Day – 4 LSP Assignments

1. File Management:

a. Write a script that takes a directory path as input and creates a new directory within it named "Backups_$(date +%Y-%m-%d)".

```
rps@rps-virtual-machine:~$ vim b1.sh
rps@rps-virtual-machine:~$ bash b1.sh
Enter the directory path: dir1
Directory created: dir1/Backups_2024-07-22.
rps@rps-virtual-machine:~$ cat b1.sh
#!/bin/src/env bash


read -p "Enter the directory path: " dir_path

if [ -d "$dir_path" ]; then
        backup_dir="$dir_path/Backups_$(date +%Y-%m-%d)"
        mkdir -p "$backup_dir"
        echo "Directory created: $backup_dir."
else
        echo "Directory does not exist."
fi
rps@rps-virtual-machine:~$
```

b. Create a script that renames all files in a directory with the extension ".txt" to have a prefix of "report_".

```
rps@rps-virtual-machine:~$ vim b1a.sh
rps@rps-virtual-machine:~$ bash b1a.sh
Enter the directory name: dir2
b1a.sh: line 6: syntax error near unexpected token `if'
b1a.sh: line 6: `        for file in "$dir_path"/*.txt; if [ -e "$file" ]; then'
rps@rps-virtual-machine:~$ cat b1a.sh
#!/bin/src/env bash

read -p "Enter the directory name: " dir_path

if [ -d "$dir_path" ]; then
        for file in "$dir_path"/*.txt; if [ -e "$file" ]; then
                mv "$file" "dir_path/report_$basename "$file")"
        done
        echo "All .txt files have been renamed with prefix 'report_'."
else
        echo " Directory does not exist."
        fi

rps@rps-virtual-machine:~$
```

2. User Interaction:

a. Write a script that prompts the user for their name and age, then greets them with a personalized message.

```
-rw-rw-r-- 1 rps rps     0 Jul 20 16:11  man
-rw-rw-r-- 1 rps rps   153 Jul 22 09:14  b1.sh
rps@rps-virtual-machine:~$ vim q2.sh
rps@rps-virtual-machine:~$ bash q2.sh
Enter your name: Jimmy George
Enter your age: 22
Hello, Jimmy George! you are 22 years old.
rps@rps-virtual-machine:~$ cat q2.sh
#!/bin/bash

read -p "Enter your name: " name
read -p "Enter your age: " age

echo "Hello, $name! you are $age years old."
rps@rps-virtual-machine:~$
```

b. Design a script that displays a menu with options like "List files," "Create directory," and "Exit." Allow the user to choose an option and perform the corresponding action.

```
rps@rps-virtual-machine:~$ cat q2a.sh
#!/bin/bash

while true; do
        echo " Menu:"
        echo "1. List files "
        echo "2. Create directory \n"
        echo "3. Exit."
        read -p " Choose one option: " option

case $option in
        1)
        ls
        ;;
2)
read -p "Enter directory name: "dir_name
mkdir -p "$dir_name"
echo "Directory created: $dir_name"
;;
3)
        echo "Exiting"
        break
        ;;
*)
        echo "Invalid option"
        ;;
  esac
done
```

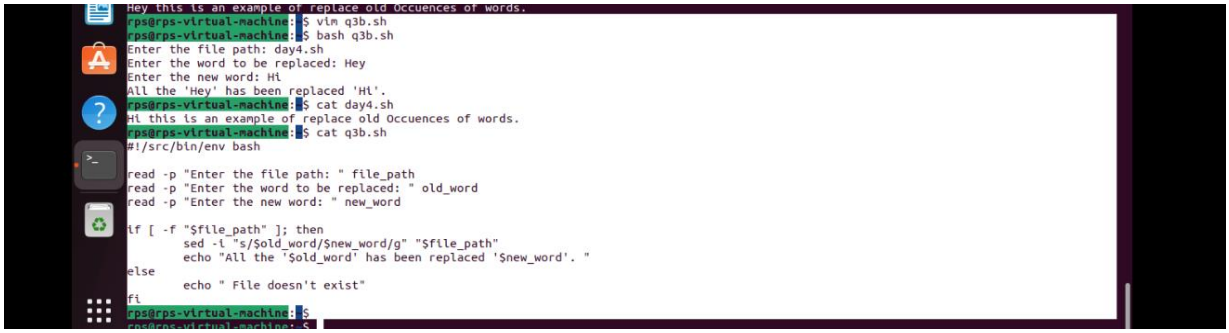
```
rps@rps-virtual-machine:~$ vim q2a.sh
rps@rps-virtual-machine:~$ bash q2a.sh
 Menu:
1. List files
2. Create directory \n
3. Exit.
 Choose one option: 1
' '                 date_alarm.sh   error.txt            file_manager.sh   linuxtoolbox   p1.sh      q4a.sh             t3.sh       test4.sh
 b1.sh              day4.sh         ex2                  g1.sh             listall.sh     Pictures   q4b.sh             t4.sh       test5.cpp
 backup             Desktop         Exam1                github            m1.log         project    q5a.sh             t5.sh       test.sh
 backup.tar.gz      dir1            example1.txt         hello             man            Projects   q6a.sh             t6.sh       Videos
 cal.sh             dir2            exercises            hello1            Music          Public     sales_backup.sql   t7.sh
 d1.sh              directory2      failure_success.sh   hello1.cpp        myddir         q2a.sh     server.sh          t8.sh
 d2.sh              dir_p1          file_exist1.sh       hello.cpp         my_dir         q2.sh      snap               t9.sh
 data               Documents       file_exist2.sh       important_files   myfile.txt     q3a.sh     t1.sh              Templates
 data1.bak          Downloads       file_exist.sh        invite            my_scripts     q3b.sh     t2.sh              test3.sh
 Menu:
1. List files
2. Create directory \n
3. Exit.
 Choose one option: 3
Exiting
```

3. Text Processing:

a. Write a script that reads the contents of a file line by line, counts the number of lines, and prints the total.

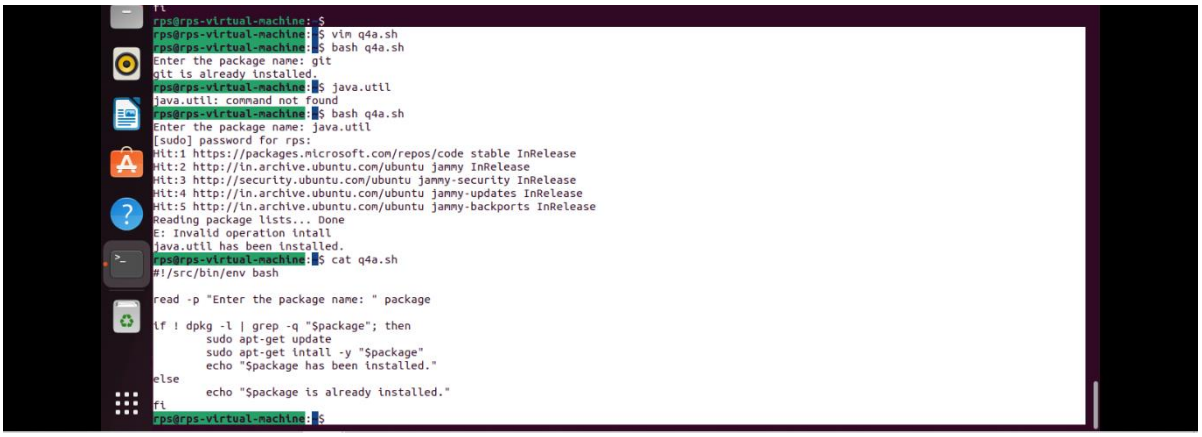
```
 Choose one option: ^C
rps@rps-virtual-machine:~$ vim q3a.sh
rps@rps-virtual-machine:~$ bash q3a.sh
Enter the file: file_path^C
rps@rps-virtual-machine:~$ vim q3a.sh
rps@rps-virtual-machine:~$ bash q3a.sh
Enter the file: failure_success.sh
Total number of lines:
rps@rps-virtual-machine:~$ cat failure_success.sh
#!/usr/bin/env bash

result=$(($RANDOM % 2))
if [ $result -eq 0 ]
then
        true
        echo "$?"
else
        false
        echo "$?"
fi
rps@rps-virtual-machine:~$ cat q3a.sh
#!/bin/bash

read -p "Enter the file: " file_path

if [ -f "$file_path" ]; then
        ine_count=$(wc -l < "$file_path")
        echo "Total number of lines: $line_count"
else
        echo "file doesn't exist."
fi
rps@rps-virtual-machine:~$
```

b. Create a script that takes a text file as input and replaces all occurrences of a specific word with another word.



4. System Administration:

a. Write a script that checks if a specific package is installed and, if not, install it using the appropriate package manager (e.g., apt-get, yum).



b. Create a script that monitors disk usage and sends an email notification if it exceeds a certain threshold.

5. Data Manipulation:

a. Write a script that reads a CSV file, calculates the average of a specific column, and prints the result.

```
echo "Backup completed."
rps@rps-virtual-machine:~$ vim 5a.sh
rps@rps-virtual-machine:~$ cat 5a.sh
#!/bin/src/env bash

read -p "Enter the CSV file path: "csv_file
read -p "Enter the column number to calculate average: " column

if [ -f "$csv_file" ]; then
        average=$(awk -F',' -v col="$column" '{sum += $col} END {print sum/NR}' "$csv_file")
        echo "The average of coloumn $column is: $average"
else
        echo "FIle doesn't exist."
fi
rps@rps-virtual-machine:~$
```

b. Create a script that generates a random password of a specified length, meeting certain criteria like uppercase, lowercase, numbers, and symbols.

```
rps@rps-virtual-machine:~$ vim q5a.sh
rps@rps-virtual-machine:~$ bash q5a.sh
Enter the desired password length: Jimmy@#123
head: invalid number of bytes: 'Jimmy@#123'
Generated password:
rps@rps-virtual-machine:~$ bash q5a.sh
Enter the desired password length: rps@2345
head: invalid number of bytes: 'rps@2345'
Generated password:
rps@rps-virtual-machine:~$ cat q5a.sh
#!/bin/bash

read -p "Enter the desired password length: " length

password=$(tr -dc 'A-Za-z0-9!@#$%^&*()_+' </dev/urandom | head -c "$length")
echo "Generated password: $password"
rps@rps-virtual-machine:~$
```

6. Network Operations:

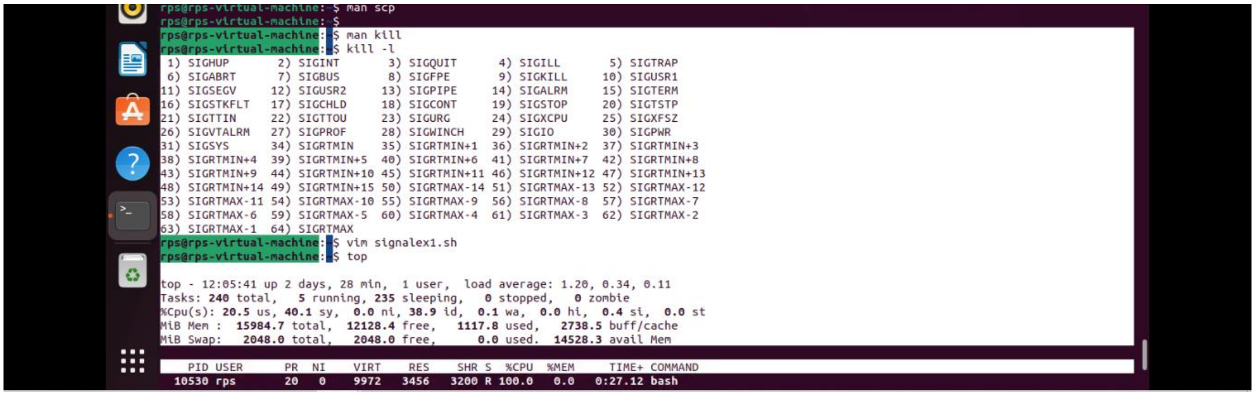
a. Write a script that pings a list of servers and reports if any are unreachable.

```
rps@rps-virtual-machine:~$ vim q6a.sh
rps@rps-virtual-machine:~$ bash q6a.sh
Enter the file path containing the lit of servers: google.com
q6a.sh: line 5: [: missing `]'
File doesn't exist.
rps@rps-virtual-machine:~$ touch server.sh
rps@rps-virtual-machine:~$ cat server.sh
rps@rps-virtual-machine:~$ cat > server.sh
www.google.com^C
rps@rps-virtual-machine:~$ cat > server.sh
www.google.com
rps@rps-virtual-machine:~$ bash q6a.sh
Enter the file path containing the lit of servers:
q6a.sh: line 15: : No such file or directory
rps@rps-virtual-machine:~$ bash q6a.sh
Enter the file path containing the lit of servers: server.sh
q6a.sh: line 5: [: missing `]'
File doesn't exist.
rps@rps-virtual-machine:~$ cat q6a.sh
#!/bin/src/env bash

read -p "Enter the file path containing the lit of servers: " server_list

if       [ -f "$server_list"]; then
        while IFS= read -r server; do
                if ping -c 1 "$server" &> /dev/null; then
                        echo "$server is reachable."
                else
                echo "$server is unreachable."
                fi
        done <"$server_list"
else
echo "File doesn't exist."
fi
```

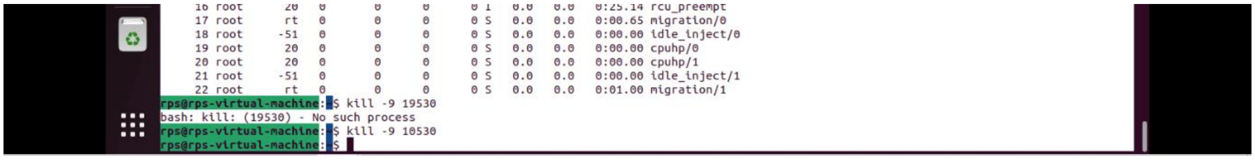
b. **Create a script that backs up a remote directory on another server to a local directory using tools like scp.**

```
rps@rps-virtual-machine:~$ vim 6b.sh
rps@rps-virtual-machine:~$ cat 6b.sh
#!/bin/src/env bash

read -p "Enter the remote user: " remote_user
read -p "Enter the remote server: "remote_server
read -p "Enter the remote directory: "remote_dir
read -p "Enter the local directory: "local_dir

scp -r "$remote_user@$remote_$server:$remote_dir" "$local_dir"
echo "Backup completed."
rps@rps-virtual-machine:~$ 
```

7. Training Codes



```
rps@rps-virtual-machine:~$ man scp
rps@rps-virtual-machine:~$
rps@rps-virtual-machine:~$ man kill
rps@rps-virtual-machine:~$ kill -l
 1) SIGHUP       2) SIGINT       3) SIGQUIT      4) SIGILL       5) SIGTRAP
 6) SIGABRT      7) SIGBUS       8) SIGFPE       9) SIGKILL     10) SIGUSR1
11) SIGSEGV     12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
21) SIGTTIN     22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
31) SIGSYS      34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3  62) SIGRTMAX-2
63) SIGRTMAX-1  64) SIGRTMAX
rps@rps-virtual-machine:~$ vim signalex1.sh
rps@rps-virtual-machine:~$ top

top - 12:05:41 up 2 days, 28 min,  1 user,  load average: 1.20, 0.34, 0.11
Tasks: 240 total,   5 running, 235 sleeping,   0 stopped,   0 zombie
%Cpu(s): 20.5 us, 40.1 sy,  0.0 ni, 38.9 id,  0.1 wa,  0.0 hi,  0.4 si,  0.0 st
MiB Mem :  15984.7 total,  12128.4 free,   1117.8 used,   2738.5 buff/cache
MiB Swap:   2048.0 total,   2048.0 free,      0.0 used.  14528.3 avail Mem

    PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
  10530 rps       20   0    9972   3456   3200 R 100.0   0.0   0:27.12 bash
```



```
     16 root      20   0       0      0      0 I   0.0   0.0   0:25.14 rcu_preempt
     17 root      rt   0       0      0      0 S   0.0   0.0   0:00.65 migration/0
     18 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/0
     19 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
     20 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/1
     21 root     -51   0       0      0      0 S   0.0   0.0   0:00.00 idle_inject/1
     22 root      rt   0       0      0      0 S   0.0   0.0   0:01.00 migration/1
rps@rps-virtual-machine:~$ kill -9 19530
bash: kill: (19530) - No such process
rps@rps-virtual-machine:~$ kill -9 10530
rps@rps-virtual-machine:~$ 
```

```
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Killed
rps@rps-virtual-machine:~$ dd
```

```
rps        10540 99.3  0.0   9972  3456 pts/1    R+   12:10   0:46 bash signalex1.sh
rps@rps-virtual-machine:~$ kill sighup 10540
bash: kill: sighup: arguments must be process or job IDs
rps@rps-virtual-machine:~$ sighup 10540
sighup: command not found
rps@rps-virtual-machine:~$ kill sigint 10540
bash: kill: sigint: arguments must be process or job IDs
bash: kill: (10540) - No such process
rps@rps-virtual-machine:~$ ps aux | grep signalex1.sh | grep -v grep
rps@rps-virtual-machine:~$ 
```

```
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Terminated
rps@rps-virtual-machine:~$
```

```
rps         10562  101  0.0   9972  3456 pts/1    R+   12:15   0:07 bash signalex1.sh
rps@rps-virtual-machine:~$ kill sigtrap 10562
bash: kill: sigtrap: arguments must be process or job IDs
rps@rps-virtual-machine:~$ man sigtrap
No manual entry for sigtrap
rps@rps-virtual-machine:~$ kill
kill: usage: kill [-s sigspec | -n signum | -sigspec] pid | jobspec ... or kill -l [sigspec]
rps@rps-virtual-machine:~$ ps aux | grep signalex1.sh | grep -v grep
rps         10567 96.6  0.0   9972  3456 pts/1    R+   12:16   0:28 bash signalex1.sh
rps@rps-virtual-machine:~$ kill 10567
rps@rps-virtual-machine:~$ ps aux | grep signalex1.sh | grep -v grep
rps         10571 95.5  0.0   9972  3328 pts/1    R+   12:16   0:22 bash signalex1.sh
rps@rps-virtual-machine:~$ kill -2 10571
rps@rps-virtual-machine:~$
```

```
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro

rps@rps-virtual-machine:~$ bash signalex1.sh
```

```
rps         10582  102  0.0   9972  3456 pts/1    R+   12:18   0:05 bash signalex1.sh
rps@rps-virtual-machine:~$ kill -3 10582
rps@rps-virtual-machine:~$ ps aux | grep signalex1.sh | grep -v grep
rps         10586 97.8  0.0   9972  3456 pts/1    R+   12:19   0:05 bash signalex1.sh
rps@rps-virtual-machine:~$ kill -1 10586
rps@rps-virtual-machine:~$
```

```
Hello bro
Hello bro
Hello bro
Hangup
rps@rps-virtual-machine:~$
```

```
rps         10590 91.3  0.0   9972  3456 pts/1    R+   12:20   0:05 bash signalex1.sh
rps@rps-virtual-machine:~$ kil -19 10590
Command 'kil' not found, did you mean:
  command 'kig' from snap kig (23.08.3)
  command 'kif' from snap kif (0.2.0)
  command 'kic' from deb kic (2.4c-1)
  command 'cil' from deb cil (0.07.00-12)
  command 'kig' from deb kig (4:21.12.3-0ubuntu1)
  command 'pil' from deb picolisp (21.12-2)
  command 'kile' from deb kile (4:2.9.93-2)
  command 'ull' from deb ull (2.3.8-3)
  command 'kill' from deb procps (2:3.3.17-6ubuntu2.1)
See 'snap info <snapname>' for additional versions.
rps@rps-virtual-machine:~$ ps aux | grep signalex1.sh | grep -v grep
rps         10603  104  0.0   9972  3584 pts/1    R+   12:20   0:10 bash signalex1.sh
rps@rps-virtual-machine:~$ kill -13 10603
rps@rps-virtual-machine:~$ ps aux | grep signalex1.sh | grep -v grep
rps         10607 70.6  0.0   9972  3456 pts/1    R+   12:22   0:04 bash signalex1.sh
rps@rps-virtual-machine:~$ kill -14 10607
rps@rps-virtual-machine:~$
```

```
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro Alarm clock
rps@rps-virtual-machine:~$
```

```
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
Hello bro
```

8. Assignment - 2

(Time : 3:37)

1. Basic Handling vs. Advanced Control:

Implement signal handling using both signal and sigaction (in separate program runs). Observe the behavior. Which API allows for more control over the signal handler? Explain the key difference in a comment within your code.

1. signal handling code

```
rps@rps-virtual-machine:~/signal_ex$ cat  signal2.cpp
#include <iostream>
#include <csignal>
#include <unistd.h>

// Signal handler function
void signalHandler(int signum) {
        std::cout << "Whatever, Interrupt signal (" << signum << ") recieved.\n";
        // cleanup and close up stuff here
        // Terminate program
            exit(signum);
}


void Iamhere(int signal){
        std::cout<<"Here I am \n";
        exit(signal);

}

int main() {
        // Register signal handler for SIGINT
           signal(SIGINT, signalHandler);
           signal(SIGSEGV, signalHandler);
           signal(SIGTSTP, Iamhere);
           while(true) {
                   std::cout << "Running.......\n";
                   sleep(1);  // Sleep for 1 sec
                        }
           return 0;
}
}
rps@rps-virtual-machine:~/signal_ex$  ./ signal2.cpp
bash: ./: Is a directory
rps@rps-virtual-machine:~/signal_ex$ ./signal2
Running.......
Running.......
Running.......
Running.......
Running.......
Running.......
Running.......
Running.......
Running.......
Running.......
^CWhatever, Interrupt signal (2) recieved.
rps@rps-virtual-machine:~/signal_ex$ ./signal2
Running.......
Running.......
Running.......
Running.......
Running.......
^ZHere I am
rps@rps-virtual-machine:~/signal_ex$
```

2. sigaction handling code

```
rps@rps-virtual-machine:~/signal_ex$ cat sigaction.cpp
#include <iostream>
#include <csignal>
#include <unistd.h>
using namespace std;

// Signal handler functions

void signalHandler(int signum) {
        cout <<"Interrupt signal (" << signum << ") recieved.\n";
        // Cleanup and close up stuff here
        // terminate program
        exit(signum);
}

int main(){
        struct sigaction action;
        action.sa_handler = signalHandler;
        sigemptyset(&action.sa_mask);
        action.sa_flags = 0;

        // register signal handles for SIGINT using sigaction
        if (sigaction(SIGINT, &action, nullptr) < 0) {
                cerr << "Error registering signal handler" <<endl;
                return 1;
        }
        while (true) {
                cout << "Running...Press  Ctrl+C to exit.\n";
                sleep(1); // Sleep for 1 second
                        }
        return 0;
}
```

```
rps@rps-virtual-machine:~/signal_ex$ cat  signal2.cpp
#include <iostream>
#include <csignal>
#include <unistd.h>

// Signal handler function
void signalHandler(int signum) {
        std::cout << "Whatever, Interrupt signal (" << signum << ") recieved.\n";
        // cleanup and close up stuff here
        // Terminate program
            exit(signum);
}


void Iamhere(int signal){
        std::cout<<"Here I am \n";
        exit(signal);

}
int main() {
        // Register signal handler for SIGINT
           signal(SIGINT, signalHandler);
           signal(SIGSEGV, signalHandler);
           signal(SIGTSTP, Iamhere);
           while(true) {
                   std::cout << "Running.......\n";
                   sleep(1);  // Sleep for 1 sec
                           }
           return 0;
}
```

In the **signal()** example, the signal handling behavior is more limited and can be less predictable across different Unix-like systems. It provides basic functionality to catch signals.

In the **sigaction()** example, the API offers more control and flexibility, such as the ability to specify flags **(e.g., SA_RESTART, SA_NOCLDSTOP)** and to handle more complex signal handling scenarios.

. sigaction() is the recommended way to handle signals in modern Unix-like systems because of its robustness and portability.

3. Graceful Termination with Signal Handling

Objective: Modify your program to demonstrate graceful termination upon receiving a specific signal (e.g., SIGINT). Within the signal handler, perform any necessary cleanup tasks (e.g., closing files, releasing resources) before exiting the program gracefully.

Implementation:

In your signal handler function, include code to perform cleanup actions.This might involve closing open files, releasing memory, or writing data to disk.

Use exit(0) or similar methods to terminate the program after cleanup is complete.

```
/home/rps/signal_ex
rps@rps-virtual-machine:~/signal_ex$ vim signal_file_Handler.cpp
rps@rps-virtual-machine:~/signal_ex$ make signal_file_Handler
g++      signal_file_Handler.cpp   -o signal_file_Handler
rps@rps-virtual-machine:~/signal_ex$ ./signal_file_Handler
^CCaught signal 2
Closed the file successfully.
rps@rps-virtual-machine:~/signal_ex$ cat example.txt
Running...
Running...
Running...
Running...
rps@rps-virtual-machine:~/signal_ex$ cat signal_file_Handler.cpp
#include <stdio.h>
#include <stdlib.h>
#include <signal.h>
#include <unistd.h>

// File pointer for cleanup demonstration
```

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

// File pointer for cleanup demonstration
 FILE *file;

void signal_handler(int signum) {
        printf("Caught signal %d\n", signum);
// perform necessry cleanup
if (file) {
 fclose(file);
printf("Closed the file successfully.\n");
}
exit(0);
}

int main(){
        file = fopen("example.txt","w");
        if (!file) {
                perror(" Failed to open file");
                return 1;
        }

        // Register signal handler for SIGINT
        signal(SIGINT, signal_handler);

        // Infinite loop to keep the program running
        while (1) {
                fprintf(file, "Running...\n");
                fflush(file);
                sleep(1);
        }
        return 0;
}
rps@rps-virtual-machine:~/signal_ex$
```