Name - Moitrish Maity
Roll - 36
CSE 3rd year 5th sem

Assignment on Signals in UNIX

1)

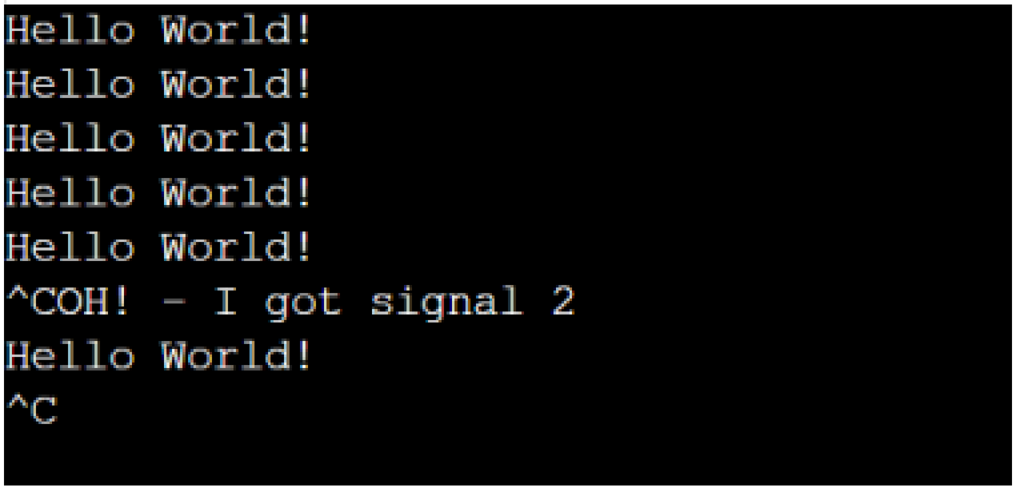
```
//Use of sigint (^C)

#include <signal.h>

#include <stdio.h>

#include <unistd.h>

void oh(int sig)

{

printf("OH! - I got signal %d\n", sig);

signal(SIGINT,oh);/* THIS LINE WILL CONTINUE THE EXECUTION OF FUNCTION OH

*/ //signal(SIGINT,SIG_DFL);

}

int main()

{

signal(SIGINT, oh);

while(1)

{

printf("Hello World!\n");

sleep(1);

}

}
```

Output-
Infinite loop

2)

```
//Use of SIG_DFL (reset to default)

#include <signal.h>

#include <stdio.h>

#include <unistd.h>

void oh(int sig)

{

printf("OH! - I got signal %d\n", sig);

signal(SIGINT,SIG_DFL);/*resets the signal to default*/

}

int main()

{

signal(SIGINT, oh);

while(1)

{

printf("Hello World!\n");

sleep(1);

}

}
```

Output-



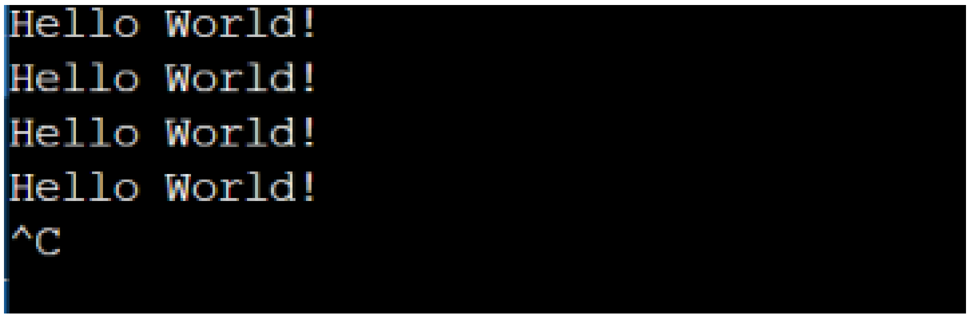
```
Hello World!
Hello World!
Hello World!
Hello World!
Hello World!
^COH! - I got signal 2
Hello World!
^C
```

3)

//Use of SIGQUIT (^C to quit process)

```
#include <signal.h>

#include <stdio.h>

#include <unistd.h>

void oh(int sig)

{

printf("OH! - I got signal %d\n", sig);

signal(SIGINT,oh);/* THIS LINE WILL CONTINUE THE EXECUTION OF FUNCTION OH

*/ //signal(SIGQUIT,SIG_DFL);

}

int main()

{

signal(SIGQUIT, oh);
while(1)

{

printf("Hello World!\n");

sleep(1);

}

}
```

Output



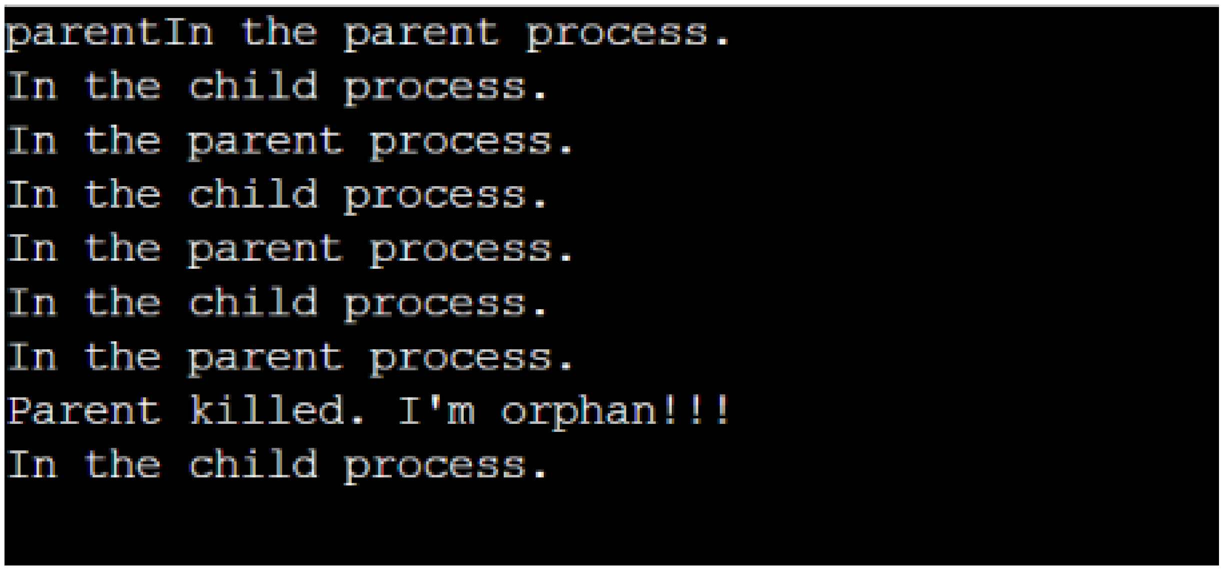
```
Hello World!
Hello World!
Hello World!
Hello World!
^C
```

4)

```
//Use of SIGKILL (kill parent from child)

#include <stdio.h>
```

```
#include <stdlib.h>

#include <unistd.h>

#include <signal.h>

int main(void)

{

pid_t ppid,pid,cpid;

ppid=getpid();

pid = fork();

if(ppid==getpid())
printf("parent");

else if (cpid==getpid())

printf("child");

if(pid > 0)

{

int i = 0;

while(i++ < 5)

{

printf("In the parent process.\n");

sleep(1);

}

}

else if (pid == 0)

{

int i = 0;

while(i++ < 10)

{

printf("In the child process.\n");
sleep(1);

if(i==3)

{

kill(ppid,SIGKILL); /* SIGKILL Kills the process ( it cannot be caught or

ignored)*/ printf("Parent killed. I'm orphan!!!\n");

}

}
}

else

{

//something bad happened.

printf("Something bad happened.");

exit(EXIT_FAILURE);

}

return 0;

}
```

Output

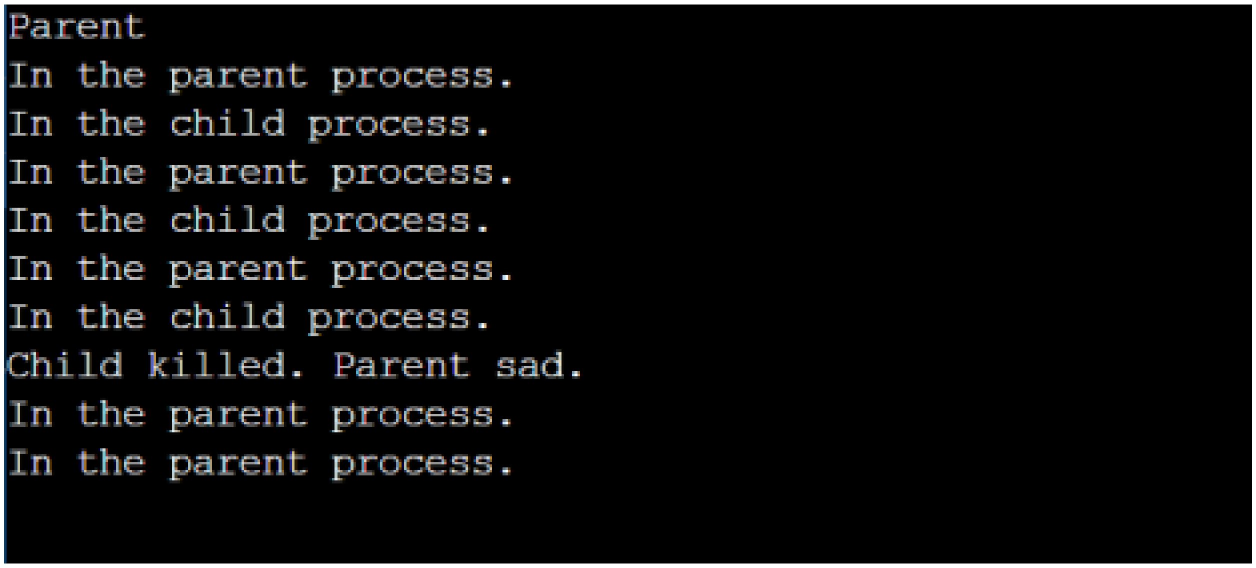


```
parentIn the parent process.
In the child process.
In the parent process.
In the child process.
In the parent process.
In the child process.
In the parent process.
Parent killed. I'm orphan!!!
In the child process.
```

5) Write a program to kill the child process from the parent process using SIGNAL.

```
//Use of SIGKILL (kill parent from child)

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>
#include <signal.h>

int main(void)

{

    pid_t ppid, pid, cpid;

    ppid = getpid();

    pid = fork();

     if(ppid == getpid())

    printf("Parent\n");

    else if (cpid == getpid())

    printf("Child\n");

    if(pid > 0)

    {

        int i = 0;

        while(i++ < 5)

        {

            printf("In the parent process.\n");

            sleep(1);

            if(i == 3)

            {

                kill(pid, SIGKILL); /* SIGKILL Kills the process (it cannot be caught or ignored)*/

                printf("Child killed. Parent sad.\n");

            }

        }

    }
    else if (pid == 0)
    {

    int i = 0;

    while(i++ < 5)

    {

        printf("In the child process.\n");

        sleep(1);

    }

    }

    else

    {

        //something bad happened.

        printf("Something bad happened.");

        exit(EXIT_FAILURE);

    }

    return 0;

}
```

Output-



```
Parent
In the parent process.
In the child process.
In the parent process.
In the child process.
In the parent process.
In the child process.
Child killed. Parent sad.
In the parent process.
In the parent process.
```