Ejercicio

Se dispone de un archivo de texto que contiene una lista de números enteros. El primer valor indica cuantos números hay en el resto del archivo.

Escriba un programa que lea el archivo en un array en memoria principal y calcule el valor máximo utilizando varios hilos de ejecución. Para calcular el valor máximo se repartirá el array en tantas partes como hilos en ejecución y se asignará una parte del array a cada uno de los hilos de ejecución. Cada hilo deberá calcular el valor máximo de los elementos en su parte asignada. Posteriormente el programa principal calculará el máximo global.

Solución

```
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

struct parametros {
  int inicio;
  int fin;
  int * vec;
  int max;
};
typedef struct parametros param_t;

void * busca_max(void * p) {
  int i;
  param_t * param = (param_t*)p;
  param->max = param->vec[param->inicio];
  for (i=0;i!=param->fin;++i) {
    if (param->vec[i]>param->max) {
      param->max = param->vec[i];
    }
  }
  pthread_exit(0);
  return NULL;
}
```

```
int calcula_maximo(int * vec, int tam, int nhilos) {
  param_t * param;
  pthread_t * th;
  int maximo;
  int i;

  param = malloc(sizeof(param_t) * nhilos);
  th = malloc(sizeof(pthread_t) * nhilos);

  for (i=0;i!=nhilos;++i) {
    param[i].inicio = tam/nhilos;
    param[i].fin=(i==nhilos-1)?tam:(tam+1)/nhilos;
    param[i].vec = vec;
    param[i].max=0;
  }

  for (i=0;i!=nhilos;++i) {
    pthread_create(&th[i], NULL, busca_max, (void*)&param[i]);
  }

  for (i=0;i!=nhilos;++i) {
    pthread_join(th[i], NULL);
  }

  maximo = param[0].max;
  for (i=1;i!=nhilos;++i) {
    if (param[i].max > maximo)
      maximo = param[i].max;
  }

  free(param);
  free(th);

  return maximo;
}
```

```
int main(int argc, char ** argv) {
  FILE * fich;
  int n, nhilos, i;
  int * vec;
  int ret;
  int maximo;

  if (argc!=3) {
    fprintf(stderr, "Error en número de argumentos\n");
    exit(-1);
  }

  fich = fopen(argv[1], "r");
  if (fich == NULL) {
    perror("No se puede abrir fichero");
    exit(-2);
  }

  ret = fscanf(fich, "%d", &n);
  if (ret !=1) {
    fprintf(stderr, "No se puede leer tamaño\n");
    exit(-3);
  }

  nhilos = atoi(argv[2]);

  vec = malloc(sizeof(int) * n);
  for (i=0;i!=n;++i) {
    ret = fscanf(fich, "%d", &vec[i]);
    if (ret !=1) {
      fprintf(stderr, "No se puede leer elemento nro %d\n", i);
      fclose(fich);
      free(vec);
    }
  }

  maximo = calcula_maximo(vec,n,nhilos);
  printf("Máximo: %d\n", maximo);

  fclose(fich);
  free(vec);
  return 0;
}
```