```
//---------------------------------------------------------------------------
// File:   practica_3.cpp
// Authors:Sergio Salesa Quintanilla NIP:811019
//         Rubén Martín González NIP:841886
// Date:   5 de noviembre de 2022
// Coms:   Práctica 3 de PSCD
//         Compilar mediante
//         make -f Makefile_p3
//---------------------------------------------------------------------------

#include <iostream>
#include <thread>
#include <string>
#include <fstream>
#include <atomic>
#include "librerias/Logger_V3/Logger.hpp"
#include "librerias/Semaphore_V4/Semaphore_V4.hpp"
using namespace std;


// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
const int N_EST = 60; // # de estudiantes
const int N_FIL = N_EST /2; // # de filas en la matriz
const int N_COL = 1000; // # de columnas

// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
void meterDatos(int D[][N_COL],string nombre){
        ifstream f(nombre);
        if (f.is_open())
        {
        for (unsigned int i = 0; i < N_FIL; i++)
        {
        for (unsigned int j = 0; j < N_COL; j++)
        {
                f >> D[i][j];
        }
        }
        }
}
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -


/// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
// Pre : < fila > es un ´ı ndice de fila de <D >
// Post : devuelve el m ´a ximo de la fila < fila >
int maxFila ( int D [ N_FIL ][ N_COL ] , int fila ) {
        int max = 0;
        for (unsigned int i = 0; i < N_COL; i++)
```

```
        {
        if (D[fila][i]>max)
        {
        max=D[fila][i];
        }
        }
        return max ;
}
// Pre : < fila > es un ´ı ndice de fila de <D >
// Post : devuelve la suma de los els . de la fila < fila >
int sumaFila ( int D [ N_FIL ][ N_COL ] , int fila ) {
        int sum = 0;
        for (unsigned int i = 0; i < N_COL; i++)
        {
        sum = sum + D[fila][i];
        }
        return sum ;
}
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
void Estudiante (int nip,int& libre,int examinado[2],int parejas[N_EST],int D [ N_FIL ][ N_COL
],bool fin[N_EST],int result[N_EST] ,int& terminados) {
   <await libre>0
         libre--;
         examinado[libre]=nip;
   >

        <await parejas[nip]!=-1>        // esperar me sea asignada pareja y fila

        if (nip < miPareja) {
        res[parejas[nip]]=maxFila(D,fila);       // calcular max de mi fila
        <fin[pareja[nip]]=true;>          // hacerselo llegar a mi pareja
        }
        else {
        int suma=sumaFila(D,fila);     // calcular la suma de mi fila
        <await fin[nip]
        filaExam++;               // comunicar finalizacion
        >
        cout<<"El máximo de la fila es "+to_string(res[nip])+" y la suma de la fila es
"+to_string(suma)<<endl;                                    // coger info de max ( de mi pareja )
        // mostrar resultados
        }
}
// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
void Profesor (int& libre,bool fin[N_EST], int parejas[N_EST],int examinado[2],int
asigFila[N_FIL],int& filaExam) {
        for ( int i =0; i < N_FIL ; i ++) {
        <await libre===0
        pareja[examinado[1]]=examinado[0];
```

```
        pareja[examinado[0]]=examinado[1];
        asigFila[examinado[1]]=i;
        asigFila[examinado[0]]=i;
        libre=2;
        >
        // esperar a que haya dos
        // comunicar a cada uno su pareja , y la fila que les toca
        }
        <await filaExam==N_FIL>
        // esperar que todos hayan terminado
}

// - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
int main () {
int D [ N_FIL ][ N_COL ]; // para almacenar los datos
        int filaExam = 0; // filas completadas
        int parejas [ N_EST ]; // pareja [ i ] ser ´a la pareja asignada
        int examinado[2];
        int result[N_EST];
        bool fin[N_EST]={false};
        int libre=2;
        string nom = "datos.txt";
        // cargar " datos . txt " en " D "
        meterDatos(D,nom);

        thread th_1(&Profesor,ref(libre),fin,parejas,examinado,asigFila,ref(filaExam));   //se
inician los procesos
        for(int i=0;i<N_EST;i++){
        P[i] = thread(&Estudiante,i,ref(libre),examinado,parejas,D,fin,result,ref(filaExam));
        }

        th_1.join();      //se bloquean los procesos
  for(int i=0;i<N_EST;i++){
        P[i].join();
        }

        cout << " Prueba finalizada \n " ;
        return 0;
}
```