1. Linux shell script program to swap two numbers Swapping using third variable and without third VARIABLE.

#!/bin/bash

echo "Enter first number: "

read n1

echo "Enter second number: "

read n2

echo "Before swapping: n1 = $n1, n2 = $n2"

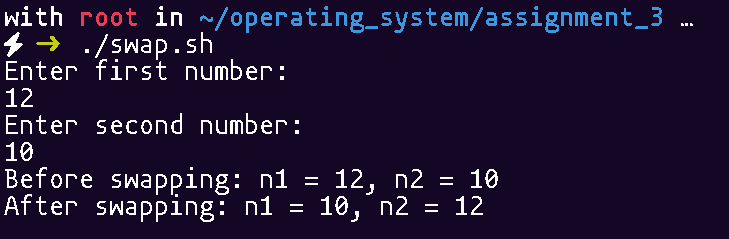
n1=$((n1 ^ n2))

n2=$((n1 ^ n2))

n1=$((n1 ^ n2))

echo "After swapping: n1 = $n1, n2 = $n2"

OUTPUT:



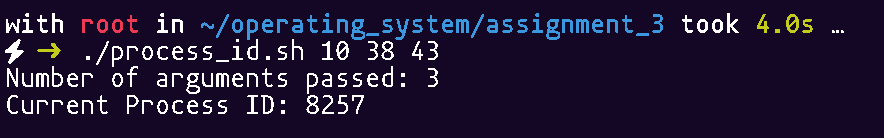
1. Linux shell script program to demonstrate the '$#' variable and Linux shell script program to print thecurrent process id

#!/bin/bash

echo "Number of arguments passed: $#"

echo "Current Process ID: $$"

OUTPUT:



1. Linux shell script to demonstrate comparision operators

#!/bin/bash

echo "Enter first number: "

read num1

echo "Enter second number: "

read num2

if [ $num1 -eq $num2 ]; then

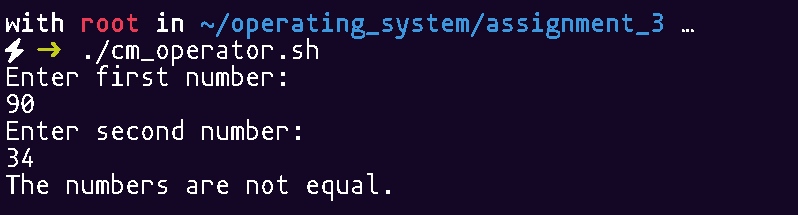
echo "The numbers are equal."

else

echo "The numbers are not equal."

fi

OUTPUT:



1. Write a Shell Script Program to Checking if a File Exists or Not

#!/bin/bash

echo "Enter the file name: "

read filename

if [ -e "$filename" ]; then

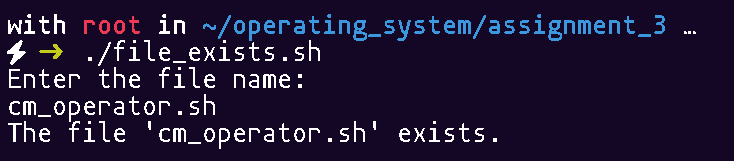
echo "The file '$filename' exists."

else

echo "The file '$filename' does not exist."

fi

OUTPUT:



1. Write a Shell script to determine if a number is positive, negative, or zero

#!/bin/bash

echo "Enter a number: "

read num

if [ $num -gt 0 ]; then

echo "The number $num is positive."

elif [ $num -lt 0 ]; then

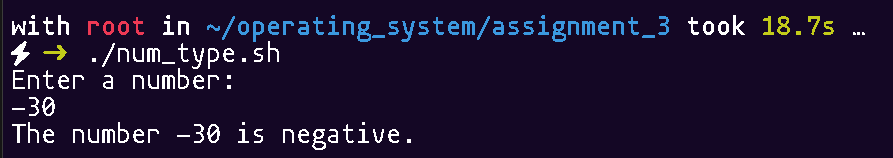
echo "The number $num is negative."

else

echo "The number is zero."

fi

OUTPUT:



1. Write a shell script to manage user accounts on a Linux system. The script should allow the administrator to create, delete, or modify user accounts based on user input.

#!/bin/bash

create\_user() {

read -p "Enter username to create: " username

if id "$username" &>/dev/null; then

echo "User '$username' already exists!"

else

sudo useradd "$username"

echo "User '$username' created successfully!"

sudo passwd "$username"

fi

}

delete\_user() {

read -p "Enter username to delete: " username

if id "$username" &>/dev/null; then

sudo userdel -r "$username"

echo "User '$username' deleted successfully!"

else

echo "User '$username' does not exist!"

fi

}

modify\_user() {

read -p "Enter username to modify: " username

if id "$username" &>/dev/null; then

echo "1. Change Password"

echo "2. Lock User"

echo "3. Unlock User"

echo "4. Add User to Group"

read -p "Choose an option: " choice

case $choice in

1) sudo passwd "$username" ;;

2) sudo usermod -L "$username"; echo "User '$username' locked!" ;;

3) sudo usermod -U "$username"; echo "User '$username' unlocked!" ;;

4) read -p "Enter group name: " group

sudo usermod -aG "$group" "$username"

echo "User '$username' added to group '$group'!" ;;

\*) echo "Invalid option!" ;;

esac

else

echo "User '$username' does not exist!"

fi

}

while true; do

echo "User Management Script"

echo "1. Create User"

echo "2. Delete User"

echo "3. Modify User"

echo "4. Exit"

read -p "Choose an option: " option

case $option in

1) create\_user ;;

2) delete\_user ;;

3) modify\_user ;;

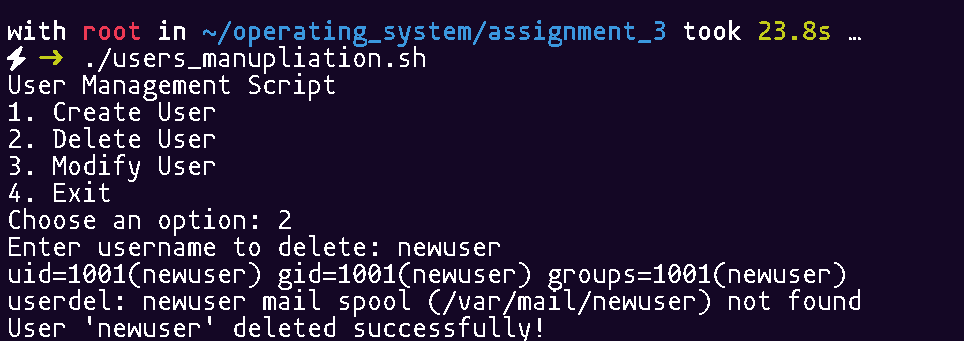
4) echo "Exiting..."; exit 0 ;;

\*) echo "Invalid option! Try again." ;;

esac

done

OUTPUT:



1. Create a Sample Shell Script that removes in the Current Directory that are older than 4 days based on their file name.

#!/bin/bash

days=4

find . -type f -mtime +$days -exec rm -f {} \;

echo "All files older than $days days have been deleted from the current directory."

OUTPUT:



1. Create a script that counts the number of words, lines, and characters in 2 text files.

#!/bin/bash

if [ $# -ne 2 ]; then

echo "Usage: $0 <file1> <file2>"

exit 1

fi

file1=$1

file2=$2

if [ ! -f "$file1" ]; then

echo "Error: File '$file1' does not exist!"

exit 1

fi

if [ ! -f "$file2" ]; then

echo "Error: File '$file2' does not exist!"

exit 1

fi

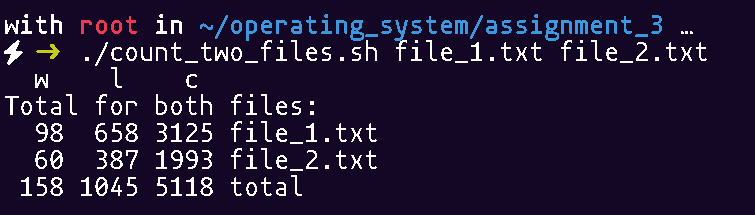
echo " w l c"

echo "Total for both files:"

wc "$file1" "$file2"

exit 0

OUTPUT:



1. Write a Shell Script programs using Switch Case and read the following instruction for that Programs :

1. Create a custom menu using echo statement and show the menu

2. Create an infinite loop using while statement that accept the user input option and generate the output continuously until the user input matches the exit pattern.

3. Take input from the user using read statement and store it in a variable.

4. Use case statement to check if the input matches with the pattern.

5. Create custom pattern.

6. Exit the case statement using esac keyword.

#!/bin/bash

echo "SELECT YOUR FAVORITE FRUIT"

echo "1. Apple"

echo "2. Grapes"

echo "3. Mango"

echo "4. Exit from menu"

while true; do

read -p "Enter your menu choice [1-4]: " choice

case $choice in

1)echo "Selected Fruit is Apple."

;;

2) echo "Selected Fruit is Grapes."

;;

3) echo "Selected Fruit is Mango."

;;

4) echo "Quitting ..."

exit 0

;;

\*) echo "Invalid choice! Please enter a number between 1 and 4."

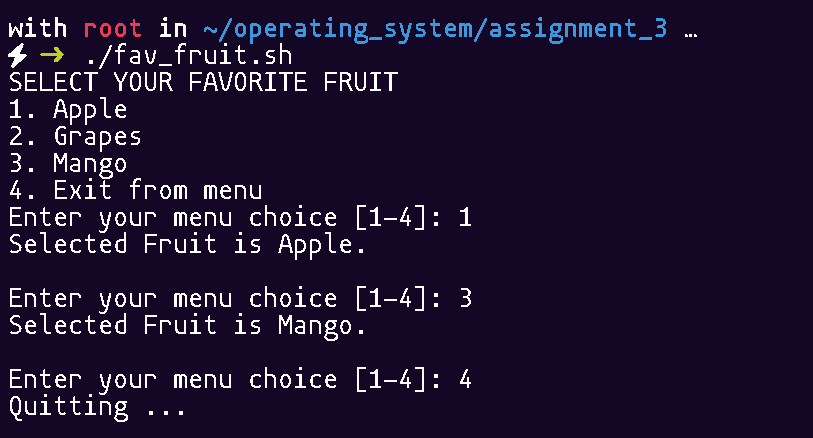
;;

esac

echo

done

OUTPUT:



10.) Write a Shell Script Programs to take input a Char from User and check the Char is Lower case ,Upper Case and Vowels & Consonants

#!/bin/bash

read -p "Enter a single character: " char

if [[ ${#char} -ne 1 ]]; then

echo "Error: Please enter only a single character!"

exit 1

fi

if [[ "$char" =~ [a-z] ]]; then

echo "Lowercase letter."

if [[ "$char" =~ [aeiou] ]]; then

echo "It is a vowel."

else

echo "It is a consonant."

fi

elif [[ "$char" =~ [A-Z] ]]; then

echo "Uppercase letter."

if [[ "$char" =~ [AEIOU] ]]; then

echo "It is a vowel."

else

echo "It is a consonant."

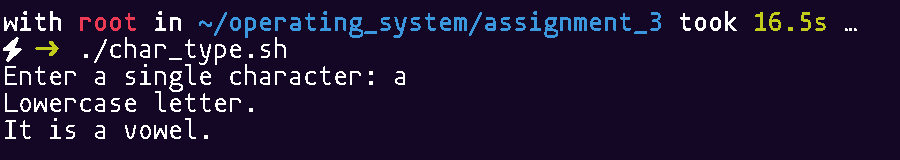
fi

else

echo "The character '$char' is not a letter."

fi

OUTPUT:



1. Write a shell script to find out the greatest among three inputs.

#!/bin/bash

read -p "Enter first number: " num1

read -p "Enter second number: " num2

read -p "Enter third number: " num3

if (( num1 >= num2 && num1 >= num3 )); then

echo "The greatest number is: $num1"

elif (( num2 >= num1 && num2 >= num3 )); then

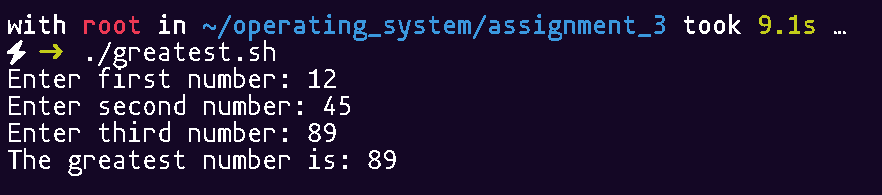
echo "The greatest number is: $num2"

else

echo "The greatest number is: $num3"

fi

OUTPUT:



12.) Write a shell script to calculate the net salary of an employee in a particular month considering various allowances (TA, DA, HRA) and deductions (INCOME TAX, PROVIDEND FUND) as:

a.TA=15 percent of basic salary

b.DA=2 percent of basic salary

c.HRA=10 percent of basic salary

d.INCOME TAX=5 percent of salary

e.PROVIDEND FUND=10 percent of salary

#!/bin/bash

read -p "Enter Basic Salary: " basic\_salary

TA=$(echo "$basic\_salary \* 0.15" | bc)

DA=$(echo "$basic\_salary \* 0.02" | bc)

HRA=$(echo "$basic\_salary \* 0.10" | bc)

gross\_salary=$(echo "$basic\_salary + $TA + $DA + $HRA" | bc)

income\_tax=$(echo "$gross\_salary \* 0.05" | bc)

provident\_fund=$(echo "$gross\_salary \* 0.10" | bc)

net\_salary=$(echo "$gross\_salary - $income\_tax - $provident\_fund" | bc)

echo "Basic Salary: $basic\_salary"

echo "TA (15%): $TA"

echo "DA (2%): $DA"

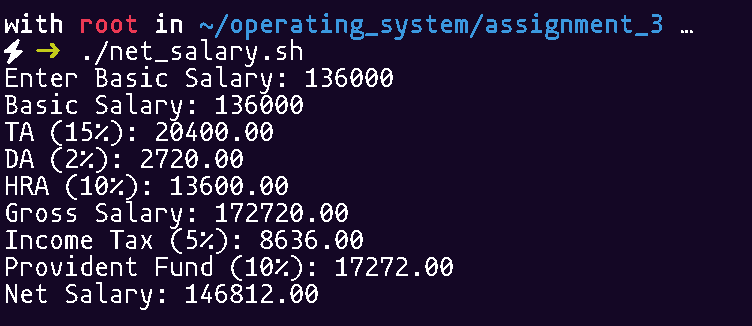
echo "HRA (10%): $HRA"

echo "Gross Salary: $gross\_salary"

echo "Income Tax (5%): $income\_tax"

echo "Provident Fund (10%): $provident\_fund"

echo "Net Salary: $net\_salary"

OUTPUT: 

13.) A departmental store announces its festival scheme to customers on cash payment. The scheme is asfollows

a.If purchase amount is less than 1000 then Tax=2% and discount=10%.

b.If purchase amount is greater than 1000 then Tax=5 % and discount=20%.

#!/bin/bash

read -p "Enter purchase amount: " amount

if (( amount < 1000 )); then

tax=$(( amount \* 2 / 100 ))

discount=$(( amount \* 10 / 100 ))

else

tax=$(( amount \* 5 / 100 ))

discount=$(( amount \* 20 / 100 ))

fi

final\_amount=$(( amount + tax - discount ))

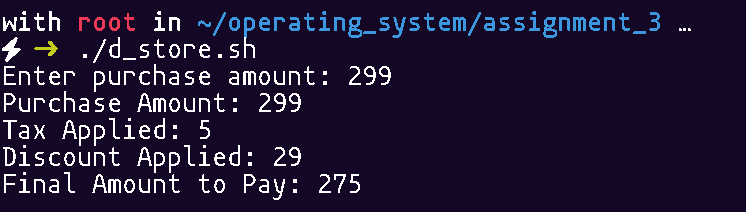
echo "Purchase Amount: $amount"

echo "Tax Applied: $tax"

echo "Discount Applied: $discount"

echo "Final Amount to Pay: $final\_amount"

OUTPUT:



14.) Write a C/JAVA Program for system calls of Unix operating systems (opendir, readdir, closedir)

#include <stdio.h>

#include <dirent.h>

int main() {

struct dirent \*entry;

DIR \*dir = opendir(".");

if (dir == NULL) {

perror("Unable to open directory");

return 1;

}

printf("Files in the current directory:\n");

while ((entry = readdir(dir)) != NULL) {

printf("%s\n", entry->d\_name);

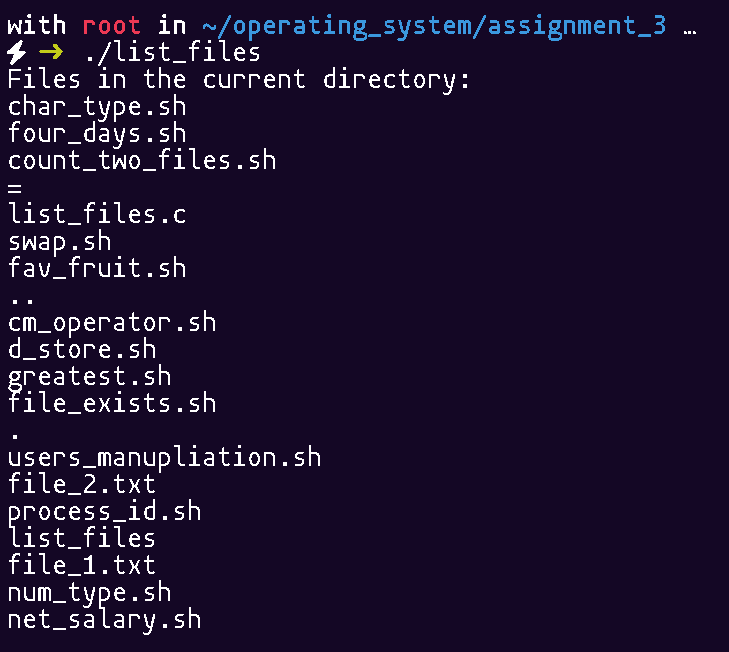
}

closedir(dir);

return 0;

}

OUTPUT:



15.) Write a C/JAVA Program for Process system calls of Unix operating systems (fork, getpid, exit)

import java.io.File;

public class list\_files {

public static void main(String[] args) {

File directory = new File("."); // Current directory

if (!directory.isDirectory()) {

System.out.println("Error: Not a valid directory.");

return;

}

String[] files = directory.list();

if (files != null) {

System.out.println("Files in the current directory:");

for (String file : files) {

System.out.println(file);

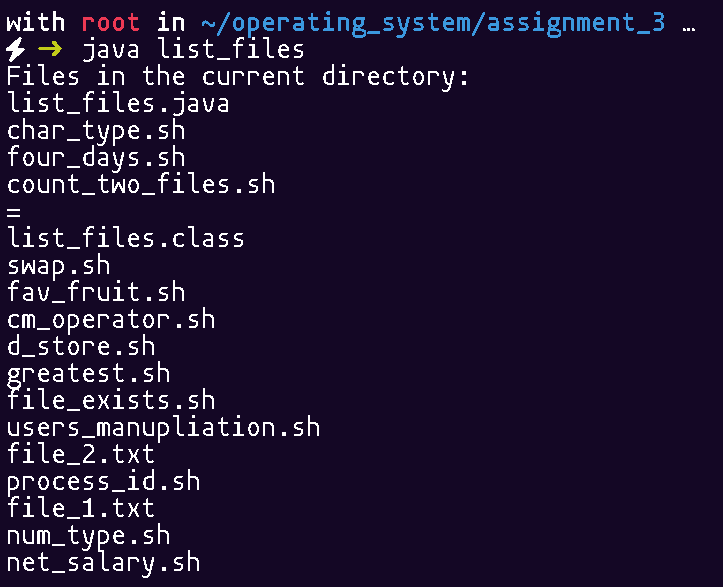
}

} else {

System.out.println("Error reading directory.");

}}}

OUTPUT:



16.) Write a C/JAVA Program to implement the system calls wait ( ) and exit ( )

class Task extends Thread {

public void run() {

System.out.println("Task started...");

try { Thread.sleep(2000); } catch (InterruptedException e) {}

System.out.println("Task completed.");

synchronized (this) { notify(); }

}}

public class wait\_exit {

public static void main(String[] args) throws InterruptedException {

Task task = new Task();

task.start();

synchronized (task) {

System.out.println("Main thread waiting...");

task.wait();

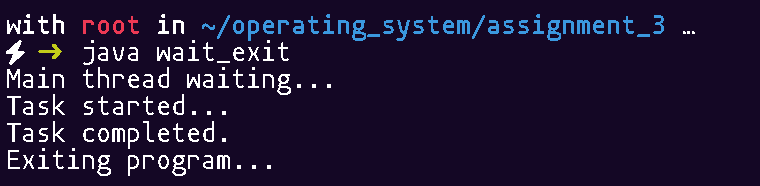
}

System.out.println("Exiting program...");

System.exit(0);

}}

OUTPUT:



17.) Write a C/JAVA Program to implement the system call execl ( ).

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class execc {

public static void main(String[] args) {

try {

System.out.println("Executing ls command...");

Process process = Runtime.getRuntime().exec("ls");

BufferedReader reader = new BufferedReader(new InputStreamReader(process.getInputStream()));

String line;

while ((line = reader.readLine()) != null) {

System.out.println(line);

}

process.waitFor();

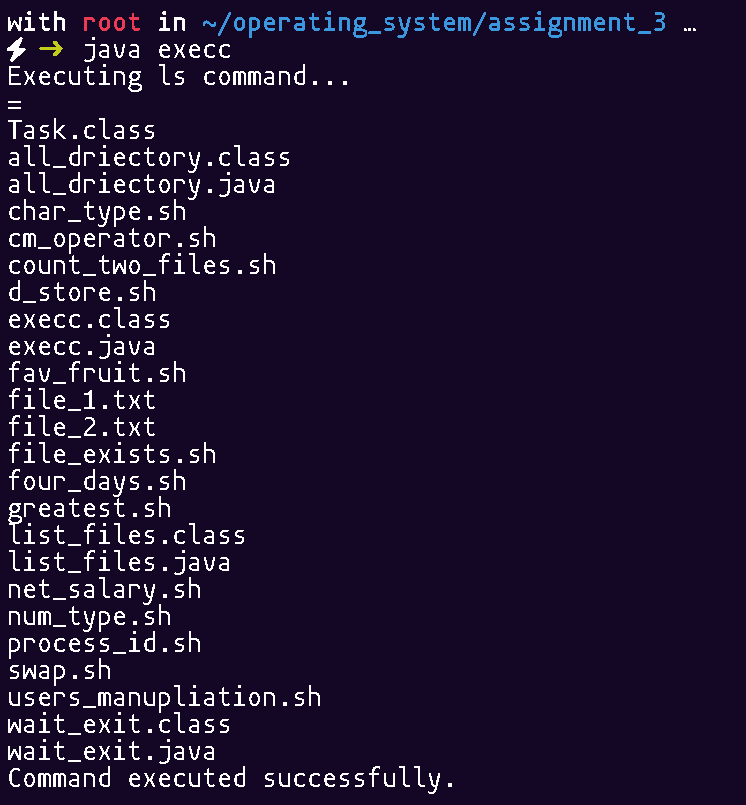
System.out.println("Command executed successfully.");

} catch (IOException | InterruptedException e) {

e.printStackTrace();

} }

}OUTPUT:



18.) Write a C/JAVA Program to implement the system call execv ( )

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

public class list\_l {

public static void main(String[] args) {

try {

String[] command = {"/bin/ls", "-l", "/root/operating\_system"};

Process process = Runtime.getRuntime().exec(command);

BufferedReader reader = new BufferedReader(new InputStreamReader(process.getInputStream()));

String line;

while ((line = reader.readLine()) != null) {

System.out.println(line);

}

process.waitFor();

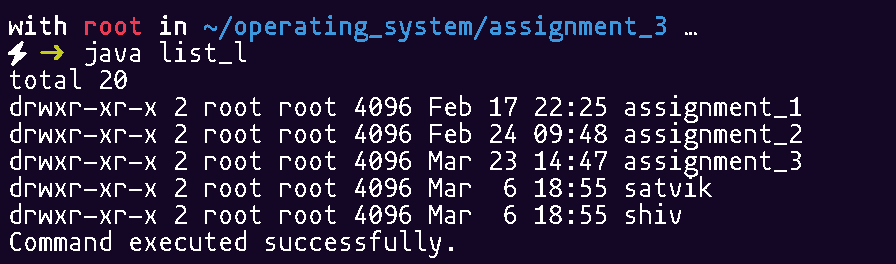
System.out.println("Command executed successfully.");

} catch (IOException | InterruptedException e) {

e.printStackTrace();

}}}

OUTPUT:



19.) Write a ‘c’ program for I/O system calls.

#include <stdio.h>

#include <fcntl.h>

#include <unistd.h>

void main() {

int fd;

char buffer[100];

// Creating and writing to a file

fd = open("write.txt", O\_CREAT | O\_WRONLY, 0644);

if (fd < 0) {

perror("Error opening file");

return;

}

write(fd, "Hello, this is a test for I/O system calls.\n", 44);

close(fd);

printf("Data written to write.txt\n");

// Reading from a file

fd = open("write.txt", O\_RDONLY);

if (fd < 0) {

perror("Error opening file for reading");

return;

}

printf("Reading from write.txt:\n");

int bytes\_read = read(fd, buffer, sizeof(buffer) - 1);

if (bytes\_read > 0) {

buffer[bytes\_read] = '\0';

printf("%s\n", buffer);

}

close(fd);

// Taking user input

printf("Enter some text: ");

bytes\_read = read(STDIN\_FILENO, buffer, sizeof(buffer) - 1);

buffer[bytes\_read - 1] = '\0'; // Remove newline

//Writing user input to a new file

fd = open("user\_input.txt", O\_CREAT | O\_WRONLY, 0644);

if (fd < 0) {

perror("Error opening file for writing");

return;

}

write(fd, buffer, bytes\_read);

close(fd);

printf("User input saved to user\_input.txt\n");

}

OUTPUT:

