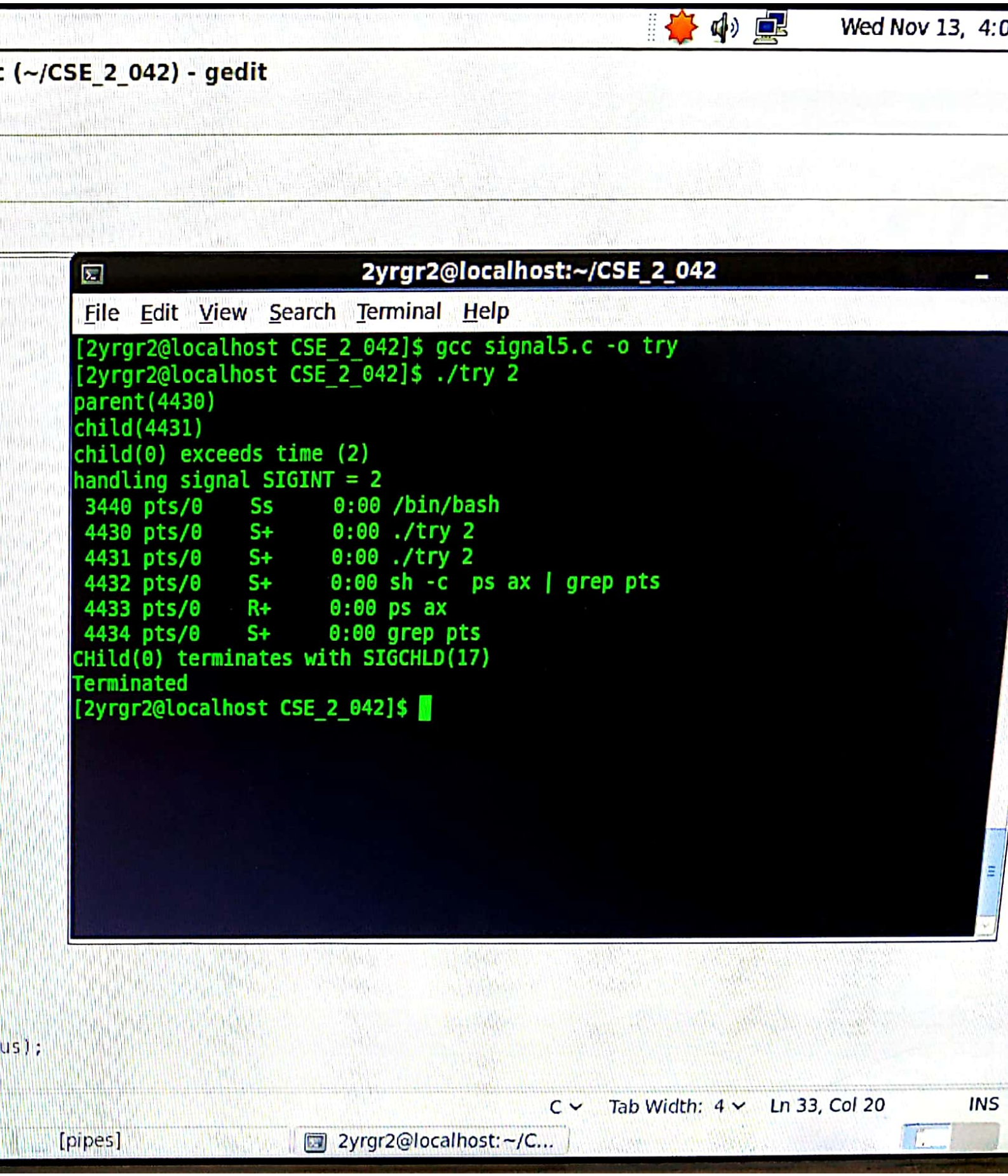
Open Save Undo

signal1.c | signal23.c | signal4.c | signal5.c

```
#include<unistd.h>
#include<signal.h>
pid_t pid;int delay;
void call(int signum)
{
    printf("CHild(%d) terminates with SIGCHLD(%d)\n",pid,signum);
}
void call1(int signum)
{
    printf("child(%d) exceeds time (%d)\n",pid,delay);
    printf("handling signal SIGINT = %d\n",signum);
    system(" ps ax | grep pts");
    kill(pid,SIGTERM);
}
int main(int argc,char *argv[])
{
    int childpid,status;
    signal(SIGINT,call1);
    signal(SIGCHLD,call);
    delay=atoi(argv[1]);
    if((pid=fork())==0)
    {
        printf("child(%d)\n",getpid());
        sleep(2);
    }
    else if(pid>0)
    {
        printf("parent(%d)\n",getpid());
        sleep delay);
        kill(pid,SIGINT);
        childpid=wait(&status);
        printf("wait() returned childpid(%d),status(%d)\n",childpid,status);
    }
```

2yrgr2 | CSE_2_042 | signal5.c (~/CSE_2_04...

Scanned by CamScanner



Scanned by CamScanner