Nombres: Javier Julve Yubero, Alejandro Benedí Andrés
NIPs: 840710, 843826

TRABAJO PREVIO PRÁCTICA 5

Explicación

El vector de enteros <pareja> cada posición representa un nip diferente. En cada una de las posiciones se encontrará la pareja de dicho nip

El vector de booleanos <conoce_su_nip> cada posición del vector representa a un nip distinto.Si esa posición esta en “true” significa que el representante si que conoce el nip del representado.Si la posicion se encuentra en false significa que no lo conoce aún.

En el vector de booleanos <tiene_pareja> cada posición del vector representa un nip diferente.Si la posicion esta en “true” significa que ese estudiante si que tiene una pareja establecida.Si la posicion esta en “false” significa que no tiene pareja asignada

La variable <representantes_terminados> representa el número de procesos representantes que han terminado.

El vector de booleanos <calculado> tiene 30 posiciones una por cada fila.Si esa posición está en “true” significa que el nip menor de la pareja ha terminado de calcular la fila.

Utilizamos un canal para pasar diferentes datos (fila, nipPareja, …) o para recibir resultado de las operaciones realizadas por los estudiantes (suma o máximo de las filas). Además usamos el sincronismo para poder sincronizar el servidor con el cliente

En el proceso main tan solo habrá que iniciar el proceso profesor e inicializar alguna variable

Código

```
integer N_EST = 60              // numero de estudiantes
integer N_FIL = 30              // numero de filas de la matriz
channel of string chan
```

Proceso profesor // servidor

```
        Socket chan(SERVER_PORT)        // crea el canal chan
        // Bind(  )
        // Listen( )

        integer pareja[N_EST]
        integer cliente
        integer sillas_ocupadas = 0;
        integer NIPS[2]

        for (fila, 0..N_FIL){

                for (sillas_ocupadas, 0..2){
                        cliente = chan.accept
                        thread(&representante, ref(chan), cliente,
ref(NIPS[sillas_ocupadas]),pareja,fila)
                }

                for (i,0..2){
                        <await conoce_su_nip[NIPS[i]]>
                }

                <pareja[NIPS[0]] = NIPS[1]
                pareja[NIPS[1]] = NIPS[0]
                tiene_pareja[NIPS[0]] = true
                tiene_pareja[NIPS[1]] = true>
        }
        <await represnetantes_terminados = 30>

        // join de todos los procesos representantes

        close()
```

```
Proceso representante(Socket chan, integer cliente, integer nip,  integer pareja[], integer fila)
// intermediario

        string mensaje

        chan(cliente) => mensaje
        trocea(mensaje,nip)     // divide el mensaje y guarda en <nip> el nip del representado

        conoce_su_nip[nip] = true

        <await tiene_pareja[nip]>
        mensaje = "<pareja[nip]>,<fila>"
        chan(cliente)<= mensaje                 // manda el mensaje con la pareja y la fila

        if(nipPareja > nip){

                chan(cliente)=> maximo[fila]            // vector de strings
                calculado[fila] = true
        }
        else{
                chan(cliente)=> suma
                <await calculado[fila]>         // en la funcion pasa de string a enteros
                mostrar resultado()
                < representantes_terminados ++ >
        }
        close(cliente)

proceso Estudiante(Matriz m, integer nip) // cliente

         cliente = chan.connect()

        string mensaje = "sentar,<nip>
        chan(cliente)<= mensaje       // manda mensaje

        chan(cliente) => mensaje      // recibe el mensaje con la fila y con la pareja
        trocea(mensaje,fila,nipPareja)          // trocea el mensaje y guarda la fila en <fila> y
el nip de la pareja en <nipPareja>

        if(nipPareja > nip){

                chan(cliente)<= maximoFila(fila)       // envía el resultado en string
        }
        else{
                chan(cliente)<= sumaFila(fila)          // envía el resultado en string
        }


        close()
```