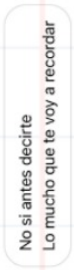
GestorDeImpresion.pdf

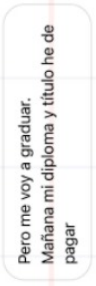
Anónimo

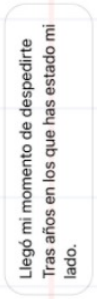
Análisis de Algoritmos y Estructuras de Datos

2º Grado en Ingeniería Informática

Escuela Superior de Ingeniería
Universidad de Cádiz







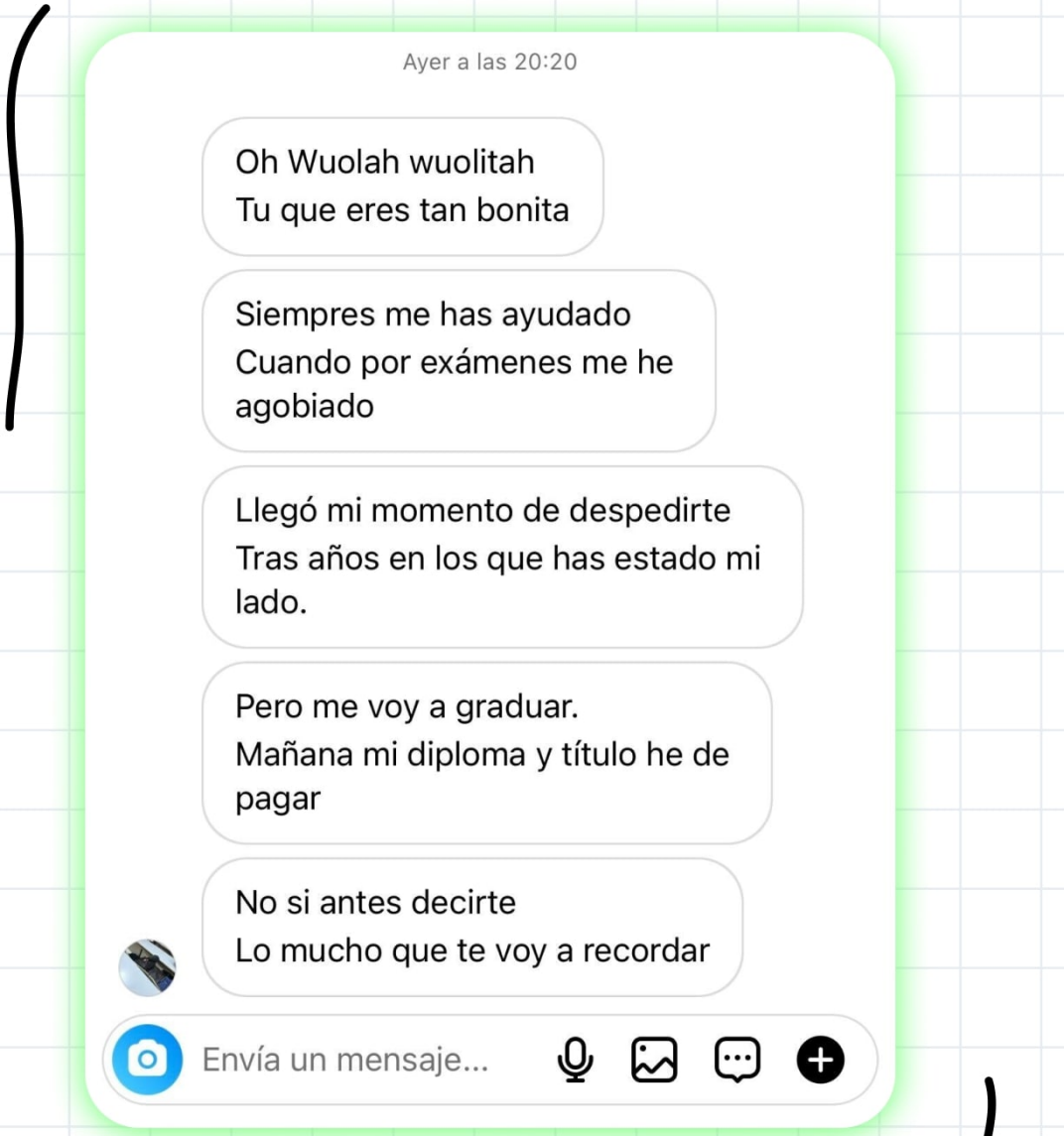
Gestor de impresión

```
class GestorDeImpresion{
public:
   // PostCondicion:crea un gestor de impresion para un numero determinado de usuarios
   GestorDeImpresion(int numeroUsuarios);
   // Precondición : el tipo de urgencia solo puede ser  U - urgente  o N - no urgente
   // PostCondicion: si el codigo esta añadido ya no hace nada, si el codigo del trabajo está
pero cambia la urgencia se le cambia la prioridad
   // si el usuario no tiene trabajo en la cola de impresión , se añade el trabajo en la cola
de impreseion y se le asigna un turno,
   // si el usuario ya está en la cola de impresion se añade un trabajo de un tipo de
urgencia determinada
   void añadirTrabajo(string user, bool esUrgente, string codTrabajo);
   // PostCondicion: elimina el trabajo que esta el primero en la cola de impresion, si la
cola de impresion está vacia no hace nada
   void eliminarTrabajo(); // el siguiente que se va a imprimir entiendo
   // PostCondición: cancela todos los trabajos, tanto urgentes como no , que el usuario
tiene en la cola de impresion, si el usuario no
   // tiene trabajos pendientes para imprimir en la cola de impresion no hace nada
   void cancelarTrabajos (string user);
   ~GestorDeImpresion();
private:
   typedef struct {
      string codigoUsuario;
      Cola<string> trabajoUrgente(), trabajoNoUrgente();
   } usuario;

   int numeroUsuarios_;
   Cola<usuario> turnoUsuario(numeroUsuarios_); //opcion 1

   Cola<usuario> turnoUsuario(0); //opcion 2
};

GestorDeImpresion::GestorDeImpresion(int numeroUsuarios){
   numeroUsuarios_ = numeroUsuarios;
   Cola<usuario> turnoUsuarioAux(numeroUsuarios_); // opcion 2
   turnoUsuario = turnoUsuarioAux; //opcion 2
}

void GestorDeImpresion::añadirTrabajo(string user, bool esUrgente, string codTrabajo){
```

```
   // PostCondicion: si el codigo esta añadido ya no hace nada, si el codigo del trabajo está
pero cambia la urgencia se le cambia la prioridad
   // si el usuario no tiene trabajo en la cola de impresión , se añade el trabajo en la cola
de impreseion y se le asigna un turno,
   // si el usuario ya está en la cola de impresion se añade un trabajo de un tipo de
urgencia determinada

   if (turnoUsuario.vacio()){

      usuario u;

      u.codigoUsuario = user;
      if (esUrgente) u.trabajoUrgente.push(codTrabajo);
      else u.trabajoNoUrgente.push(codTrabajo);

      turnoUsuario.push(u);
   }
   else {
      Cola<string> colaAuxiliar();

      while (!turnoUsuario.vacio() && turnoUsuario.frente().codigoUsuario != user){
         colaAuxiliar.push(turnoUsuario.frente());
         turnoUsuario.pop();
      }

      if(turnoUsuario.vacia()) {
         turnoUsuario = colaAuxiliar;

         usuario u;

         u.codigoUsuario = user;
         if (esUrgente) u.trabajoUrgente.push(codTrabajo);
         else u.trabajoNoUrgente.push(codTrabajo);

         turnoUsuario.push(u);
      }
      while(!colaAuxiliar.vacia()){

         Cola<string> auxTrabajosNoUrgentes, auxTrabajosUrgentes;
         bool trabajoEncontrado = false;

         while(!turnoUsuario.frente().trabajoNoUrgente.vacia() && !trabajoEncontrado){
            if(turnoUsuario.frente().trabajoNoUrgente.frente() == codTrabajo){
               trabajoEncontrado = true;
```

Que no te escriban poemas de amor cuando terminen la carrera 🚩🚩🚩🚩🚩🚩

(a nosotros por suerte nos pasa)



WUOLAH

```
            if (esUrgente){
               turnoUsuario.frente().trabajoNoUrgente.frente().pop();
               turnoUsuario.frente().trabajoUrgente.push(codTrabajo);
            }
         }
         else{

auxTrabajosNoUrgentes.push(turnoUsuario.frente().trabajoNoUrgente.frente());
            turnoUsuario.frente().trabajoNoUrgente.pop();
         }
      }

      turnoUsuario.frente().trabajoNoUrgente = auxTrabajosNoUrgentes;

      if(!trabajoEncontrado){
         while(!turnoUsuario.frente().trabajoUrgente.vacia() && !trabajoEncontrado){
            if(turnoUsuario.frente().trabajoUrgente.frente() == codTrabajo){
               trabajoEncontrado = true;
            }else{
               auxTrabajosUrgentes.push(turnoUsuario.frente().trabajoUrgente.frente());
               turnoUsuario.frente().trabajoUrgente.pop();
            }
         }

         turnoUsuario.frente().trabajoUrgente = auxTrabajosUrgentes;

         if(!trabajoEncontrado){
            turnoUsuario.frente().trabajoUrgente.push(codTrabajo);
         }
      }
   }
}

void GestorDeImpresion::eliminarTrabajo(){
   if ( !turnoUsuario.vacia()){
      if(! (turnoUsuario.frente().trabajoUrgente.vacia()))
turnoUsuario.frente().trabajoUrgente.pop();
      else if ( !turnoUsuario.frente().trabajoNoUrgente.vacia())
turnoUsuario.frente().trabajoNoUrgente.pop();

      if( turnoUsuario.frente().trabajoUrgente.vacia() &&
turnoUsuario.frente().trabajoNoUrgente.vacia()) turnoUsuario.pop();
      else {
```

```
        turnoUsuario.push(turnoUsuario.frente());
        turnoUsuario.pop();
      }
    }
}

void GestorDeImpresion::cancelarTrabajos (string user){

   if(!turnoUsuario.vacio()){ // si el usuario no está en cola de impresion no tiene trabajos
para imprimir por lo que no se hace nada
      Cola<usuario> auxiliar;

      while(!turnoUsuario.vacio() && turnoUsuario.frente().codigoUsuario == user) {
        auxiliar.push(turnoUsuario.frente());
        turnoUsuario.pop();
      }

      if (turnoUsuario.frente().codigoUsuario == user){

        turnoUsuario.frente().~trabajosUrgente();
        turnoUsuario.frente().~trabajosNoUrgente();
        turnoUsuario.pop();

        while (!turnoUsuario.vacio()){
          auxiliar.push(turnoUsuario.frente());
          turnoUsuario.pop();
        }

        turnoUsuario = auxiliar;
      }
    }
}

GestorDeImpresion::~GestorDeImpresion(){

   while(!turnoUsuario.vacia()){
      turnoUsuario.~codigoUsuario();
      turnoUsuario.frente().~trabajoUrgente();
      turnoUsuario.frente().~trabajoNoUrgente();
      turnoUsuario.pop();
   }

   ~turnoUsuario();
}
```