INF152 Estructuras Discretas
Profesores: M. Bugueño – S. Gallardo
Universidad Técnica Federico Santa María
Departamento de Informática – Octubre 15, 2021.

Ayudantes
valentina.arostica@sansano.usm.cl
bryan.gonzalezr@usm.cl

Pseudocódigo

1. ¿Qué es el pseudocódigo?

Es un lenguaje creado especialmente para la realización de algoritmos; la característica principal de éste es que se pensó para el entendimiento del humano y no el de la máquina. Por ello es que se considera un lenguaje sencillo. Como todos los algoritmos se deben ejecutar en una máquina, es necesario traducir el pseudocódigo a un lenguaje de programación, siendo considerado un borrador, por esto es utilizado en textos donde no está definido un lenguaje de programación en particular, haciendo de él uno universal.

El pseudocódigo es considerado un lenguaje de **alto nivel** (se caracteriza por ser más entendible por el humano que los de un nivel inferior) y posee una estructura secuencial.

2. Características

El pseudocódigo comparte muchas características con lenguajes de programación formales, entre las que podemos encontrar:

- **Variables:** Son espacios de memoria que pueden cambiar su contenido a lo largo de la ejecución de su pseudocódigo, y se accede a través del identificador (nombre)
  Constantes: No cambia en el transcurso de la ejecución
- **Entradas y salidas(INPUT/OUTPUT):** Las entradas son los datos que el usuario ingresa para procesar en el programa. Y las salidas son las respuestas que el programa entrega al usuario.
- **Instrucciones de control:** Principalmente se utilizan 2: If y for (o foreach), que vienen en el paquete latex que detallaremos más adelante (\usepackage{algorithm} y \usepackage{algorithmic})

3. Paquete *"algorithm"* y *"algorithmic"*

LaTeX es una muy buena herramienta para plasmar pseudocódigo en un pdf, pues existe el paquete *algorithm*, el cual facilita mucho la creacion de pseudocódigo, además de tener un diseño muy estético.

3.1. Introduccion a alogorithm

Lo primero que deben hacer para trabajar con el paquete, es incluirlo en su código LaTeX, a través de \usepackage{algorithm} y \usepackage{algorithmic}. Ambos paquetes son complementarios

Para comenzar un algoritmo con este paquete, lo primero que debemos hacer es crear el ambiente en el que trabajaremos. Esto se puede ver a mas detalle en el siguiente ejemplo (para ver el código, por favor revise el archivo .tex):

Algorithm 1 Algoritmo de prueba

```
1: procedure test
2: variable ← 1
```

En el ejemplo podemos apreciar como asignar un valor a una variable, algo que es muy útil para resolver problemas en pseudocódigo. Para cada linea de codigo (que no sea alguna instruccion de control), utilizaremos \STATE. Antes de definir nuestra variable, por ejemplo.

También habrán notado que comenzamos con un **procedure**. Para efectos de este curso, siempre debemos comenzar con **procedure**, pues es a través de esta sintaxis que definimos el nombre del algoritmo que estamos trabajando, y eventualmente podríamos llamarlo dentro de otro.

3.2. Condicionales

Para utilizar condicionales en el pseudocódigo latex, el paquete viene con un comando incluido. Si queremos utilizarlo, en vez de \STATE, debemos utilizar \IF, luego escribir los condicionales, y en las lineas siguientes los argumentos del condicional.
Para terminar el condicional, se **DEBE** colocar un \ENDIF

A continuacion, se mostrará un ejemplo:

Algorithm 2 Algoritmo de prueba condicional

```
1: procedure condicional
2: variable ← 1
3: if variable == 1 then
4:     variable2 ← 2
5: end if
```

Adicionalmente, los condicionales también pueden hacer uso de un else if (como el "elif"de python), y pueden ser anidados, como se puede ver en el siguiente ejemplo:

Algorithm 3 Condicional compuesto

```
1: procedure condicionalcompuesto
2: variable ← 1
3: if variable == 1 then
4:     variable2 ← 2
5:     if variable2 == 2 then
6:         variable3 ← 5
7:     end if
8: else if variable == 3 then
9:     variable2 ← 3
10: end if
```

3.3. Ciclos

Como ya deben conocer de otros lenguajes de programación, los ciclos principalmente son los for (o foreach en algunos lenguajes), y los while.
En LaTeX, utilizando las librerías anteriormente vistas, podemos escribir los ciclos a través de los comandos \FOR y \WHILE para representar cada tipo de ciclo. Cabe recalcar que al igual que los condicionales, hace falta cerrar el ciclo.

Veamos un ejemplo para cada uno:

Algorithm 4 Ciclo For

```
1: procedure usoFor
2: palabra ← "Paralelepipedo"
3: for letra in palabra do
4:     print letra
5: end for
```

Algorithm 5 Ciclo While

```
1: procedure usoWhile
2: palabra ← "Paralelepipedo"
3: contador ← 0
4: while contador ≤ len(palabra) do
5:     print palabra[contador]
6:     contador ← contador+1
7: end while
```

Ps. Notese que en este caso, los algoritmos quedaron uno al lado del otro, esto es posible gracias al ambiente minipage, los invitamos a investigarlo.