

mini-TPE - “La sala de reuniones”

Laboratorio Algoritmos y Estructura de Datos I



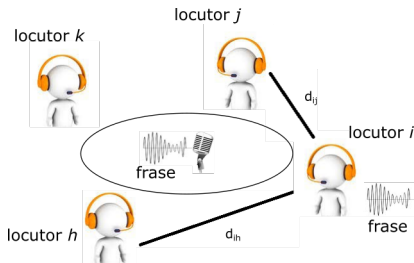
DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

2do Cuatrimestre 2019

Contexto

En la empresa AlgoSRL se realizan reuniones de trabajo periódicamente. Confeccionar una minuta de la reunión suele ser una tarea larga y aburrida. La empresa nos encargó idear un sistema que permita hacer la transcripción automáticamente a través de las voces grabadas. Cada persona presente posee un micrófono individual que va registrando cuando habla y guardándolo en su propio archivo de sonido.



Detalles Técnicos

Un micrófono, es un dispositivo electrónico que captura fluctuaciones de presión en el aire. Luego, estos datos se almacenan (de forma analógica o digital) para poder procesarlos.

Entendemos por:

- ▶ **señal analógica** es una señal que varía de forma continua a lo largo del tiempo.
- ▶ **señal digital** transforma la *señal*/analógica en una secuencia de valores discretos, tomados a intervalos regulares de tiempo.

Detalles Técnicos

Estas señales digitales serán almacenadas en forma de secuencia de números enteros (**AMPLITUDES**) que varían en un cierto umbral positivo y negativo dependiendo de la **PROFUNDIDAD** (medido en bits). La cantidad de bits, determina la cantidad de niveles (números posibles) que puede registrarse. Los valores posibles estarán definidos en el rango $[-2^{(P-1)}, 2^{(P-1)} - 1]$ donde P se refiere a la profundidad.

Ejemplos:

Profundidad: 8 bits

- ▶ **Niveles:** 256
- ▶ **Amplitud** ($P = 8$): $[-2^7, 2^7 - 1] = [-128, 127]$

Detalles Técnicos

Por otra parte, la cantidad de muestras por segundo que registramos se conoce como FRECUENCIA DE MUESTREO y se mide en Hertz (muestras por segundo). Comúnmente la frecuencia de muestreo varía entre 8.000 Hz (teléfonos por ejemplo) y hasta 96.000 Hz (calidad DVDs).

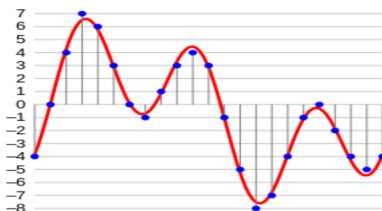


Figura: ejemplo señal de 4 bits

Detalles Técnicos

Por otra parte, la cantidad de muestras por segundo que registramos se conoce como FRECUENCIA DE MUESTREO y se mide en Hertz (muestras por segundo). Comúnmente la frecuencia de muestreo varía entre 8.000 Hz (teléfonos por ejemplo) y hasta 96.000 Hz (calidad DVDs).

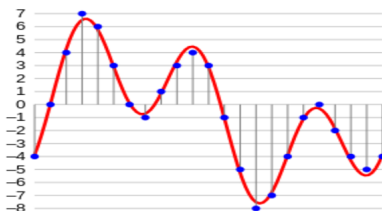


Figura: ejemplo señal de 4 bits

señal:

-4 0 4 7 6 3 0 -1 1 3 4 3 -1 -5 -8 -7 -4 -1 0 ...

Representación

La Sala de Reunión

El sistema de registro en la Sala de Reunión considera un número fijo de micrófonos, los cuales comienzan todos a grabar en forma simultánea, y luego se apagan en el mismo momento al finalizar. Cada micrófono registra los sonidos que serán almacenados en secuencias de números enteros.

Por lo tanto, contaremos con una lista de secuencias

$M : seq\langle seq\langle \mathbb{Z} \rangle \rangle$ en donde cada secuencia será la que contenga la información de cada uno de los micrófonos. Supondremos que cada micrófono sólo capta el sonido de la persona que lo utiliza y no el sonido de otras personas en la sala (caso skype con auriculares por ejemplo).

Representación

La Sala de Reunión

El sistema de registro en la Sala de Reunión considera un número fijo de micrófonos, los cuales comienzan todos a grabar en forma simultánea, y luego se apagan en el mismo momento al finalizar. Cada micrófono registra los sonidos que serán almacenados en secuencias de números enteros.

Por lo tanto, contaremos con una lista de secuencias

$M : seq\langle seq\langle \mathbb{Z} \rangle \rangle$ en donde cada secuencia será la que contenga la información de cada uno de los micrófonos. Supondremos que cada micrófono sólo capta el sonido de la persona que lo utiliza y no el sonido de otras personas en la sala (caso skype con auriculares por ejemplo).

type amplitud = \mathbb{Z}

type audio = $seq\langle amplitud \rangle$

type intervalo = $\langle \mathbb{Z}, \mathbb{Z} \rangle$

Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- ▶ Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:
 1. La frecuencia de muestreo es 44100 Hz
 2. La profundidad es 16 bits.
 3. Los números están en rango según la profundidad prof.
 4. La duración de la señal es mayor a 10 segundos.
 5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- ▶ Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:
 1. La frecuencia de muestreo es 44100 Hz
 2. La profundidad es 16 bits.
 3. Los números están en rango según la profundidad prof.
 4. La duración de la señal es mayor a 10 segundos.
 5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

```
proc esGrabacionValida (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , out result: Bool) {  
  Pre {}  
  Post {}  
}
```

Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- ▶ Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:
 1. La frecuencia de muestreo es 44100 Hz
 2. La profundidad es 16 bits.
 3. Los números están en rango según la profundidad prof.
 4. La duración de la señal es mayor a 10 segundos.
 5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

proc esGrabacionValida (in a: audio, in prof: \mathbb{Z} , in freq: \mathbb{Z} , out result: Bool) {
 Pre { True }

Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:

1. La frecuencia de muestreo es 44100 Hz
2. La profundidad es 16 bits.
3. Los números están en rango según la profundidad prof.
4. La duración de la señal es mayor a 10 segundos.
5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

```
proc esGrabacionValida (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , out result: Bool) {  
  Pre { True }  
  Post { result = true
```

Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:

1. La frecuencia de muestreo es 44100 Hz
2. La profundidad es 16 bits.
3. Los números están en rango según la profundidad prof.
4. La duración de la señal es mayor a 10 segundos.
5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

proc esGrabacionValida (in a: *audio*, in prof: \mathbb{Z} , in freq: \mathbb{Z} , out result: Bool) {
 Pre { *True* }
 Post { *result* = *true* \leftrightarrow }

Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- ▶ Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:
 1. La frecuencia de muestreo es 44100 Hz
 2. La profundidad es 16 bits.
 3. Los números están en rango según la profundidad prof.
 4. La duración de la señal es mayor a 10 segundos.
 5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

```
proc esGrabacionValida (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , out result: Bool) {  
  Pre { True }  
  Post { result = true  $\leftrightarrow$  audioValido(a, prof, freq) }  
}
```

Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:

1. La frecuencia de muestreo es 44100 Hz
2. La profundidad es 16 bits.
3. Los números están en rango según la profundidad prof.
4. La duración de la señal es mayor a 10 segundos.
5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

```
proc esGrabacionValida (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , out result: Bool) {  
  Pre { True }  
  Post { result = true  $\leftrightarrow$  audioValido(a, prof, freq) }  
}  
  
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  freqValida(freq) profValida(prof) enRango(a, prof)  
  micFunciona(a) duraMasDe(10, a)  
}
```


Ejercicios

esGrabaciónValida

proc esGrabaciónVálida(in a: *audio*, in prof : \mathbb{Z} , in freq: \mathbb{Z} , out result : Bool):

- ▶ Que dada una señal, una frecuencia de muestreo y una profundidad compruebe que:
 1. La frecuencia de muestreo es 44100 Hz
 2. Los números están en rango según la profundidad p.
 3. La profundidad es 16 bits.
 4. La duración de la señal es mayor a 10 segundos.
 5. El micrófono funciona: No existe una cadena de 0s de longitud mayor a 1 segundo.

```
proc esGrabacionValida (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , out result: Bool) {  
  Pre { True }  
  Post { result = true  $\leftrightarrow$  audioValido(a, prof, freq) }  
}  
  
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a, prof)  $\wedge$   
  micFunciona(a)  $\wedge$  duraMasDe(10, a)  
}
```

Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a,prof)  $\wedge$   
  duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}
```

Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
    (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a,prof)  $\wedge$   
    duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}  
pred freqValida(freq:  $\mathbb{Z}$ ) {  
    freq = 44100  
}  
pred profValida(prof:  $\mathbb{Z}$ ) {  
    prof = 16  
}
```

Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a,prof)  $\wedge$   
  duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}  
pred freqValida(freq:  $\mathbb{Z}$ ) {  
  freq = 44100  
}  
pred profValida(prof:  $\mathbb{Z}$ ) {  
  prof = 16  
}  
pred enRango(a: audio, prof:  $\mathbb{Z}$ ) {  
  ( $\forall x : \mathbb{Z}$ ) ( $x \in a \longrightarrow -2^{(prof-1)} \leq x \leq 2^{(prof-1)} - 1$ )  
}
```

Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a,prof)  $\wedge$   
  duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}  
pred freqValida(freq:  $\mathbb{Z}$ ) {  
  freq = 44100  
}  
pred profValida(prof:  $\mathbb{Z}$ ) {  
  prof = 16  
}  
pred enRango(a: audio, prof:  $\mathbb{Z}$ ) {  
  ( $\forall x : \mathbb{Z}$ ) ( $x \in a \longrightarrow -2^{(prof-1)} \leq x \leq 2^{(prof-1)} - 1$ )  
}  
pred duraMasDe(t: tiempo, a: audio) {  
  duracion(a) > t  
}
```

Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a,prof)  $\wedge$   
  duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}  
pred freqValida(freq:  $\mathbb{Z}$ ) {  
  freq = 44100  
}  
pred profValida(prof:  $\mathbb{Z}$ ) {  
  prof = 16  
}  
pred enRango(a: audio, prof:  $\mathbb{Z}$ ) {  
  ( $\forall x : \mathbb{Z}$ ) ( $x \in a \longrightarrow -2^{(prof-1)} \leq x \leq 2^{(prof-1)} - 1$ )  
}  
pred duraMasDe(t: tiempo, a: audio) {  
  duracion(a) > t  
}  
aux duracion (a : audio) : tiempo = enSegundos(|a|);
```

Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a,prof)  $\wedge$   
  duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}  
pred freqValida(freq:  $\mathbb{Z}$ ) {  
  freq = 44100  
}  
pred profValida(prof:  $\mathbb{Z}$ ) {  
  prof = 16  
}  
pred enRango(a: audio, prof:  $\mathbb{Z}$ ) {  
   $(\forall x : \mathbb{Z}) (x \in a \longrightarrow -2^{(prof-1)} \leq x \leq 2^{(prof-1)} - 1)$   
}  
pred duraMasDe(t: tiempo, a: audio) {  
  duracion(a) > t  
}  
aux duracion (a : audio) : tiempo = enSegundos(|a|);  
aux enSegundos (n :  $\mathbb{Z}$ ) : tiempo = n/44100,0;
```

Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a,prof)  $\wedge$   
  duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}  
pred freqValida(freq:  $\mathbb{Z}$ ) {  
  freq = 44100  
}  
pred profValida(prof:  $\mathbb{Z}$ ) {  
  prof = 16  
}  
pred enRango(a: audio, prof:  $\mathbb{Z}$ ) {  
  ( $\forall x : \mathbb{Z}$ ) ( $x \in a \longrightarrow -2^{(prof-1)} \leq x \leq 2^{(prof-1)} - 1$ )  
}  
pred duraMasDe(t: tiempo, a: audio) {  
  duracion(a) > t  
}  
aux duracion (a : audio) : tiempo = enSegundos(|a|);  
aux enSegundos (n :  $\mathbb{Z}$ ) : tiempo = n/44100,0;  
pred micFunciona(a: audio) {  
   $\neg(\exists i, j : \mathbb{Z})(0 \leq i < j < |a| \wedge_L$   
    duraMasDe(1, subseq(a, i, j))  $\wedge$  sonTodosCeros(subseq(a, i, j)))  
}
```


Ejercicios

esGrabaciónValida

```
pred audioValido(a: audio, prof:  $\mathbb{Z}$ , freq:  $\mathbb{Z}$ ) {  
  (freqValida(freq)  $\wedge$  profValida(prof))  $\wedge_L$  enRango(a, prof)  $\wedge$   
  duraMasDe(10, a)  $\wedge$  micFunciona(a)  
}  
pred freqValida(freq:  $\mathbb{Z}$ ) {  
  freq = 44100  
}  
pred profValida(prof:  $\mathbb{Z}$ ) {  
  prof = 16  
}  
pred enRango(a: audio, prof:  $\mathbb{Z}$ ) {  
  ( $\forall x : \mathbb{Z}$ ) ( $x \in a \longrightarrow -2^{(prof-1)} \leq x \leq 2^{(prof-1)} - 1$ )  
}  
pred duraMasDe(t: tiempo, a: audio) {  
  duracion(a) > t  
}  
aux duracion (a : audio) : tiempo = enSegundos(|a|);  
aux enSegundos (n :  $\mathbb{Z}$ ) : tiempo = n/44100,0;  
pred micFunciona(a: audio) {  
   $\neg(\exists i, j : \mathbb{Z})(0 \leq i < j < |a| \wedge_L$   
    duraMasDe(1, subseq(a, i, j))  $\wedge$  sonTodosCeros(subseq(a, i, j)))  
}  
pred sonTodosCeros(a: audio) {  
  ( $\forall i : \mathbb{Z}$ ) ( $0 \leq i < |a| \longrightarrow_L a[i] = 0$ )  
}
```

Ejercicios

proc reacomodar(inout **m**: *seq* \langle *audio* \rangle , in **freq**: \mathbb{Z} , in **prof**: \mathbb{Z}): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia *m*, reordenar *m* según la potencia de cada audio (de menor a mayor).

```
proc reacomodar (inout m: seq<audio>, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ ) {  
  Pre { $|m| > 0 \wedge_L \text{esMatriz}(m) \wedge (\forall a : \mathbb{Z}) (0 \leq a < |m| \longrightarrow_L$   
     $\text{audioValido}(m[a], \text{prof}, \text{freq})) \wedge m0 = m$ }  
  Post { $|m0| = |m| \wedge \text{mismosAudios}(m0, m) \wedge \text{ordenadaXPotencia}(m)$ }  
}
```

Ejercicios

proc reacomodar(inout **m**: $\text{seq}\langle\text{audio}\rangle$, in **freq**: \mathbb{Z} , in **prof**: \mathbb{Z}): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia m , reordenar m según la potencia de cada audio (de menor a mayor).

```
proc reacomodar (inout m:  $\text{seq}\langle\text{audio}\rangle$ , in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ ) {  
  Pre { $|m| > 0 \wedge_L \text{esMatriz}(m) \wedge (\forall a : \mathbb{Z}) (0 \leq a < |m| \longrightarrow_L$   
     $\text{audioValido}(m[a], \text{prof}, \text{freq})) \wedge m0 = m$ }  
  Post { $|m0| = |m| \wedge \text{mismosAudios}(m0, m) \wedge \text{ordenadaXPotencia}(m)$ }  
} pred esMatriz(m:  $\text{seq}\langle\text{audio}\rangle$ ) {  
  ( $\forall a : \mathbb{Z}) 0 \leq a < |m| \longrightarrow_L |m[a]| = |m[0]|$ )  
}
```

Ejercicios

proc reacomodar(inout *m*: *seq*⟨*audio*⟩, in *freq*: \mathbb{Z} , in *prof*: \mathbb{Z}): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia *m*, reordenar *m* según la potencia de cada audio (de menor a mayor).

```
proc reacomodar (inout m: seq⟨audio⟩, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ ) {  
  Pre { $|m| > 0 \wedge_L \text{esMatriz}(m) \wedge (\forall a : \mathbb{Z}) (0 \leq a < |m| \longrightarrow_L$   
     $\text{audioValido}(m[a], \text{prof}, \text{freq})) \wedge m0 = m$ }  
  Post { $|m0| = |m| \wedge \text{mismosAudios}(m0, m) \wedge \text{ordenadaXPotencia}(m)$ }  
} pred esMatriz(m: seq⟨audio⟩) {  
  ( $\forall a : \mathbb{Z}) 0 \leq a < |m| \longrightarrow_L |m[a]| = |m[0]|$ )  
}  
pred mismosAudios(m1: seq⟨audio⟩, m2: seq⟨audio⟩) {  
  estaContenida(m1, m2)  $\wedge$  estaContenida(m2, m1)  
}
```

Ejercicios

proc reacomodar(inout *m*: *seq*⟨*audio*⟩, in *freq*: \mathbb{Z} , in *prof*: \mathbb{Z}): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia *m*, reordenar *m* según la potencia de cada audio (de menor a mayor).

```
proc reacomodar (inout m: seq⟨audio⟩, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ ) {  
  Pre { $|m| > 0 \wedge_L \text{esMatriz}(m) \wedge (\forall a : \mathbb{Z}) (0 \leq a < |m| \longrightarrow_L$   
     $\text{audioValido}(m[a], \text{prof}, \text{freq})) \wedge m0 = m\}$   
  Post { $|m0| = |m| \wedge \text{mismosAudios}(m0, m) \wedge \text{ordenadaXPotencia}(m)\}$   
} pred esMatriz(m: seq⟨audio⟩) {  
  ( $\forall a : \mathbb{Z}) 0 \leq a < |m| \longrightarrow_L |m[a]| = |m[0]|$ )  
}  
pred mismosAudios(m1: seq⟨audio⟩, m2: seq⟨audio⟩) {  
  estaContenida(m1, m2)  $\wedge$  estaContenida(m2, m1)  
}  
pred estaContenida(m1: seq⟨audio⟩, m2: seq⟨audio⟩) {  
  ( $\forall a : \text{audio})(a \in m1 \longrightarrow a \in m2)$ )  
}
```

Ejercicios

proc reacomodar(inout m : $seq\langle audio \rangle$, in $freq$: \mathbb{Z} , in $prof$: \mathbb{Z}): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia m , reordenar m según la potencia de cada audio (de menor a mayor).

```
proc reacomodar (inout  $m$ :  $seq\langle audio \rangle$ , in  $prof$ :  $\mathbb{Z}$ , in  $freq$ :  $\mathbb{Z}$ ) {  
  Pre { $|m| > 0 \wedge_L esMatriz(m) \wedge (\forall a : \mathbb{Z}) (0 \leq a < |m| \longrightarrow_L$   
     $audioValido(m[a], prof, freq)) \wedge m0 = m$ }  
  Post { $|m0| = |m| \wedge mismosAudios(m0, m) \wedge ordenadaXPotencia(m)$ }  
} pred esMatriz( $m$ :  $seq\langle audio \rangle$ ) {  
  ( $\forall a : \mathbb{Z}) 0 \leq a < |m| \longrightarrow_L |m[a]| = |m[0]|$ )  
}  
pred mismosAudios( $m1$ :  $seq\langle audio \rangle$ ,  $m2$ :  $seq\langle audio \rangle$ ) {  
   $estaContenida(m1, m2) \wedge estaContenida(m2, m1)$   
}  
pred estaContenida( $m1$ :  $seq\langle audio \rangle$ ,  $m2$ :  $seq\langle audio \rangle$ ) {  
  ( $\forall a : audio)(a \in m1 \longrightarrow a \in m2)$ )  
}  
pred ordenadaXPotencia( $m$ :  $seq\langle audio \rangle$ ) {  
  ( $\forall i : \mathbb{Z})(0 \leq i < |m| - 1 \longrightarrow_L potencia(m[i]) \leq potencia(m[i + 1]))$ )  
}
```

Ejercicios

proc reacomodar(inout **m**: *seq*⟨*audio*⟩, in **freq**: \mathbb{Z} , in **prof**: \mathbb{Z}): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia *m*, reordenar *m* según la potencia de cada audio (de menor a mayor).

```
proc reacomodar (inout m: seq⟨audio⟩, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ ) {  
  Pre { $|m| > 0 \wedge_L \text{esMatriz}(m) \wedge (\forall a : \mathbb{Z}) (0 \leq a < |m| \longrightarrow_L$   
     $\text{audioValido}(m[a], \text{prof}, \text{freq})) \wedge m0 = m$ }  
  Post { $|m0| = |m| \wedge \text{mismosAudios}(m0, m) \wedge \text{ordenadaXPotencia}(m)$ }  
  { pred esMatriz(m: seq⟨audio⟩) {  
    ( $\forall a : \mathbb{Z}) 0 \leq a < |m| \longrightarrow_L |m[a]| = |m[0]|$ )  
  }  
  pred mismosAudios(m1: seq⟨audio⟩, m2: seq⟨audio⟩) {  
     $\text{estaContenida}(m1, m2) \wedge \text{estaContenida}(m2, m1)$   
  }  
  pred estaContenida(m1: seq⟨audio⟩, m2: seq⟨audio⟩) {  
    ( $\forall a : \text{audio})(a \in m1 \longrightarrow a \in m2)$ )  
  }  
  pred ordenadaXPotencia(m: seq⟨audio⟩) {  
    ( $\forall i : \mathbb{Z})(0 \leq i < |m| - 1 \longrightarrow_L \text{potencia}(m[i]) \leq \text{potencia}(m[i + 1]))$ )  
  }  
}
```

aux potencia (a : audio) : $\mathbb{R} = \frac{\sum_{i=0}^{|a|-1} a[i]^2}{|a|}$;

Ejercicios

proc silencios(in a: audio, in prof: \mathbb{Z} , in freq: \mathbb{Z} , in umbral: amplitud, out listaSilencios: seq<intervalo>): Que dado un audio, determina todos los intervalos de silencios. Un silencio se define por una secuencia del audio en el que el valor absoluto de la señal no supera un umbral por al menos 1 segundos. El intervalo es una tupla conteniendo la posición del inicio (p_i) y final (p_f) del intervalo de silencio de la señal. Es importante aclarar que antes del inicio y después del final de los silencios encontrados, no puede seguir habiendo silencio.

```
proc silencios (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , in umbral: amplitud, out
listaSilencios: seq<intervalo>) {
  Pre {audioValido(a, prof, freq)}
  Post {intervalosValidos(a, umbral, listaSilencios)  $\wedge_L$ 
        sonTodosLosSilenciosDelAudio(a, umbral, listaSilencios)}
}
```


Ejercicios

proc silencios(in a: audio, in prof: \mathbb{Z} , in freq: \mathbb{Z} , in umbral: amplitud, out listaSilencios: seq(intervalo)): Que dado un audio, determina todos los intervalos de silencios. Un silencio se define por una secuencia del audio en el que el valor absoluto de la señal no supera un umbral por al menos 1 segundos. El intervalo es una tupla conteniendo la posición del inicio (p_i) y final (p_f) del intervalo de silencio de la señal. Es importante aclarar que antes del inicio y después del final de los silencios encontrados, no puede seguir habiendo silencio.

```
proc silencios (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , in umbral: amplitud, out
listaSilencios: seq(intervalo)) {
  Pre {audioValido(a, prof, freq)}
  Post {intervalosValidos(a, umbral, listaSilencios)  $\wedge_L$ 
        sonTodosLosSilenciosDelAudio(a, umbral, listaSilencios)}
}

pred intervalosValidos(a: audio, umbral, silencios: seq(intervalo)) {
  ( $\forall p : \text{intervalo}$ )  $p \in \text{silencios} \longrightarrow_L$ 
  (interEnRango(a, p)  $\wedge_L$  (duraMasDe(1, subseq(a, p0, p1 + 1)))
}
```

Ejercicios

proc silencios(in a: audio, in prof: \mathbb{Z} , in freq: \mathbb{Z} , in umbral: amplitud, out listaSilencios: seq(intervalo)): Que dado un audio, determina todos los intervalos de silencios. Un silencio se define por una secuencia del audio en el que el valor absoluto de la señal no supera un umbral por al menos 1 segundos. El intervalo es una tupla conteniendo la posición del inicio (p_i) y final (p_f) del intervalo de silencio de la señal. Es importante aclarar que antes del inicio y después del final de los silencios encontrados, no puede seguir habiendo silencio.

```
proc silencios (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , in umbral: amplitud, out
listaSilencios: seq(intervalo)) {
  Pre {audioValido(a, prof, freq)}
  Post {intervalosValidos(a, umbral, listaSilencios)  $\wedge_L$ 
        sonTodosLosSilenciosDelAudio(a, umbral, listaSilencios)}
}

pred intervalosValidos(a: audio, umbral, silencios: seq(intervalo)) {
  ( $\forall p : \text{intervalo}$ )  $p \in \text{silencios} \longrightarrow_L$ 
  (interEnRango(a, p)  $\wedge_L$  (duraMasDe(1, subseq(a, p_0, p_1 + 1)))
}

pred interEnRango(a: audio, p: intervalo) {
   $0 \leq p_0 < p_1 < |a|$ 
}
```

Ejercicios

```
proc silencios (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , in umbral: amplitud, out  
listaSilencios: seq<intervalo>) {  
  Pre {audioValido(a, prof, freq)}  
  Post {intervalosValidos(a, listaSilencios)  $\wedge_L$   
        sonTodosLosSilenciosDelAudio(a, umbral, listaSilencios)}  
}
```

Ejercicios

```
proc silencios (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , in umbral: amplitud, out
listaSilencios: seq<intervalo>) {
  Pre {audioValido(a, prof, freq)}
  Post {intervalosValidos(a, listaSilencios)  $\wedge_L$ 
        sonTodosLosSilenciosDelAudio(a, umbral, listaSilencios)}
}

pred sonTodosLosSilenciosDelAudio(a: audio, umbral: amplitud,
listaSilencios: seq<intervalo>) {
  ( $\forall p: intervalo$ ) esSilencioDeMaximaLongitud(a, p, umbral)  $\rightarrow_L$ 
   $p \in listaSilencios \wedge \#apariciones(silencios, p) = 1$  }

pred esSilencioDeMaximaLongitud(a: audio, p: intervalo, umbral:  $\mathbb{Z}$ ) {
  esSilencio(a, p, umbral)  $\wedge \neg(\exists q: intervalo) (0 \leq q_0 \leq p_0 \wedge$ 
   $p_1 \leq q_1 < |a| \wedge (p_1 - p_0) < (q_1 - q_0)) \wedge_L esSilencio(a, q, umbral)$  }
```

Ejercicios

```
proc silencios (in a: audio, in prof:  $\mathbb{Z}$ , in freq:  $\mathbb{Z}$ , in umbral: amplitud, out
listaSilencios: seq<intervalo>) {
  Pre {audioValido(a, prof, freq)}
  Post {intervalosValidos(a, listaSilencios)  $\wedge_L$ 
        sonTodosLosSilenciosDelAudio(a, umbral, listaSilencios)}
}

pred sonTodosLosSilenciosDelAudio(a: audio, umbral: amplitud,
listaSilencios: seq<intervalo>) {
  ( $\forall p: intervalo$ ) esSilencioDeMaximaLongitud(a, p, umbral)  $\longrightarrow_L$ 
   $p \in listaSilencios \wedge \#apariciones(silencios, p) = 1$  }

pred esSilencioDeMaximaLongitud(a: audio, p: intervalo, umbral:  $\mathbb{Z}$ ) {
  esSilencio(a, p, umbral)  $\wedge \neg(\exists q: intervalo) (0 \leq q_0 \leq p_0 \wedge$ 
   $p_1 \leq q_1 < |a| \wedge (p_1 - p_0) < (q_1 - q_0)) \wedge_L esSilencio(a, q, umbral)$  }

pred esSilencio(a: audio, p: intervalo, umbral:  $\mathbb{Z}$ ) {
  ( $\forall i: \mathbb{Z}$ ) ( $p_0 \leq i \leq p_1 \longrightarrow_L |a[i]| < umbral$ )
}
```

Intro a \LaTeX

Vamos a aprender como escribir la especificación en formato \LaTeX .

Un poco de motivación

$$\forall y \in \{1, 2, 3\} : \left(\frac{\frac{3+2}{(5*7)+1}}{2} \right) \times y \leq 150/2$$

Unos símbolos raros, paréntesis de distinto tamaño, cosas a distinta altura. ¿Cuánto se tarda en escribirlo en un editor como Word?

Un poco de motivación

$$\forall y \in \{1, 2, 3\} : \left(\frac{\frac{3+2}{(5*7)+1}}{2} \right) \times y \leq 150/2$$

Unos símbolos raros, paréntesis de distinto tamaño, cosas a distinta altura. ¿Cuánto se tarda en escribirlo en un editor como Word?

¿ Y un auxiliar en especificación ?

$\text{pred } \textit{incluida}(s1 : \textit{seq}(\mathbb{Z}), s2 : \textit{seq}(\mathbb{Z})) \{ (\forall x : \mathbb{Z}) (x \in s1 \longrightarrow x \in s2) \}$

Otro poco de motivación

Texto plano (Notepad)	A Mano	Word, LibreOffice, etc	\LaTeX
4	6	8	10

Nota máxima por formato de entrega

Otro poco de motivación



Corrigió texto plano



Corrigió \LaTeX

WYSIWYG, WYSIWYM & WTF

- ▶ **WYSIWYG: What You See Is What You Get**
En este paradigma escribiremos directamente lo que queremos obtener. Por ejemplo, cuando queremos agregar una imagen, directamente la insertamos donde queremos ponerla.

Ejemplo: Word

- ▶ **WYSIWYM: What You See Is What You Mean** En este paradigma escribiremos texto y comandos que luego de ser compilados se convierten en nuestro archivo final. Por ejemplo, cuando queremos agregar una imagen, escribimos algo como `áca quiero poner una imagen'con la ruta a este archivo.`

Ejemplo: L^AT_EX

Cómo usarlo

De manera local:

1. Instalamos el *compilador*

- ▶ En Linux: depende de la distribución (`sudo apt-get install texlive-full` en Ubuntu, por ejemplo)
- ▶ En Windows: Texlive o MikTeX <http://miktex.org/>
- ▶ En Mac: MacTex (`brew cask install mactex`)

2. Editamos el documento y lo ejecutamos. Dos opciones:

- ▶ Usamos un entorno de desarrollo (IDE)
 - ▶ TeXmaker
 - ▶ TeXStudio
- ▶ Editor de texto + consola
 - ▶ `pdflatex <archivo.tex>`

Cómo usarlo

Online:

- ▶ Overleaf (link de invitación <https://www.overleaf.com>).

En esta es la opción recomendada, ya que:

- ▶ están presentes la mayoría de los paquetes de LATEX.
- ▶ es posible compartir la edición con otros usuarios, en forma simultánea.
- ▶ permite sincronizar con git o dropbox.

Para comenzar a utilizarlo es preciso generar un usuario.

Estructura de un archivo L^AT_EX

Para nuestro primer texto:

- ▶ Abrimos nuestro editor de texto o IDE.
- ▶ Comenzamos el documento indicando qué tipo de texto estamos escribiendo. Vamos a usar:

```
1 \documentclass{article}
```

- ▶ A continuación ponemos los paquetes que vamos a usar:

```
1 \usepackage{...}
```

- ▶ Empezamos el **contenido** de nuestro documento:

```
1 \begin{document}
```

- ▶ Escribimos el texto que queremos
- ▶ Terminamos nuestro documento con:

```
1 \end{document}
```

- ▶ Compilamos para que se genere un pdf con nuestro texto

Ejemplo 1 - Hola Mundo

Empezamos por algo básico:

```
1  \documentclass{article}
2  \begin{document}
3      Hola Mundo
4  \end{document}
```

El resultado ...

Hola Mundo

Ejemplo 2

Escribimos un documento con diferentes secciones:

```
1 \documentclass{article}
2 % Esto es un comentario
3 \usepackage[utf8]{inputenc}
4 \begin{document}
5     \section{Título principal}
6         \subsection{Título secundario}
7             Texto de párrafo
8 \end{document}
```

1 Título principal

1.1 Título secundario

Texto de párrafo

¿Y cómo hacemos los símbolos raros?

Símbolos raros:

- ▶ Para símbolos simples: <http://detexify.kirelabs.org/>
- ▶ Para cosas más complicadas:

Ej: ¿Cómo hago una sumatoria con el índice abajo y la cota arriba?)

Google: "*latex sum limits below above*"

Modo matemático

- ▶ Hay ciertos símbolos que solamente se pueden escribir en modo matemático. Por ejemplo: \forall , \exists , \mathbb{Z} .
- ▶ En esos casos tenemos que escribir el comando entre símbolos $\$$.
- ▶ Por ejemplo:
 - $\$ \backslash \text{forall} \$$ es \forall
 - $\$ \backslash \text{exists} \$$ es \exists
 - $\$ \backslash \text{mathds}\{Z\} \$$ es \mathbb{Z}

¿Y los ejercicios de TP?

¿Y los ejercicios de TP?

Macros!

- ▶ Para facilitarles la transición a \LaTeX , se va a subir a la página de la materia un template con el esqueleto de la resolución del TP.
- ▶ Además va a haber otro archivo con las definiciones de los comandos para las construcciones más comunes: definiciones de problemas, pre, post, funciones auxiliares y varias cosas más.

Un par de ejemplos

¿Qué podemos escribir?

- ▶ Problemas: $\backslash\text{begin}\{\text{proc}\}\{\text{nombreProc}\}\{\text{parám}\}\{\}$
 - ▶ $\backslash\text{pre}\{\text{fórmula}\}$
 - ▶ $\backslash\text{post}\{\text{fórmula}\}$
- ▶ Símbolos
 - ▶ $\backslash\text{ent } \textit{genera } \mathbb{Z}$
 - ▶ $\backslash\text{float } \textit{genera } \mathbb{R}$
 - ▶ $\backslash\text{implicaLuego } \textit{genera } \longrightarrow_L$
 - ▶ $\backslash\text{YLuego } \textit{genera } \wedge_L$
 - ▶ $\backslash\text{IfThenElse}\{a > 0\}\{1\}\{0\} \textit{ genera if } a > 0 \text{ then } 1 \text{ else } 0 \text{ fi}$
 - ▶ $\backslash\text{TLista}\{\mathbb{Z}\} \textit{ genera seq}\langle\mathbb{Z}\rangle$

Y más...

Un par de ejemplos

Un problema entero: Dados dos números devolver el resto de su división.

```
1 \begin{proc}{cociente}{in a,b: \ent, out result: \ent}{}
2   \pre{b \neq 0}
3   \post{result = a \ mod \ b}
4 \end{proc}
```

Se muestra así:

```
proc cociente (in a,b:  $\mathbb{Z}$ , out result:  $\mathbb{Z}$ ) {
  Pre { $b \neq 0$ }
  Post { $result = a \bmod b$ }
}
```

Un par de ejemplos

Un auxiliar:

```
1  \pred{noRepe}{l:\TLista{\ent}}{(\forall i:\ent)
2    (0 \leq i < l \implies l[i] \notin subseq(l,0,i))}
```

Se muestra así:

```
pred noRepe (l:seq<Z>) {
  (\forall i : Z)(0 \leq i < l \implies l[i] \notin subseq(l,0,i))
}
```

Un par de ejemplos

Agregar una foto (en su forma más básica) es también bastante simple. Con el siguiente código se incluye una imagen, con su tamaño reducido al 40 %.

```
1 \includegraphics[scale=0.4]{img/success.jpg}
```



Resumiendo

- ▶ Hay \LaTeX para su S.O.
- ▶ Usamos \LaTeX , para que la escritura del TP se les simplifique mucho.
- ▶ Vimos lo mínimo. Si algo no sale, busquen por Internet y/o pregunten.

Entregable del mini-TPE

Dentro de la carpeta **template-alumnos** de este taller y del enunciado del TPE encontramos los siguientes archivos:

- ▶ **Algo1Macros.tex**
- ▶ **caratula.sty**: Macro para generar la caratula del TP.
- ▶ **logo_dc.jpg**
- ▶ **logo_uba.jpg**
- ▶ **solucion.tex**: Archivo fuente del entregable en latex.

Entregable del mini-TPE

Estas son las dos páginas que se generarían al compilar la versión del TPE.



Mini-TP de Especificación

Sala de Reuniones

06 de Abril de 2019

Algoritmos y Estructuras de Datos I

Grupo 8

Integrantes	LU	Cursos de profesores
Eloni, Juan	007	bombardit.co.uk

1. Problemas

```
proc entradaInicializada (in w: seq(X), out Z: bool, out result: bool) {  
  Pre {true}  
  Post {true}  
  aux subAuxiliar (I: X) : bool == true;  
}
```

2. Predicados y Auxiliares generales

3. Decisiones tomadas



Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón 1) Pabellón 1A-2A
Instituto de Matemática - CIEMAT
Ciudad Autónoma de Buenos Aires - San Agustín
Tel/Fax: (+54 11) 4543-1111 / 4543-1112
<http://www.uba.ar>

Entregable del mini-TPE

Estas son las dos páginas que se generarían al compilar la versión del TPE.



Mini-TP de Especificación

Sala de Reuniones

08 de Abril de 2019

Algoritmos y Estructuras de Datos I

Grupo 8

Integrantes	LU	Cursos de profesores
Ikoni, Juan	001	bond@ut.ox.ac

1. Problemas

```
proc entradaInicializada (in w: seq(X), pref X, suc X, out result: bool) {  
  Pre {true}  
  Post {true}  
  aux subAuxiliar (l: X) : bool == true;  
}
```

2. Predicados y Auxiliares generales

3. Decisiones tomadas



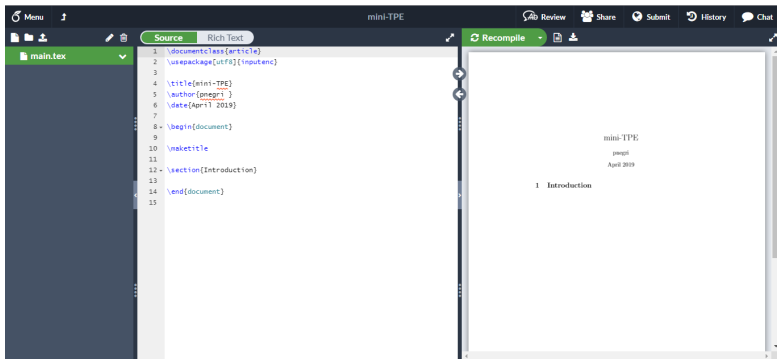
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón IV) Pabellón IV
Instituto de Matemática - C1430EHA
Ciudad Autónoma de Buenos Aires - San Agustín
Tel/Fax: (+54 11) 4948-1111 / 4948-1112
<http://www.ingenieria.uba.ar>

VAMOS A COMPILAR!!!

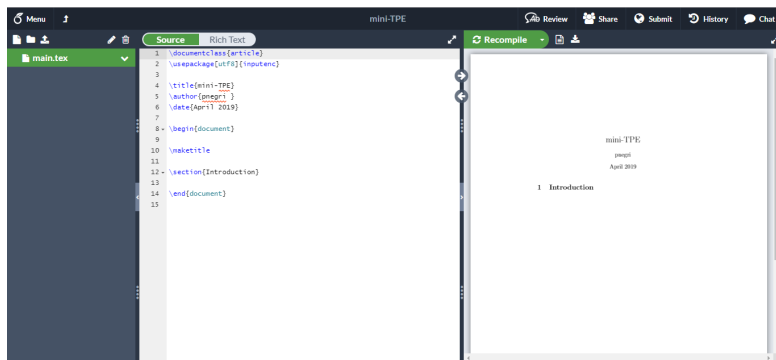
Vamos a usar la opción de compilar on-line nuestro proyecto:

- ▶ Precisamos crear una cuenta en Overleaf (link de invitación <https://www.overleaf.com?r=32909244&rm=d&rs=b>). Las cuentas son gratuitas.
- ▶ Una vez que accedemos a nuestra pantalla de Inicio, creamos un nuevo proyecto con el botón NEW PROJECT, ubicado en la esquina superior izquierda.
- ▶ Del menú desplegado, seleccionar Blank Project
- ▶ En la ventana siguiente le damos un nombre al proyecto. Por ejemplo **mini-TPE**. Clickear en Create.

Compilación L^AT_EX



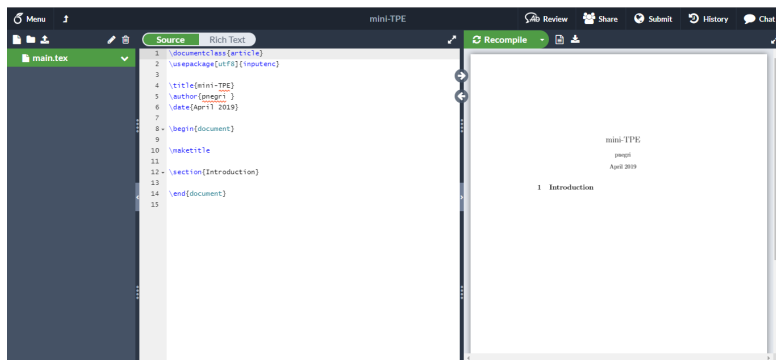
Compilación L^AT_EX



GENIAL!!

Tenemos generado un archivo pdf a partir de un código L^AT_EX...

Compilación L^AT_EX




GENIAL!!

Tenemos generado un archivo pdf a partir de un código L^AT_EX...
Pero esto no es el mini-TPE...

Compilación L^AT_EX

Vamos a agregar al proyecto los archivos para compilar la solución del mini-TPE.

- ▶ Del panel del proyecto, usamos el botoncito .
- ▶ Subimos al box los archivos del proyecto: **Algo1Macros.tex**, **caratula.sty**, **logo_dc.jpg**, **logo_uba.jpg**, **solucion.tex**.

Compilación \LaTeX

Add Files

- New File
- Upload
- From Another Project
- From External URL
- From Mendeley
- From Zotero

Drag here

or

Select from your computer

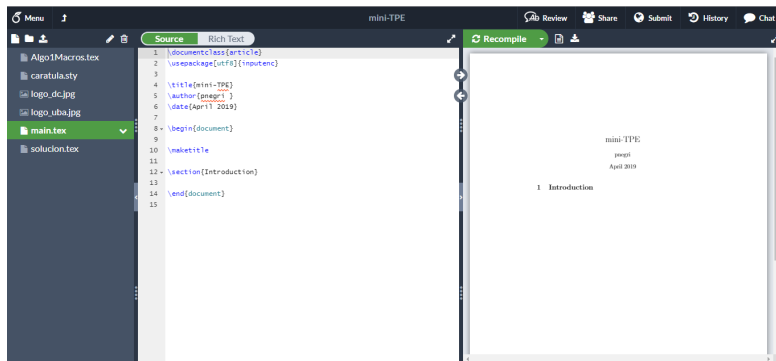
logo_dc.jpg 128.0kB Cancel

logo_uba.jpg 229.1kB Cancel Processing...

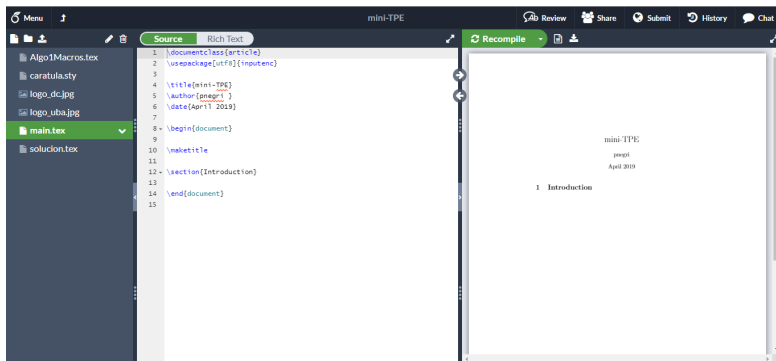
solucion.tex Cancel

Algo1Macros.tex Cancel

Compilación L^AT_EX

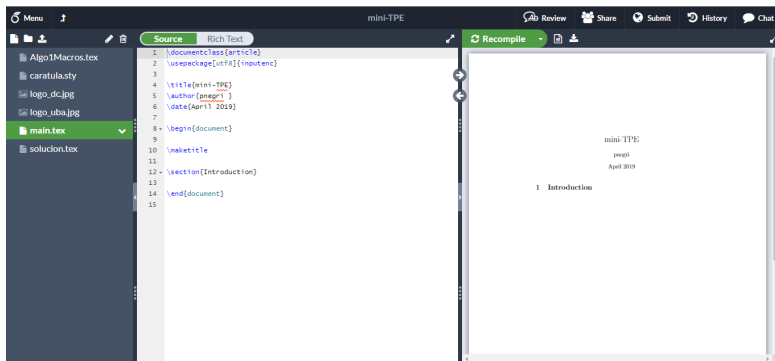


Compilación L^AT_EX



BUENISIMO!!

Compilación L^AT_EX



BUENISIMO!!

Pero el pdf no cambia, aunque le diga que **Recompile...**

Compilación L^AT_EX

- ▶ En el proyecto del mini-TPE, el archivo principal es el `solucion.tex`, no el `main.tex`, que fue creado por default por overleaf.
- ▶ Para realizar el cambio del archivo de compilación, cliqueamos el botón Menu, en la esquina superior izquierda.
- ▶ En el menu desplegado, en el ítem Main document, cambiamos de `main.tex` a `solucion.tex`.
- ▶ Salimos del menu y eliminamos el archivo `main.tex` del proyecto. Para ello al seleccionarlo, aparece la opción de hacerle Delete.
- ▶ Ahora compilamos el proyecto con el botón Recompile del panel derecho.

Compilación L^AT_EX

The image shows the mini-TPE LaTeX editor interface. On the left is a file explorer with a list of files: `Algo1Macros.tex`, `caratulasty`, `logo_dc.jpg`, `logo_uba.jpg`, and `solucion.tex` (selected). The main editor displays the source code for `solucion.tex`, which is a LaTeX document class for a paper. The code includes package loading, document structure commands, and a list of participants. On the right, the compiled PDF document is shown. The header of the PDF includes the logo of the 'DEPARTAMENTO DE COMPUTACION' and the title 'Mini-TP de Especificación'. The document content includes a title page with the title 'Mini-TP de Especificación', a subtitle 'Sala de Reuniones', a date '08 de Abril de 2019', and a group name 'Grupo 8'. Below this is a table with two columns: 'Integrante' and 'Correo electrónico'. The table lists four members: Bond, James; Apellido, Nombre2; Apellido, Nombre3; and Apellido, Nombre4. The bottom of the PDF shows a watermark of the 'UNIVERSIDAD DE BUENOS AIRES' logo.

```
1 \documentclass[10pt]{article}
2
3 \setlength{\parskip}{0.1em}
4 \newcommand{\tab}{~ \quad}
5 \input{Algo1Macros}
6 \usepackage{caratula} % Version modificada para usar las macros de
7 % algo de -> https://github.com/bcardiff/dc-tex
8 \begin{document}
9
10 \title{Mini-TP de Especificaci\on}
11 \subtitle[Sala de Reuniones]
12 \fecha[08 de Abril] de 2019]
13 \materia[Algoritmos y Estructuras de Datos I]
14 \grupo[Grupo 8]
15 \newcommand{\seal}{\textit{seal}}
16
17 % Pongan cuantos integrantes quieran
18 \integrante[Bond, James]{007}{bond@m16.co.uk}
19 \integrante[Apellido, Nombre2]{002/01}{email2@dominio.com}
20 \integrante[Apellido, Nombre3]{003/01}{email3@dominio.com}
21 \integrante[Apellido, Nombre4]{004/01}{email4@dominio.com}
22
23 \maketitle
24
25 \section{Problemas}
26 \begin{proc}[esGrabacionValida]{In a: \TLista\ent, prof: \ent,
27   \frac{\ent}{\Out result: \bool}}{}
```

DEPARTAMENTO DE COMPUTACION
FACULTAD DE CIENCIAS EXACTAS Y NATURALES

Mini-TP de Especificación

Sala de Reuniones

08 de Abril de 2019

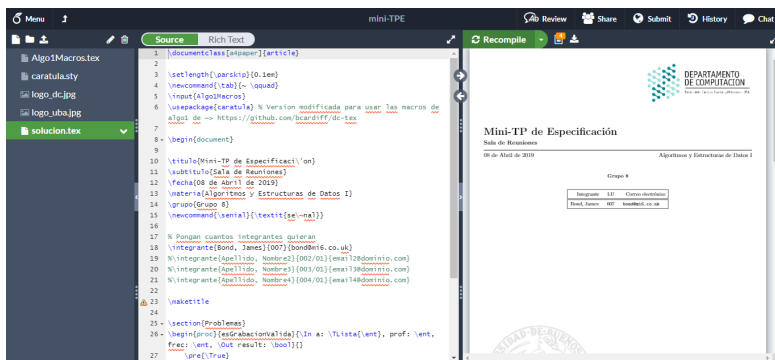
Algoritmos y Estructuras de Datos I

Grupo 8

Integrante	LI	Correo electrónico
Bond, James	007	bond@m16.co.uk

UNIVERSIDAD DE BUENOS AIRES

Compilación L^AT_EX



The screenshot displays the mini-TPE web interface, which is used for compiling LaTeX documents. The interface is divided into three main sections: a file explorer on the left, a source code editor in the center, and a preview window on the right.

File Explorer (Left): Shows a list of files including `Algo1Macros.tex`, `caratulasty`, `logo_dc.jpg`, `logo_uba.jpg`, and `solucion.tex` (which is selected).

Source Code Editor (Center): Displays the LaTeX source code for `solucion.tex`. The code includes a document class, preamble with packages like `caratula`, and a main body with a title, subtitle, date, and a list of participants. The code is as follows:

```
1 \documentclass[12pt]{article}
2
3 \setlength{\parskip}{0.1em}
4 \newcommand{\tab}{~\quad}
5 \input{Algo1Macros}
6 \usepackage{caratula} % Version modificada para usar las macros de
7 % algo de -> https://github.com/bcardiff/dc-tex
8 \begin{document}
9
10 \title{Mini-TP de Especificaci\on}
11 \subtitle[Sala de Reuniones]
12 \fecha[08 de Abril de 2019]
13 \materia[Algoritmos y Estructuras de Datos I]
14 \grupo[Grupo 8]
15 \newcommand{\seal}{\textit{seal}}
16
17 % Pongan cuantos integrantes quieran
18 \integrante[Bond, James]{007}{bond@m16.co.uk}
19 %integrante[Apellido, Nombre2]{002/01}{email2@dominio.com}
20 %integrante[Apellido, Nombre3]{003/01}{email3@dominio.com}
21 %integrante[Apellido, Nombre4]{004/01}{email4@dominio.com}
22
23 \maketitle
24
25 \section{Problemas}
26 \begin{proc}[esGrabacionValida]{In a: \Tlista\ent, prof: \ent,
27   \frac{\ent, \Out result: \bool}{}
28   \pre{True}
```

Preview Window (Right): Shows the rendered output of the LaTeX document. The title is "Mini-TP de Especificación" and the subtitle is "Sala de Reuniones". The date is "08 de Abril de 2019". The subject is "Algoritmos y Estructuras de Datos I". The group is "Grupo 8". The participants are listed in a table:


Integrante	LI	Cuerpo distribuido
Bond, James	007	bond@m16.co.uk

AHORA SIII!

Compilación L^AT_EX



AHORA SIII!

Para bajar el archivo pdf creado, hago un click en el botón . El archivo lo guarda con el mismo nombre del proyecto.

Ejercicios

En el laboratorio, vamos a resolver y pasar al documento \LaTeX los siguientes problemas:

proc elQueMasHablaba(in m: seq<audio>, in freq: \mathbb{Z} , in prof : \mathbb{Z} , out res: audio): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia m , devolver el audio que corresponde a la persona que mas tiempo hablo en la reunion. Utilizar la definición de silencios del ejercicio precedente.

proc sePusoAlgidaLaDiscusion(in m: seq<audio>, in freq: \mathbb{Z} , in prof : \mathbb{Z} , out res: intervalo): Dadas todas las grabaciones válidas realizadas en la sala y almacenadas en la secuencia m , devolver el intervalo, de más de 3 segundos, donde hay más de 2 personas hablando al mismo tiempo.