PRACTICAL NO: 07 [A]

Name: Riyan Shaikh

Roll no: T512051 [B]

CODE:

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <string.h>

#include <sys/types.h>

#include <sys/wait.h>

#define BUFFER\_SIZE 1024

// Function to count characters, words, and lines

void count\_stats(const char\* input, int\* chars, int\* words, int\* lines) {

\*chars = 0;

\*words = 0;

\*lines = 0;

int in\_word = 0;

for (int i = 0; input[i] != '\0'; i++) {

(\*chars)++;

if (input[i] == '\n') (\*lines)++;

if (input[i] == ' ' || input[i] == '\n' || input[i] == '\t') {

in\_word = 0;

} else if (in\_word == 0) {

in\_word = 1;

(\*words)++;

}

}

}

int main() {

int pipe1[2], pipe2[2];

pid\_t pid;

char buffer[BUFFER\_SIZE];

int chars, words, lines;

FILE \*file;

// Create two pipes

if (pipe(pipe1) == -1 || pipe(pipe2) == -1) {

perror("pipe");

exit(EXIT\_FAILURE);

}

pid = fork();

if (pid < 0) {

perror("fork");

exit(EXIT\_FAILURE);

}

if (pid > 0) { // Parent process (Process 1)

close(pipe1[0]); // Close reading end of pipe1

close(pipe2[1]); // Close writing end of pipe2

printf("Enter a sentence: ");

fgets(buffer, BUFFER\_SIZE, stdin);

// Write the sentence to process 2

write(pipe1[1], buffer, strlen(buffer) + 1);

close(pipe1[1]); // Close writing end of pipe1 after sending

// Wait for response from process 2

wait(NULL);

read(pipe2[0], buffer, BUFFER\_SIZE);

close(pipe2[0]); // Close reading end of pipe2

printf("Result from process 2:\n%s", buffer);

} else { // Child process (Process 2)

close(pipe1[1]); // Close writing end of pipe1

close(pipe2[0]); // Close reading end of pipe2

// Read the sentence from process 1

read(pipe1[0], buffer, BUFFER\_SIZE);

close(pipe1[0]); // Close reading end of pipe1 after receiving

// Count characters, words, and lines

count\_stats(buffer, &chars, &words, &lines);

// Write the result to a file

file = fopen("output.txt", "w");

if (file == NULL) {

perror("fopen");

exit(EXIT\_FAILURE);

}

fprintf(file, "Characters: %d\nWords: %d\nLines: %d\n", chars, words, lines);

fclose(file);

// Read the contents of the file and send them to process 1

file = fopen("output.txt", "r");

if (file == NULL) {

perror("fopen");

exit(EXIT\_FAILURE);

}

fread(buffer, sizeof(char), BUFFER\_SIZE, file);

fclose(file);

// Write the file contents to pipe2

write(pipe2[1], buffer, strlen(buffer) + 1);

close(pipe2[1]); // Close writing end of pipe2 after sending

exit(0);

}

return 0;

}

OUTPUT:

