RIPHAH INTERNATIONAL UNIVERSITY

Name: Deher Zainab

Sap ID: 49710

BSCS-5th Semester

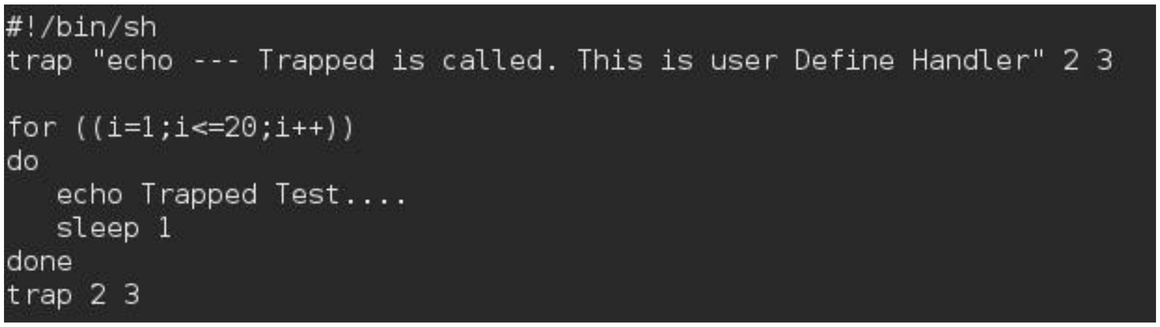
OS Lab Tasks

LAB # 12

Submitted to:

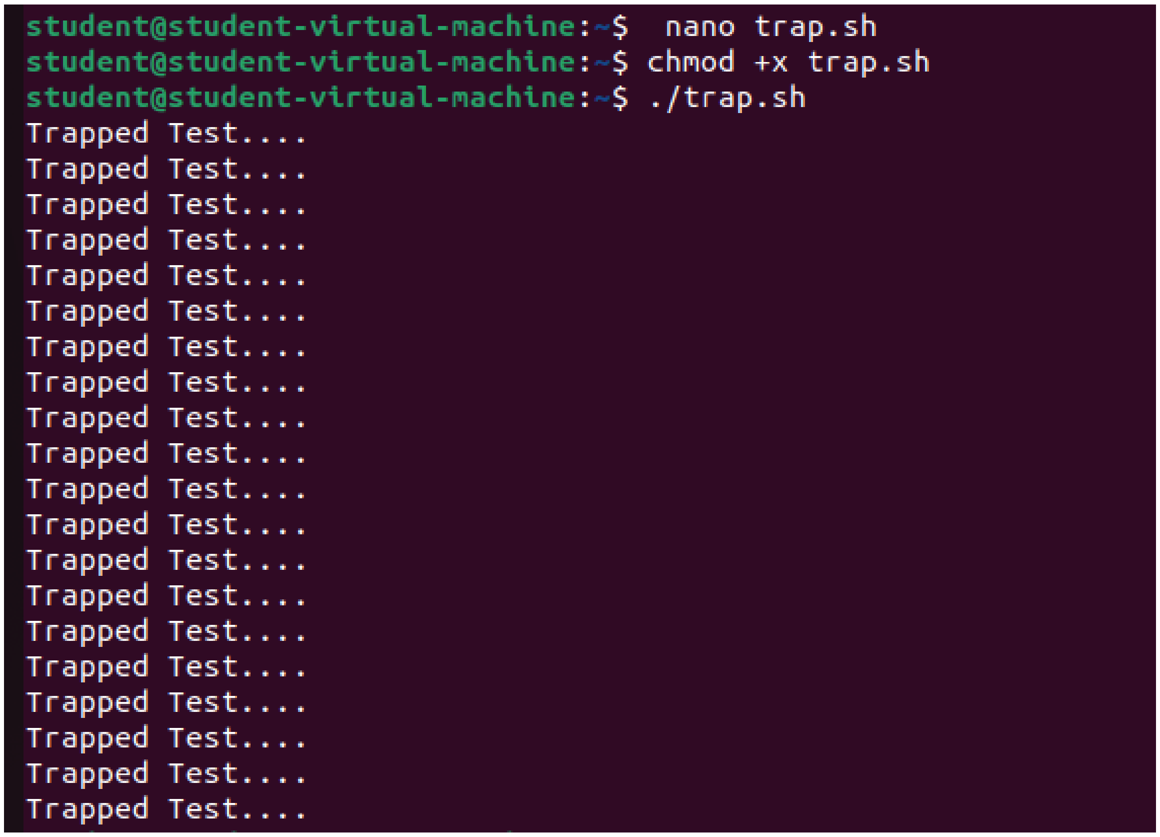
Mam Ayesha

Task 1:
Execute following program:



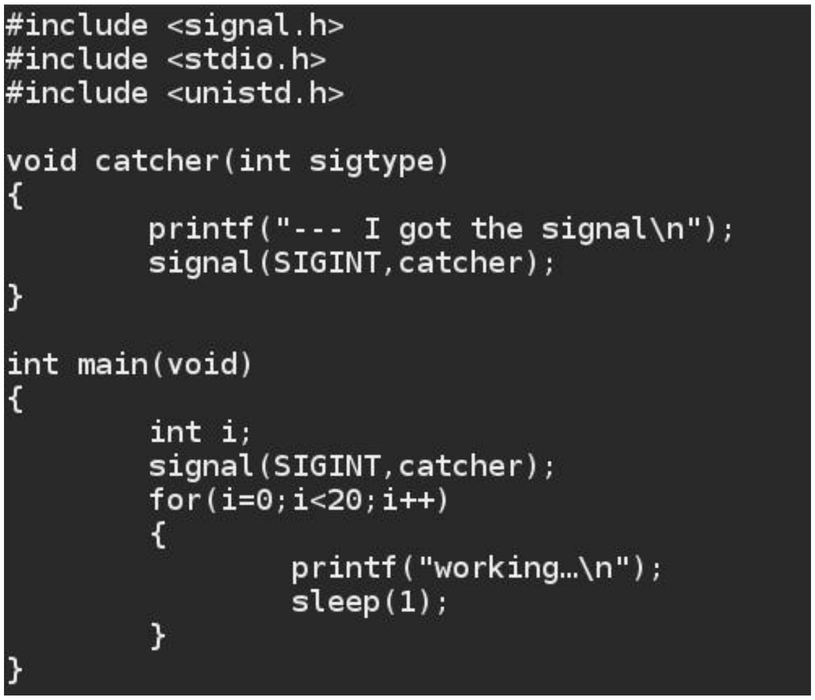
```
#!/bin/sh
trap "echo --- Trapped is called. This is user Define Handler" 2 3

for ((i=1;i<=20;i++))
do
   echo Trapped Test....
   sleep 1
done
trap 2 3
```

Solution:



```
student@student-virtual-machine:~$  nano trap.sh
student@student-virtual-machine:~$ chmod +x trap.sh
student@student-virtual-machine:~$ ./trap.sh
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
Trapped Test....
```

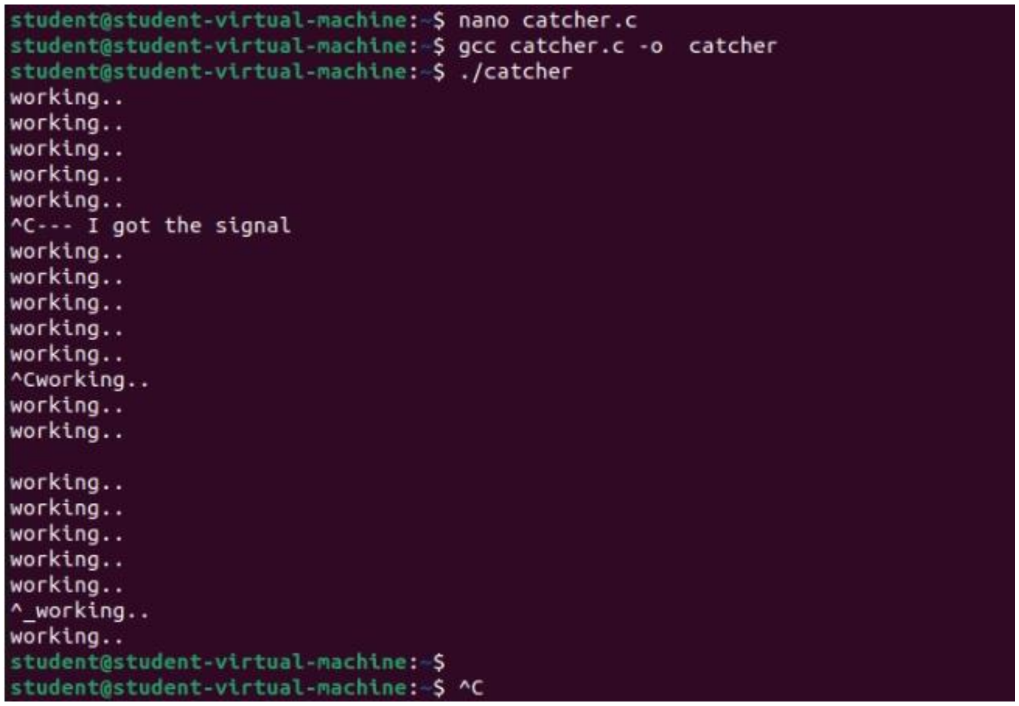
Task 2:
Execute it.



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

void catcher(int sigtype)
{
        printf("--- I got the signal\n");
        signal(SIGINT,catcher);
}

int main(void)
{
        int i;
        signal(SIGINT,catcher);
        for(i=0;i<20;i++)
        {
                printf("working…\n");
                sleep(1);
        }
}
```

Press <Ctrl-C> while the program is running and see what happens.

Solution:



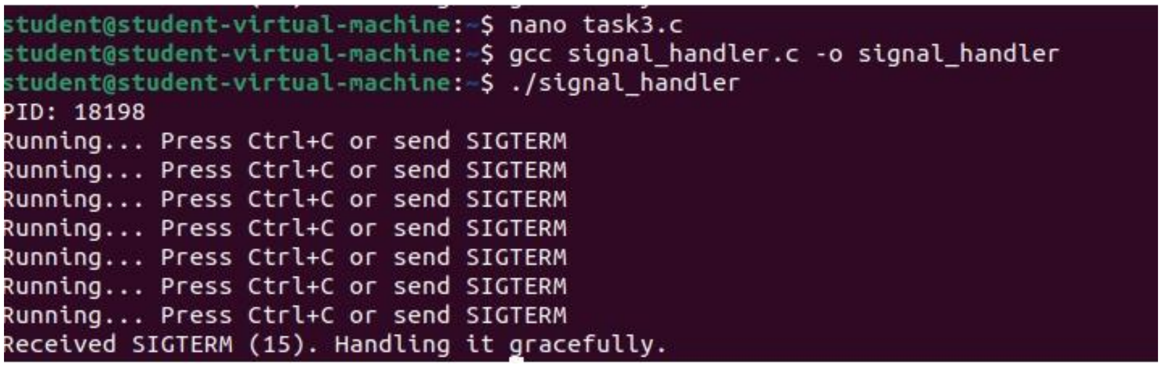
```
student@student-virtual-machine:~$ nano catcher.c
student@student-virtual-machine:~$ gcc catcher.c -o  catcher
student@student-virtual-machine:~$ ./catcher
working..
working..
working..
working..
working..
^C--- I got the signal
working..
working..
working..
working..
working..
^Cworking..
working..
working..

working..
working..
working..
working..
working..
^_working..
working..
student@student-virtual-machine:~$
student@student-virtual-machine:~$ ^C
```

Task 3:
Write a c programs that performs the following functions

a. **signal handler for ignoring the signal**
b. **signal handler for default action**
c. **signal handler for kill command**

Your programs should be well commented.

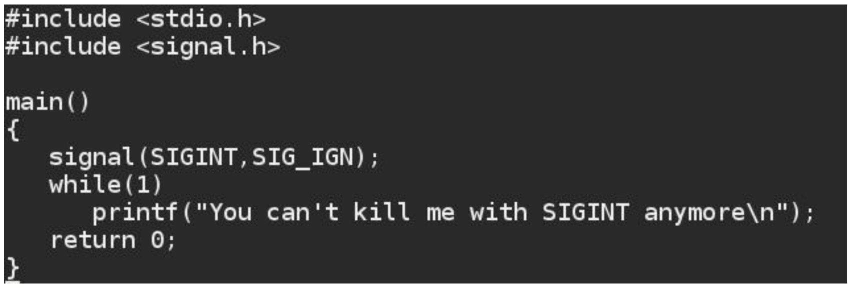
Solution:



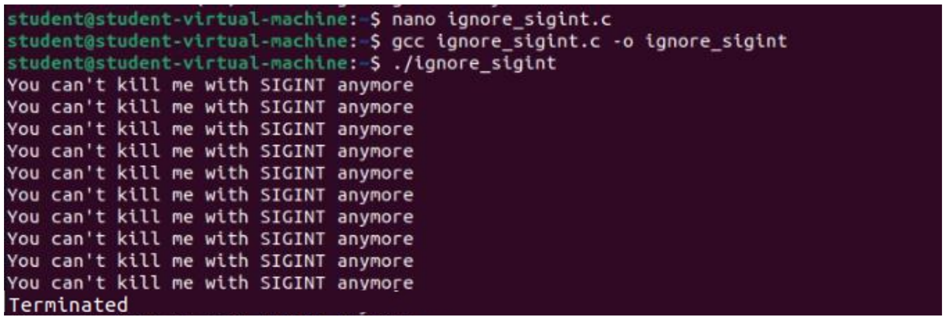
```
student@student-virtual-machine:~$ nano task3.c
student@student-virtual-machine:~$ gcc signal_handler.c -o signal_handler
student@student-virtual-machine:~$ ./signal_handler
PID: 18198
Running... Press Ctrl+C or send SIGTERM
Running... Press Ctrl+C or send SIGTERM
Running... Press Ctrl+C or send SIGTERM
Running... Press Ctrl+C or send SIGTERM
Running... Press Ctrl+C or send SIGTERM
Running... Press Ctrl+C or send SIGTERM
Running... Press Ctrl+C or send SIGTERM
Received SIGTERM (15). Handling it gracefully.
```

Task 4:
Execute following code.

Solution:



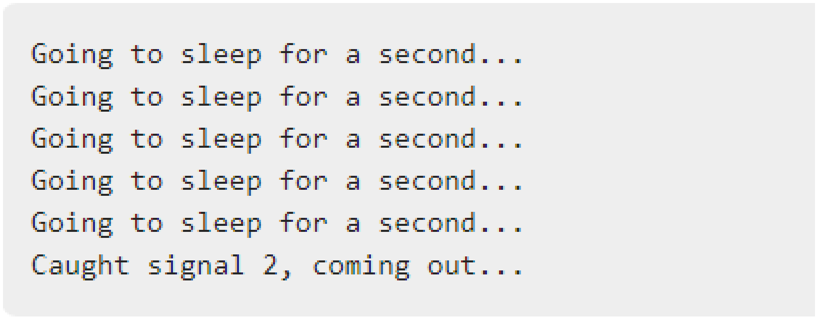
```
#include <stdio.h>
#include <signal.h>

main()
{
   signal(SIGINT,SIG_IGN);
   while(1)
      printf("You can't kill me with SIGINT anymore\n");
   return 0;
}
```



```
student@student-virtual-machine:~$ nano ignore_sigint.c
student@student-virtual-machine:~$ gcc ignore_sigint.c -o ignore_sigint
student@student-virtual-machine:~$ ./ignore_sigint
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
You can't kill me with SIGINT anymore
Terminated
```

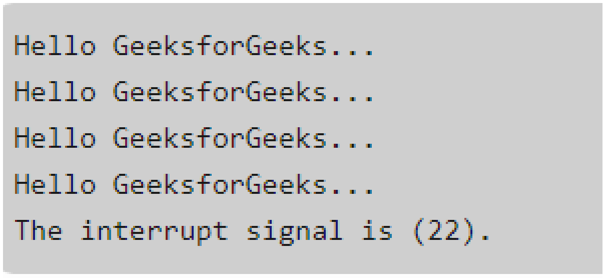
Task 5:

Write C programs that shows following output
a.



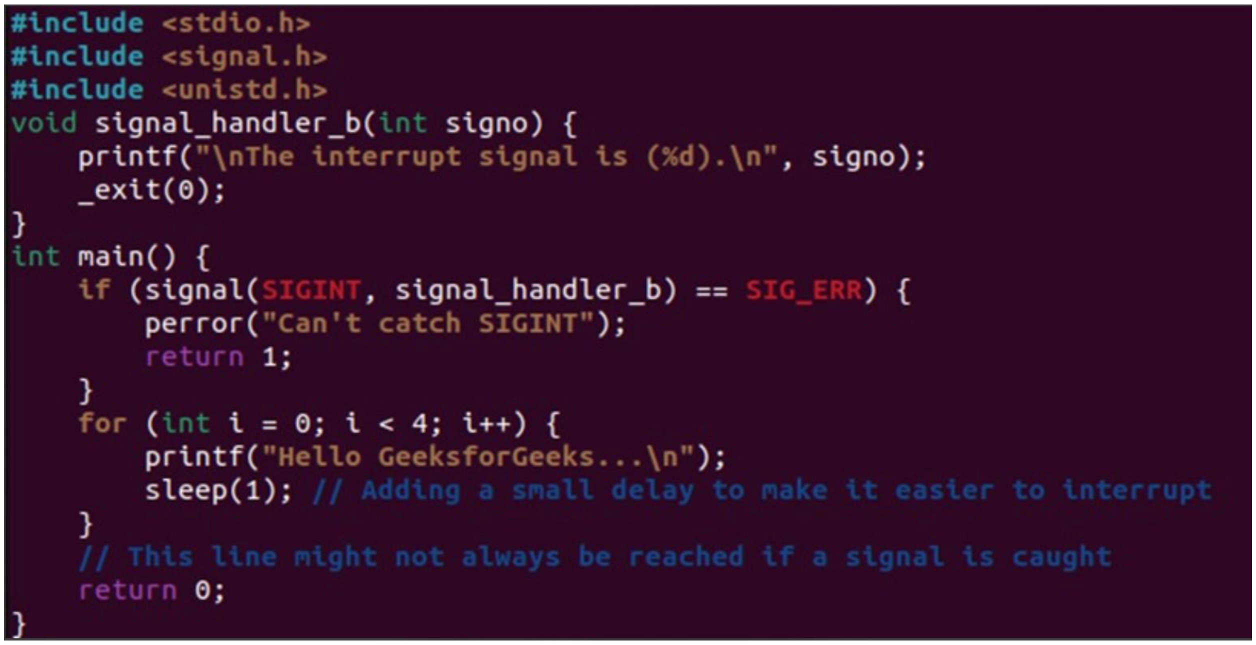
```
Going to sleep for a second...
Going to sleep for a second...
Going to sleep for a second...
Going to sleep for a second...
Going to sleep for a second...
Caught signal 2, coming out...
```

b.

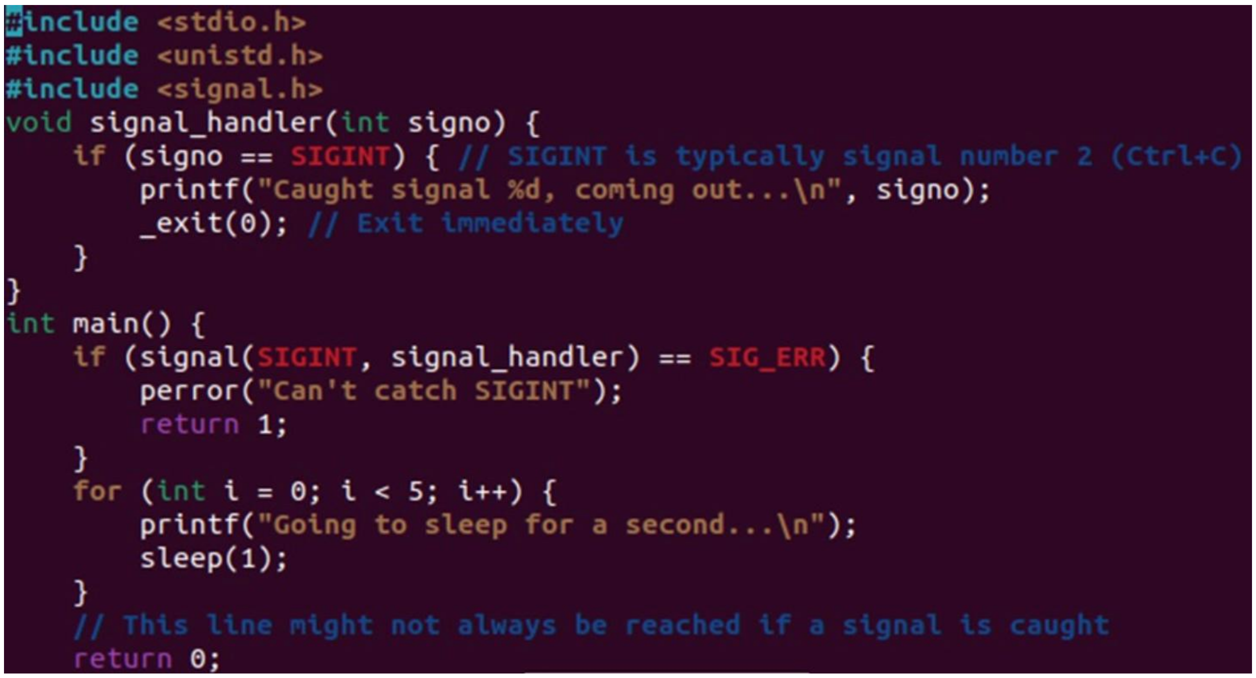


```
Hello GeeksforGeeks...
Hello GeeksforGeeks...
Hello GeeksforGeeks...
Hello GeeksforGeeks...
The interrupt signal is (22).
```

Solution:



```c
#include <stdio.h>
#include <signal.h>
#include <unistd.h>
void signal_handler_b(int signo) {
    printf("\nThe interrupt signal is (%d).\n", signo);
    _exit(0);
}
int main() {
    if (signal(SIGINT, signal_handler_b) == SIG_ERR) {
        perror("Can't catch SIGINT");
        return 1;
    }
    for (int i = 0; i < 4; i++) {
        printf("Hello GeeksforGeeks...\n");
        sleep(1); // Adding a small delay to make it easier to interrupt
    }
    // This line might not always be reached if a signal is caught
    return 0;
}
```



```
#include <stdio.h>
#include <unistd.h>
#include <signal.h>
void signal_handler(int signo) {
    if (signo == SIGINT) { // SIGINT is typically signal number 2 (Ctrl+C)
        printf("Caught signal %d, coming out...\n", signo);
        _exit(0); // Exit immediately
    }
}
int main() {
    if (signal(SIGINT, signal_handler) == SIG_ERR) {
        perror("Can't catch SIGINT");
        return 1;
    }
    for (int i = 0; i < 5; i++) {
        printf("Going to sleep for a second...\n");
        sleep(1);
    }
    // This line might not always be reached if a signal is caught
    return 0;
```