**#include <stdio.h>**

**#include <stdlib.h>**

**#define CRITICAL\_THRESHOLD 23.0**

**#define SAFE\_MAX 50.0**

**float celsius\_to\_fahrenheit(float celsius) {**

**return (celsius \* 9 / 5) + 32;**

**}**

**int main() {**

**FILE \*file;**

**char filename[] = "temp.txt";**

**float temperature\_c, temperature\_f;**

**file = fopen(filename, "r");**

**if (file == NULL) {**

**printf("Error: Could not open file %s\n", filename);**

**return 1;**

**}**

**printf("| %-10s | %-10s | %-10s |\n", "Temp (C)", "Temp (F)", "Status");**

**printf("----------------------------------------------\n");**

**while (1) {**

**int result = fscanf(file, "%f", &temperature\_c);**

**if (result != 1) {**

**break;**

**}**

**temperature\_f = celsius\_to\_fahrenheit(temperature\_c);**

**char status[20];**

**if (temperature\_f < CRITICAL\_THRESHOLD) {**

**snprintf(status, sizeof(status), "Below");**

**} else if (temperature\_f > SAFE\_MAX) {**

**snprintf(status, sizeof(status), "Above");**

**} else {**

**snprintf(status, sizeof(status), "Right");**

**}**

**printf("| %-10.2f | %-10.2f | %-10s |\n", temperature\_c, temperature\_f, status);**

**}**

**fclose(file);**

**return 0;**

**}**

gedit temp.txt

gedit Q1.c

gcc Q1.c

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <unistd.h>**

**#include <string.h>**

**#include <ctype.h>**

**#include <sys/wait.h>**

**#define FILENAME "textfile.txt"**

**#define MAX\_LINE\_LENGTH 256**

**void writeToFile() {**

**FILE \*file = fopen(FILENAME, "w");**

**if (file == NULL) {**

**perror("Error opening file");**

**exit(EXIT\_FAILURE);**

**}**

**printf("Enter lines of text (type 'END' to stop):\n");**

**char line[MAX\_LINE\_LENGTH];**

**while (1) {**

**fgets(line, sizeof(line), stdin);**

**if (line[0] == 'E' && line[1] == 'N' && line[2] == 'D' &&**

**(line[3] == '\n' || line[3] == '\0')) {**

**break;**

**}**

**fputs(line, file);**

**}**

**fclose(file);**

**}**

**void convertToUppercase() {**

**FILE \*file = fopen(FILENAME, "r+");**

**if (file == NULL) {**

**perror("Error opening file");**

**exit(EXIT\_FAILURE);**

**}**

**char ch;**

**long pos;**

**while ((pos = ftell(file)) >= 0) {**

**ch = fgetc(file);**

**if (ch == -1) {**

**break;**

**}**

**if (islower(ch)) {**

**fseek(file, pos, SEEK\_SET);**

**fputc(toupper(ch), file);**

**}**

**}**

**fclose(file);**

**}**

**void printFileContent() {**

**FILE \*file = fopen(FILENAME, "r");**

**if (file == NULL) {**

**perror("Error opening file");**

**exit(EXIT\_FAILURE);**

**}**

**char line[MAX\_LINE\_LENGTH];**

**printf("\nFinal content of the file:\n");**

**while (fgets(line, sizeof(line), file) != NULL) {**

**printf("%s", line);**

**}**

**fclose(file);**

**}**

**int main() {**

**pid\_t pid1, pid2;**

**writeToFile();**

**pid1 = fork();**

**if (pid1 == -1) {**

**perror("Fork failed");**

**exit(EXIT\_FAILURE);**

**} else if (pid1 == 0) {**

**convertToUppercase();**

**exit(EXIT\_SUCCESS);**

**}**

**waitpid(pid1, NULL, 0);**

**pid2 = fork();**

**if (pid2 == -1) {**

**perror("Fork failed");**

**exit(EXIT\_FAILURE);**

**} else if (pid2 == 0) {**

**printFileContent();**

**exit(EXIT\_SUCCESS);**

**}**

**waitpid(pid2, NULL, 0);**

**return 0;**

**}**

gedit Q2.c

gcc Q2.c

./a.out

**#include <stdio.h>**

**#include <stdlib.h>**

**#include <unistd.h>**

**#include <string.h>**

**#include <semaphore.h>**

**#include <fcntl.h>**

**void print\_PID() {**

**int PID = getpid();**

**char str[100];**

**sprintf(str, "My process id is %d\n", PID);**

**for (int i = 0; i < strlen(str); i++) {**

**fprintf(stderr, "%c", str[i]);**

**sleep(1);**

**}**

**}**

**int main() {**

**sem\_t \*sem;**

**sem = sem\_open("/print\_semaphore", O\_CREAT, 0644, 1);**

**if (sem == SEM\_FAILED) {**

**perror("sem\_open");**

**exit(1);**

**}**

**pid\_t pid;**

**int num\_children = 3;**

**for (int i = 0; i < num\_children; i++) {**

**pid = fork();**

**if (pid < 0) {**

**perror("fork");**

**exit(1);**

**}**

**if (pid == 0) {**

**sem\_wait(sem);**

**print\_PID();**

**sem\_post(sem);**

**exit(0);**

**}**

**}**

**for (int i = 0; i < num\_children; i++) {**

**wait(NULL);**

**}**

**sem\_close(sem);**

**sem\_unlink("/print\_semaphore");**

**return 0;**

**}**

gedit Q3.c

gcc Q3.c -lrt

./a.out