NOMBRE: CHRISTOFER FABIÁN CHÁVEZ CARAZAS

1. Ejercicio 1

```c
int sum = 0;

void * hilo(void * arg){
    int id = *((int*)arg);
    int sumt = 0;
    for(int i = 0; i < 10; i++){
        int r = rand() %100 + 1;
        sumt += r;
        printf("Soy el hilo %d, numero generado -> %d\n",id,r);
        //sleep(rand() %3 + 1);
        sleep(1);
    }
    printf("Soy el hilo %d, la suma obtenida es %d\n",id,sumt);
    sum += sumt;
}

int main(){
    int n = 3;
    srand(time(NULL));
    pthread_t hilos[n];
    int id[3];
    for(int i = 0; i < n; i++){
        id[i] = i;
        pthread_create(&hilos[i],NULL,hilo,(void*)&id[i]);
    }
    for(int i = 0; i < n; i++){
        pthread_join(hilos[i],NULL);
    }
    printf("Soy el padre, la suma total es %d\n",sum);
}
```

2. Ejercicio 2

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <pthread.h>
#include <unistd.h>

pthread_mutex_t mutex;

int sum = 0;

struct Rango{
    int ini;
    int fin;
};

void * hilo(void * arg){
    struct Rango r = *((struct Rango*)arg);
    int sumt = 0;
    for(int i = r.ini; i <= r.fin; i++){
        sumt += i;
    }
    pthread_mutex_lock(&mutex);
    sum += sumt;
    pthread_mutex_unlock(&mutex);
}

int main(){
    int n = 1000000;
    int h = 4;
    struct Rango rangos[h];
    pthread_t hilos[h];
    int pib = n / h;
    int actual = 0;
    for(int i = 0; i < h; i++){
        rangos[i].ini = actual + 1;
        actual += pib;
        rangos[i].fin = actual;
        if(i == h-1) rangos[i].fin = n;
        pthread_create(&hilos[i],NULL,hilo,(void*)&rangos[i]);
    }
    for(int i = 0; i < h; i++){
        pthread_join(hilos[i],NULL);
    }
    printf("La suma total es %d\n",sum);
}
```

3. Ejercicio 3

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <pthread.h>
#include <unistd.h>
#include <string.h>
#include <math.h>

#define sizeID 3
#define sizeDesc 50
#define sizePre 6

typedef struct {
char id[sizeID];
char desc[sizeDesc];
int temp;
char pre[sizePre];
}Tarea;

typedef struct{
Tarea tarea;
pthread_t hilo;
}Hilo;

void * thLectura(void *arg){
    static char str[100];
    printf("Ingrese el archivo\n");
    gets(str);
    FILE * fp = fopen(str,"r");
    if(fp == NULL){
        return NULL;
    }
    return (void*) &str;
}

int leerFichero(char fichero[], Tarea listaTareas[]){
    FILE * fp = fopen(fichero,"r");
    char id[sizeID];
    char desc[sizeDesc];
    int temp = 0;
    char pre[sizePre];
    for(int i = 0; i < sizeID; i++) id[i] = '\0';
    for(int i = 0; i < sizeDesc; i++) desc[i] = '\0';
    temp = 0;
    for(int i = 0; i < sizePre; i++) pre[i] = '\0';
    int estado = 0;
    int sid = 0;
    int sdesc = 0;
    int spre = 0;
    int slt = 0;
    double stemp = 0;
    char c[100];

    while(feof(fp) == 0){
        char c = fgetc(fp);
        if(c == -1) break;
        if(estado == 0){
            if(c == '-') estado = 1;
            else{
                id[sid] = c;
                sid++;
            }
        }
        else if(estado == 1){
            if(c == '-') estado = 2;
            else{
                desc[sdesc] = c;
                sdesc++;
            }
        }
        else if(estado == 2){
            if(c == '-') estado = 3;
            else if(c == 10){
```

```c
                estado = 0;
                sid = 0;
                sdesc = 0;
                spre = 0;
                stemp = 0;
                Tarea t;
                strncpy(t.id,id,sizeID);
                strncpy(t.desc,desc,sizeDesc);
                strncpy(t.pre,pre,sizePre);
                t.temp = temp;
                printf("HOLA\n");
                listaTareas[slt] = t;
                slt++;
                for(int i = 0; i < sizeID; i++) id[i] = '\0';
                for(int i = 0; i < sizeDesc; i++) desc[i] = '\0';
                temp = 0;
                for(int i = 0; i < sizePre; i++) pre[i] = '\0';
            }
            else{
                temp = temp * pow(10,stemp);
                temp += ((int) (c - 48));
                stemp++;
            }
        }
        else if(estado == 3){
            if(c == 10){
                estado = 0;
                sid = 0;
                sdesc = 0;
                spre = 0;
                stemp = 0;
                Tarea t;
                strncpy(t.id,id,sizeID);
                strncpy(t.desc,desc,sizeDesc);
                strncpy(t.pre,pre,sizePre);
                t.temp = temp;
                listaTareas[slt] = t;
                slt++;
                for(int i = 0; i < sizeID; i++) id[i] = '\0';
                for(int i = 0; i < sizeDesc; i++) desc[i] = '\0';
                temp = 0;
                for(int i = 0; i < sizePre; i++) pre[i] = '\0';
            }
            else{
                pre[spre] = c;
                spre++;
            }
        }
    }

    fclose(fp);
    return slt;
}

void mostrarTareas(Tarea lista[], int slt){
    for(int i = 0; i < slt; i++){
        printf("%s ",lista[i].id);
        printf("%s ",lista[i].desc);
        printf("%d ",lista[i].temp);
        printf("%s\n", lista[i].pre);
        /*for(int j = 0; j < sizePre; j++){
            char c = lista[i].pre[j];
            printf("%c", c);
        }
        printf("\n");
        */
    }

}

void *hacerTarea(void * arg){
    Tarea tarea = *((Tarea*) arg);
    printf("La tarea %s ha COMENZADO\n",tarea.desc);
    sleep(tarea.temp);
    printf("La tarea %s ha TERMINADO\n",tarea.desc);
}


int main(){
    void * file = NULL;
    pthread_t comprobador;
    pthread_create(&comprobador,NULL,thLectura,NULL);
    pthread_join(comprobador,&file);
    if(file == NULL){
        printf("EL archivo no existe\n");
        return 0;
    }
    char *str = (char *) file;
    Tarea listaTareas[50];
    char *temp;
    int slt = leerFichero(str,listaTareas);
    mostrarTareas(listaTareas,slt);
    Hilo hilos[slt];
    for(int i = 0; i < slt; i++){
        Hilo hilotemp;
        hilos[i] = hilotemp;
        hilos[i].tarea = listaTareas[i];
        char temp[sizeID];
        int st = 0;
        for(int k = 0; k < sizeID; k++) temp[k] = '\0';
        for(int j = 0; j < sizePre; j++){
            if(listaTareas[i].pre[j] == '+'){
                for(int k = 0; k < i; k ++){
                    if(strcmp(hilos[k].tarea.id,temp) == 0){
                        pthread_join(hilos[k].hilo,NULL);
                        break;
                    }
                }
                st = 0;
                for(int k = 0; k < sizeID; k++) temp[k] = '\0';
            }
            else if(listaTareas[i].pre[j] == '\0'){
                for(int k = 0; k < i; k ++){
                    if(strcmp(hilos[k].tarea.id,temp) == 0){
                        pthread_join(hilos[k].hilo,NULL);
                        break;
                    }
                }
                st = 0;
                for(int k = 0; k < sizeID; k++) temp[k] = '\0';
                break;
            }
            else{
                temp[st] = listaTareas[i].pre[j];
                st++;
            }
        }
        pthread_create(&hilos[i].hilo,NULL,hacerTarea,(void*)&listaTareas[i]);
    }
    for(int i = 0; i < slt; i++){
        pthread_join(hilos[i].hilo,NULL);
    }
    //printf("%s\n", temp);

}
```