```
/* ---------------------------------------------------------------------------
* File:   practica_4.cpp
* Author: Pablo Angusto Delgado 842255 y Miguel Aréjula Aisa 850068
* Date:   noviembre 2022
* Coms:   Práctica 4 de PSCD
*         Compilar mediante
*          make -f Makefile_p4
* --------------------------------------------------------------------------- */

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

//-----------------------------------------------------
//Pre: <fila> es un  índice de fila de <D>
//Post: devuelve el máximo de la fila <fila>
int maxFila(int D[N_FIL][N_COL],int fila) {
     int max = D[fila][0];
    for(int i=1; i<N_COL; i++){
        if(max<D[fila][i]){
            max=D[fila][i];
        }
    }
    return max;
}


//Pre: <fila> es un  índice 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 (int i = 0; i < N_COL; i++)
    {
        sum += D[fila][i];
    }

    return sum;
}

//-----------------------------------------------------
void Estudiante(int nip,int& fila,  bool& hayFila, int D[N_FIL][N_COL], int& silla, int resultado[N_EST], bool exa
men_fin[N_EST], int& silla1, int& silla2, int pareja[]) {
    // esperar por una silla libre
    int miFila;
    int miPareja;
    <await (silla<2) // una de las dos esté libre

        if(silla = 0){
            silla1 = nip;
            silla++;
        }else if(silla==1){
            silla2 = nip;
            silla++;
        }
    >
```

```
   //esperar me sea asignada pareja y fila
   <await (hayFila = true)
      miFila = fila;
      miPareja = pareja[nip];
      levantado++;>
      if (nip<miPareja) {
         // calcular máx de mi

         resultado[nip] = maxFila(D,miFila);

      }
      else {
         // calcular la suma de mi fila
         resultado[nip] = sumaFila(D, miFila);
         //coger info de max (de mi pareja)
         <await(examen_fin[miPareja]= true)>
         //mostrar resultados
         cout << to_string(miFila) + "|  " + to_string(miPareja) + "-" + to_string(nip) +  "|  " + to_string(resultado[miP
areja]) +"|  " + to_string(resultado[nip]) + '\n';
         //comunicar finalizacíon



   }
      <examen_fin[nip]=true;
      terminado++;>
}

//-----------------------------------------------------
void Profesor (int& silla, int& silla1, int& silla2, int pareja[N_EST], int fila, bool& hayFila) {
   for(int i=0; i<N_FIL; i++) {

      // esperar a que haya dos
      <await silla = 2
         //comunicar a cada uno su pareja, y la fila que les toca
         pareja[silla1]= silla2;
         pareja[silla2] = silla1;
         fila = i;
         hayFila = true;>
      <await (levantado == 2)
         silla = 0;
         hayFila = false;
         levantado=0>

   }
   // esperar que todos hayan terminado
   <await terminado == 60
   //fin examen
   >

}

void leerFich(int D[N_FIL][N_COL]){
   ifstream f("material_apoyo/datos.txt");
```

```
    if(f.is_open()){
        for(int i=0; i<N_FIL; i++){
            for(int j=0; j<N_COL; j++){
                f >> D[i][j];
            }
        }
        f.close();
    }else{
        cout << "No se ha podido abrir el fichero datos.txt" << endl;
    }
}


int main(){
    int D[N_FIL][N_COL]; //para almacenar los datos
    int fila = 0;  //cada pareja coger´a una
    int pareja[N_EST]; //pareja[i] ser´a la pareja asignada
    bool examen_fin[N_EST];
    for (int i = 0; i < N_EST; i++){
        examen_fin[i] = false; // inicializar vector
    }
    int terminado=0;
    int levantado = 0;
    int silla = 0;  // 0 = 0 sillas ocupadas , 1 = una silla ocupada, 2 = 2 sillas ocupadas
    int silla1; // su valor es el del estudiante sentado
    int silla2; // su valor es el del estudiante sentado
    int resultado[N_EST];
    bool hayFila=false;


    //cargar "datos.txt" en "D"
    leerFich(D);
    thread Estu[60];
    thread Profe;

    Profe= thread(&Profesor, ref(silla), ref(silla1), ref(silla2), ref(pareja), ref(fila), ref(hayFila), ref(rprimer),ref(rsegu
ndo),ref(rtercero),ref(rcuarto),ref(rquinto), ref(rsexto), ref(examen_fin), terminado,ref(testigo), ref(a), ref(b), ref(c), r
ef(d), ref(e), ref(f), ref(levantado));

    for(int i=0; i<N_EST; i++){
        Estu[i]= thread(&Estudiante, i, ref(fila), ref(hayFila), ref(D), ref(silla), ref(resultado), ref(examen_fin), ref(silla
1), ref(silla2), ref(pareja), ref(rprimer),ref(rsegundo),ref(rtercero),ref(rcuarto),ref(rquinto), ref(rsexto),ref(testigo) ,re
f(a), ref(b), ref(c), ref(d), ref(e), ref(f), ref(terminado), ref(levantado));
    }

    Profe.join();
    for (int i = 0; i < N_EST; i++){
        Estu[i].join();//me bloqueo hasta que "P[i]" termine
    }


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

}