

- 0.1. La definición
- 0.2. Variables y constantes
- 0.3. Procedimientos y algoritmos
- 0.4. Operaciones y relaciones
- 0.5. Resumen
- 0.6. Bibliografía

UNIDAD

0

NOCIONES BÁSICAS

0. NOCIONES BÁSICAS

Antes de iniciar nuestro estudio de la Matemática Discreta, repasaremos algunos temas ya conocidos desde la escuela y estableceremos algunos conceptos mínimos comunes a todas las ciencias, en particular de extenso uso en matemática e informática.

0.1. La definición

Considere la siguiente frase sencilla en castellano:

El niño vio al marinero corriendo con su largavista.

Suponiendo que sabemos qué es un niño, qué es un marinero, qué significa correr y qué es un largavista, ¿qué quiere decir esta frase?:



Fig. 0.1: Niño mirando

- a) El niño con su largavista vio a un marinero que iba corriendo.
- b) El niño vio al marinero que corría, el cual llevaba un largavista.
- c) El niño que iba corriendo vio con un largavista, a un marinero.
- d) El niño que iba corriendo vio a un marinero que llevaba consigo un largavista.

Nuestro lenguaje natural, cotidiano o usual es ambiguo; la forma en que armamos oraciones no es siempre la correcta para lo que queremos en realidad decir, hay palabras que tienen más de un sentido, una misma oración puede tener distintos significados dependiendo del contexto, de quién la interprete e inclusive, en el lenguaje hablado, del tono en el que se pronuncia. Esto **no** puede pasar en matemática.

Durante nuestro estudio de la Matemática Discreta, iremos construyendo un lenguaje compuesto por términos técnicos y símbolos especiales que denotan conceptos que deben tener significados inequívocos. **Todos** debemos entender exactamente lo mismo cuando hablemos de *número natural*, $+$, *conjunto*, \emptyset , *grafo* o *expresión algebraica*.

La explicación de conceptos, de términos técnicos y de los símbolos especiales que los denotan se hará naturalmente en castellano, apelando a palabras comunes del lenguaje cotidiano que todos conocemos y sobre las que se supone no existe duda de su significado; además se utilizarán términos técnicos y símbolos ya establecidos con

anterioridad.

Definición: Llamamos **DEFINICIÓN** a una explicación clara y sin ambigüedades de un concepto, usando palabras comunes y términos técnicos cuyo significado se supone ya conocido.

Para despejar dudas, usualmente ejemplos del concepto y de su denotación siguiendo a la definición, servirán de clarificación y refuerzo. Así, una definición es nada más que *un acuerdo* sobre un concepto, para que todos entendamos lo mismo cuando lo nombremos.

Sin embargo, siempre habrá términos que no podremos explicar tan inequívocamente como quisiéramos. En toda disciplina hay unos pocos *términos primitivos* que solo admiten una **DEFINICIÓN INTUITIVA**, tal vez no tan precisa, pero que brinda informalmente una idea clara del concepto y que se refuerza por las propiedades que esos términos primitivos deben satisfacer. Como ejemplo, todos hemos tenido algún contacto con la geometría en la escuela, y todos sabemos qué es un *punto* ¿verdad?; bien, trate usted de dar una buena definición de punto!

Las definiciones que se utilizan en el presente texto tienen en general, según cómo se formulen, dos formatos distintos:

- a) **DEFINICIÓN DIRECTA:** explicación de un concepto utilizando términos técnicos y comunes ya conocidos.

*Un **tren** es una locomotora con uno o más vagones enganchados a ella en secuencia.*

- b) **DEFINICIÓN RECURSIVA:** explicación de un concepto utilizando términos técnicos y comunes ya conocidos y *iel mismo concepto que se está definiendo!*

*Un **tren** es una locomotora con un vagón enganchado a ella o es un **tren** con un vagón enganchado a él.*

Debe notarse que en las definiciones recursivas siempre habrá al menos dos partes: una parte directa donde se darán ejemplos mínimos del concepto (**CASO BASE**) utilizando términos conocidos, y una parte recursiva (**CASO RECURSIVO**) en la cual se utilizará el concepto que se está definiendo, para generalizar la idea a partir de los ejemplos dados.

De alguna forma, puede pensarse que una definición directa es **descriptiva** en el sentido que describe con palabras el concepto que se está definiendo; por su lado, una definición recursiva puede verse como **constructiva**, ya que presenta reglas que permiten generar ejemplos del concepto definido partiendo de los casos base, o verificar si un ejemplo puede realmente construirse usando esas reglas a partir de ellos.

Pero ... ¿por qué existen las definiciones recursivas?

En algunas ramas de la matemática y sobre todo en las ciencias de la computación, hay conceptos que se definen precisamente con una sencilla definición recursiva; para lograr la misma precisión, una definición directa suele insumir mucho tiempo y esfuerzo.

Ejemplo: Consideremos el familiar concepto de *expresión aritmética*. ¿Cómo lo definiremos?

- a) Definición intuitiva: una expresión aritmética es una combinación de números y variables numéricas, utilizando los símbolos de las operaciones básicas de adición, substracción, multiplicación y división, y eventualmente paréntesis.

Esta definición es sencilla pero no es precisa; supone que el lector conoce lo que se quiere decir con "combinación" y cómo se usan los paréntesis. Por ejemplo " $x + 1 + (/ 4^+ - ($ " cumple con la definición; es una combinación de números, variables, signos de operación y paréntesis, pero claramente *no es una expresión aritmética válida!*

- b) Definición directa: una expresión aritmética es una combinación de números y variables numéricas, utilizando los símbolos de las operaciones básicas de adición, substracción, multiplicación y división, y eventualmente paréntesis. En la combinación debe tenerse en cuenta que la expresión puede iniciar con un paréntesis abierto "(", con un número o con una variable, pero no con un signo de operación o con un paréntesis cerrado ")" salvo que el primer número sea negativo en cuyo caso se admite que la expresión inicie con un signo menos. Además, no pueden colocarse dos objetos del mismo tipo seguidos salvo que sean paréntesis del mismo tipo. Tampoco pueden ponerse seguidos un número y una variable, siempre entre ellos debe ir un signo de operación salvo que a un número o variable siga un paréntesis cerrado, en cuyo caso el signo de operación se coloca luego del paréntesis y antes de la siguiente variable o número o paréntesis abierto. Debe tenerse en cuenta también que, de usar paréntesis, siempre debe iniciarse con un paréntesis abierto y debe haber tantos paréntesis abiertos como cerrados, pero los mismos deben estar apareados ... etc., etc., etc.

En realidad, no queda muy claro qué es una expresión aritmética y aún falta mucho por escribir para que no deje lugar a equivocaciones.

- c) Definición recursiva: una expresión aritmética es:
- i) Un número.
 - ii) Una variable.
 - iii) Si **E** y **F** son expresiones aritméticas, entonces también lo son:
 $(E), E + F, E - F, E \times F, E / F$.
 - iv) Sólo son expresiones aritméticas las construidas con i), ii) e iii).

En este caso, no hay lugar a equivocaciones; cualquier expresión aritmética puede ser verificada (determinar si es válida) con estas reglas. Los dos primeros ítems de la definición son las expresiones aritméticas más simples posibles, ejemplos que conforman el *caso base*; el tercer ítem es el *caso recursivo* en el cual se está usando el término *expresión aritmética* que se está queriendo definir. El cuarto ítem, común en las definiciones recursivas, da un cierre a la definición para que no se pueda agregar nada más al concepto definido. ■

Otra ventaja de las definiciones recursivas está en la algorítmica. Sobre algunas estructuras, como los árboles que luego serán tratados en este texto, pueden establecerse distintos procedimientos para obtener resultados desde ellos o determinar algunas propiedades; estos procedimientos resultan, en algunos casos, sumamente sencillos de diseñar sobre la definición recursiva del concepto, pero extremadamente complicados de desarrollar desde la definición directa.

EJERCICIOS Y PREGUNTAS DE REPASO

- Indique todos los posibles significados de las siguientes frases en castellano:
 - Catalina no quiere a su hermana porque es muy envidiosa.
 - Nos veremos en el banco.
 - Viajaré a Mendoza solo una semana.
 - Diálogo: u) Pedro está practicando natación, o) ¿y cómo le va? u) Nada mal.
- Busque en Internet y elija la definición (intuitiva, directa o recursiva) que Usted mejor crea que describe cada concepto nombrado, o ensaye alguna propia:
 - Conjunto
 - Punto
 - Recta
 - Número
 - Colección
 - Verdad
- Encuentre y especifique al menos tres conceptos que tengan tanto una definición directa clara, como una definición recursiva también clara.
- Teniendo en cuenta la definición recursiva de *expresión aritmética* propuesta en el ejemplo anterior, desarrolle una explicación de por qué:
 - $x \cdot 1 + (/ 4^+ - ($ no es una expresión aritmética.
 - $1 + (x / 4)$ si es una expresión aritmética.
- En muchos temas de física, química, matemática o computación es necesario organizar ciertos objetos o datos en un arreglo rectangular (cargas en una placa conductora, redes cristalinas, matrices, grafos, mapas autoorganizados en redes neuronales, el juego de la Vida de Conway, etc.). Para su tratamiento suele ser importante definir “una vecindad” de un objeto O_{ij} cualquiera de la colección; la vecindad puede o no incluir el objeto en cuestión, puede tener los vecinos en cruz, todos los de alrededor más cercanos, o los que estén a una distancia de hasta 2, 3, ..., k objetos. Desarrolle una definición directa de vecindad que le parezca adecuada (indague en Internet qué se dice sobre este tema).

O_{11}	O_{12}	...	O_{1n}
O_{21}	O_{22}	...	O_{2n}
...
O_{m1}	O_{m2}		O_{mn}

0.2. Variables y constantes

Durante el tratamiento de cualquier tema, se consideran objetos individuales que tienen un significado o valor definido, el cual no cambia durante la discusión; de estos objetos se dice que son *constantes*.

Definición: Un objeto se dice que es **CONSTANTE** si no altera su valor o significado en el contexto del tratamiento del tema donde se lo está utilizando.

Claros ejemplos en matemáticas son los números **1**, **2**, **3** o el cociente constante existente entre la longitud de la circunferencia y su diámetro: el número designado usualmente por el símbolo π (letra griega pi). No solemos pensar en eso, pero los símbolos de operaciones como + para adición, – para sustracción, \times para multiplicación y / para división, por ejemplo, también son constantes ya que no cambian de significado

durante la discusión de un tema.

A veces, incluso queremos referirnos a *cualquier constante* sin determinar una específica y en ese caso suelen utilizarse las primeras letras del alfabeto **a**, **b**, **c**, ... para denotarlas.

Ejemplo: Cuando se quiere especificar el formato general de la ecuación de segundo grado se utiliza la fórmula $ax^2 + bx + c = 0$, en la cual se yuxtaponen dos letras como **bx** para indicar la operación de multiplicación entre ellas como es usual; las letras **a**, **b** y **c** designan constantes cualesquiera y para este ejemplo se pide solo que la constante **a** no sea nula, esto es, distinta de cero. Debe observarse que el ejemplo es *un formato* y no una ecuación; para obtener una ecuación específica de segundo grado, lo que el formato indica es que esas letras **a**, **b** y **c** deben ser reemplazadas por números específicos. ▮

La **x** en el anterior formato es algo bien distinto. Es usual contar con una colección de objetos y necesitar referirnos a uno de ellos con un nombre general, pero no a uno específico sino a cualquiera de ellos ya que no está determinado; decimos que es un objeto *indeterminado* de esa colección.

Definición: Un objeto se dice que es **VARIABLE** si es un objeto indeterminado de una colección, por lo cual su valor o significado puede variar en el contexto del tratamiento del tema donde se lo está utilizando.

En estos casos suelen utilizarse las últimas letras del alfabeto **x**, **y**, **z**, ... para denotarlo, y decimos de ellas que son *variables*.

Inclusive nos podríamos querer referir a una operación cualquiera (adición, sustracción, o cualquier otra) sin especificarla, y en ese caso podríamos inventar un símbolo como ***** para denotar a esa operación indeterminada. En este caso ***** funciona como una variable sobre la colección de operaciones de que se trate.

Ejemplo: Sean **x** y **y** dos objetos cualesquiera y ***** una operación definida entre ellos que permite obtener un nuevo objeto a partir de **x** y **y**. Se dice que la operación ***** es *conmutativa* si se verifica que $x * y$ tiene el mismo resultado que $y * x$. ▮

Una variable **x** se suele usar en matemática con al menos con dos significados distintos, que tienen que ver con el contexto en el cual se presenta. Por ejemplo, en una *función* definida sobre los números reales como $y = x^2$, la colección de objetos son *los números reales* y **x** designa a uno cualquiera de ellos, queriendo significar que, para cada reemplazo de la variable **x** por un número real específico, la función $y = f(x)$ tomará el valor de ese número multiplicado por sí mismo. Aquí se dice que **x** es la *variable independiente*, mientras **y** recibe el nombre de *variable dependiente*, ya que su valor depende del valor de **x**.

Por otro lado, en una *ecuación* como $x + 1 = 2$, la variable **x** recibe el nombre de *incógnita* ya que, si bien representa a cualquier número real, solo un valor de la colección, el cual suele querer determinarse, satisface la igualdad que la ecuación indica.

Finalmente, debe cuidarse no confundir lo que se entiende por *variable* en matemática y en los lenguajes de programación de computadoras. En un lenguaje de

programación, una variable es un nombre para referirse a una localización de memoria donde se almacena un valor de algún tipo de datos previsto por el lenguaje; la denominación *variable* está justificada, porque el valor en esa localización puede ser modificado por el programa durante su ejecución. Aquellas localidades de memoria a las que se les asignó un valor, pero no tienen un nombre para referirse a ellas, son las llamadas *constantes* del programa.

Ejemplo: Suponiendo se trabaja con valores de tipo entero en algún lenguaje de programación, una instrucción como $x = x + 1$ llamada *instrucción de asignación*, es perfectamente válida ya que indica que *debe sumarse la constante 1 al contenido de la locación de memoria referenciada por x y el resultado obtenido debe ser almacenado en la misma locación*, reemplazando su anterior valor. En la matemática de números enteros, la ecuación $x = x + 1$ no tiene sentido: *no existe ningún número entero que al sumarle uno dé como resultado el mismo número*. ¡Una instrucción de asignación en programación **no** es una ecuación matemática! █

EJERCICIOS Y PREGUNTAS DE REPASO

- Identifique las constantes y las variables en las siguientes expresiones aritméticas:
 - $2 + 3 = 5$
 - $x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$
 - $\text{Superficie} = \pi \cdot \text{radio}^2$
 - $P_n(x) = a_0x^0 + a_1x^1 + \dots + a_nx^n$
- Asigne un nombre adecuado para cada una de las expresiones del ejercicio anterior.
- Escriba una definición directa de *operación asociativa* e identifique en ella las variables y constantes que contenga.
- Hay lenguajes de programación que permiten asignar un nombre a una constante como en "*const PI = 3.141592*". Indague si su lenguaje de programación preferido tiene esta posibilidad y discuta brevemente sus hallazgos.

0.3. Procedimientos y algoritmos

Hay determinadas tareas que pueden efectuarse con una sola acción, de forma directa, y son tan sencillas y comunes que cualquiera puede realizarlas. *Cerrar los ojos* es una orden que todos sabemos cumplir.

Aún una tarea tan sencilla como *cerrar los ojos*, tiene sus condicionantes:

- Quien recibe la orden de efectuar la tarea debe entenderla; quien no hable castellano no cumplirá la orden (aunque seguramente sabe hacerla) porque no la comprende.
- Quien recibió y entendió la orden, debe estar capacitado para ejecutarla; podría suceder que tenga algún impedimento físico en sus párpados y no pueda

ejecutarla.

- Quien recibió, entendió la orden y está capacitada para hacerla, debe querer hacerla.

Entonces, si comprende la tarea encomendada, está capacitada para hacerla y está de acuerdo con hacerla, quien recibe una orden de este tipo podrá ejecutarla.

Hay otras tareas que son más complejas, por lo que requieren una cantidad de acciones para llevarlas a cabo. Hacer una deliciosa torta de coco o cambiar el neumático de un automóvil, son tareas que requieren una secuencia de acciones simples para completarse, ejecutadas en cierto orden. Quien tenga que desarrollar la tarea, deberá por supuesto comprender ahora cada acción simple, estar capacitado para hacerla y debe querer hacerla.

Definición: Un **PROCEDIMIENTO** es una secuencia finita de acciones simples prescriptas en un cierto orden, necesarias para completar una tarea.

Llamaremos a cada acción simple de un procedimiento, *una instrucción* y a quien sea que la ejecute el *procesador*, que siempre supondremos que la entiende, está capacitado para ejecutarla y quiere hacerlo.

Una receta de cocina es un procedimiento; suele haber en una receta, instrucciones como *agregue una pizca de sal* que no son muy precisas, por lo que su ejecución y el resultado obtenido puede ser distinto según el procesador que la lleve a cabo. Además, una tarea podría no tener un final establecido, como la de *calcular todas las cifras decimales de π* que, por ser un número irracional y trascendente, sabemos tiene infinitas cifras.

En informática interesan aquellos procedimientos bien definidos, donde cada acción está precisamente explicitada (y en general tiene una sola forma de ejecutarse) y que además siempre terminan arribando a un resultado en un tiempo finito.

Definición: Un **ALGORITMO** es una secuencia finita de instrucciones precisas dadas en un cierto orden necesarias para realizar una tarea, que siempre termina obteniendo un resultado en un tiempo finito.

En matemáticas también se suele llamar a esto un *método de resolución*, si la tarea a emprender es resolver un determinado problema matemático. Puede pensarse en un algoritmo como una *función* que transforma sus argumentos (entradas) en un resultado (su salida).

Dado un problema a resolver (la tarea), se diseña un algoritmo como una secuencia finita de instrucciones que, tomando los datos necesarios suministrados por el problema (*la entrada*), determina la solución al mismo, el resultado (*la salida*), en un número finito de pasos.

La especificación de un algoritmo puede hacerse de muchas formas:

- En lenguaje natural, esto es utilizando el lenguaje cotidiano explicando las instrucciones que lo conforman en la secuencia adecuada.

- Usando algún lenguaje de programación de computadoras (código).
- Utilizando algún lenguaje informal inventado a tal efecto (pseudocódigo).
- Mediante esquemas gráficos que especifiquen el orden en que las instrucciones deben ser ejecutadas (diagramas de flujo de control).

y de muchas otras (diagramas de Chapin, de Warnier, de Jackson, máquinas abstractas de estado finito, máquinas de Turing y otras)¹.

Uno de los algoritmos más antiguos que se conoce, es el explicitado por Euclides (siglo III AC) en sus *Elementos*² para describir el método más eficiente que se conoce para calcular el máximo común divisor; *ipor supuesto Euclides no pensaba en procesamiento automático!* Se lo discutirá luego al ver la *Teoría de Números* en este texto.

Mohammed ibn Musa al-Khwarizmi escribió en el siglo IX su obra *Kitab al-jabr wa al-muqabalah* que fue traducida en el siglo XII al latín introduciendo en Europa los números indo-arábigos y una serie de métodos de resolución de problemas matemáticos; de aquí se tomó el nombre de *álgebra* para estas temáticas. Además, generó otra obra *Algoritmi de numero Indorum* (su nombre en latín) del cual proviene el nombre *algoritmo*³.

A mediados del siglo XIX, la matemática inglesa Ada Augusta Byron, condesa de Lovelace (ahora sí pensando en procesamiento automático) realizó la construcción de algoritmos para una computadora mecánica completa que diseñó Charles Babbage, máquina que nunca pudo construirse; la tecnología de la época (engranajes, hilos, ruedas, palancas) no alcanzaban para su diseño. Por ello se considera que Ada es la primera programadora de computadoras.

Se suele atribuir a Böhm y Jacopini por su artículo de 1966, la demostración del *teorema fundamental de la programación estructurada* el cual establece que, *toda función computable*⁴ *puede ser calculada por un algoritmo que utilice sólo tres estructuras de control básicas: secuencia, alternativa y repetición*. Aquí, *estructuras de control* se refiere a la especificación del orden en que las instrucciones simples se ejecutarán dentro del procedimiento.

El cuadro de la figura 0.2 muestra algunas de las representaciones usuales de estas estructuras de control, donde **A** y **B** representan dos estructuras cualesquiera, incluida la más simple que corresponde a una instrucción individual.

En este texto, cuando sea necesario, utilizaremos el paradigma procedimental de programación y diversas formas de representar algoritmos, tanto en lenguaje natural como en diagramas de flujo; cuando se propongan fragmentos de código, salvo que se diga otra cosa, serán expresados en lenguaje Python.

No discutiremos aquí con mayor profundidad, porque excede el objetivo del texto, el tema *algoritmos*, pero quien desee profundizar puede consultar la bibliografía

¹ Para ampliar, consulte (Alcalde & García, 1987), (Frittelli, 2001), (Knuth, 2011).

² (Euclides, 2007)

³ (Ruiza, Fernández, & Tamaro, 2020)

⁴ Una función computable es aquella que puede ser calculada por el modelo formal propuesto por Alan Turing (matemático inglés 1912-1954) en 1936, hoy denominado Máquina de Turing. Ver (Giró, Vázquez, Meloni, & Constable, 2015) para una ampliación.

propuesta.

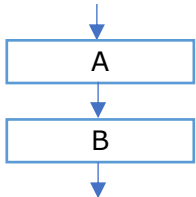
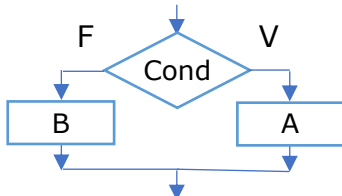
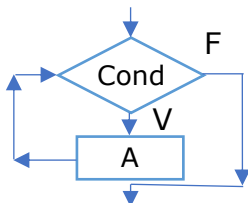
Estructura	Diagrama de flujo de control	Diagrama de Warnier	Código Python
SECUENCIA		$\left\{ \begin{array}{l} A \\ B \end{array} \right.$	A B
ALTERNATIVA		Si (cond) $\left\{ \begin{array}{l} V: A \\ F: B \end{array} \right.$	if (cond): A else: B
REPETICIÓN		Mientras (cond) $\left\{ \begin{array}{l} A \end{array} \right.$	while (cond): A

Fig. 0.2: Estructuras de la programación estructurada (**A**, **B**: estructuras cualesquiera, **cond**: condición que puede ser o bien [**V**]verdadera o bien [**F**]falsa).

Debe quedar claro que, para cada problema de cálculo a resolver, existen infinitos posibles algoritmos que lo resuelvan; todos tendrán las mismas entradas (datos) y las mismas salidas (resultados) pero variará su lógica, esto es, las instrucciones que lo componen y el orden de ejecución de ellas. Naturalmente siempre se debe optar por aquellos que sean más simples y eficientes.

EJERCICIOS Y PREGUNTAS DE REPASO

- Busque en Internet el algoritmo de Euclides para calcular el máximo común divisor, y diseñe un programa en el lenguaje de su preferencia para implementarlo.
- Elija la receta de su comida preferida, escríbala como un procedimiento e identifique en ella las entradas y salidas. Determine si esta especificación es un algoritmo o sólo un procedimiento, señalando por qué.
- Indague en Internet las biografías de todos los personajes mencionados en esta sección. En particular, encuentre cuál es el nombre de la máquina que diseñó Charles Babbage y quién fue el traductor del tratado de al-Khwarizmi que dio origen al término *álgebra*.
- Muestre al menos cinco algoritmos distintos para calcular la suma de los números naturales desde **1** hasta un número **n** indicado como entrada. Utilice el método de especificación que más le agrade.

0.4. Operaciones y relaciones

En matemáticas, computación y programación, se utilizan principalmente operaciones entre distintos elementos y relaciones definidas sobre ellos. Tanto las operaciones como las relaciones tienen sus propias características y pueden o no cumplir algunas propiedades, pero es importante identificar que son cosas bien distintas.

Durante el desarrollo de los distintos temas de Matemática Discreta se irá definiendo y trabajando permanentemente con relaciones y operaciones, pero aquí se quiere aclarar, preliminarmente, la diferencia entre estos conceptos apelando a temas ya conocidos desde la escuela.

Los números naturales son aquellos que sirven para contar cosas de la naturaleza; fueron los primeros utilizados por los seres humanos para denotar *la idea de cantidad*. Inicialmente no contenían el número cero, aunque ya al-Khwarizmi lo utiliza en sus escritos para explicar el sistema posicional de numeración hindú. Modernamente la definición de números naturales, como aquellos que permiten indicar la cantidad de elementos de un conjunto (su cardinalidad), incorpora el cero como el primer número natural.

Así, los números naturales son cero, uno, dos, tres, cuatro, ... (los tres puntos seguidos se leen “y así sucesivamente”) y usamos los símbolos 0, 1, 2, 3, 4, ... para denotarlos.

Todos hemos utilizado en la escuela los números naturales y hemos trabajado desde niños con ellos básicamente de dos formas:

- a) **Operando** con ellos mediante adición, sustracción, multiplicación o división para resolver distinto tipo de problemas.
- b) **Comparando** unos con otros para determinar si uno es mayor, menor o igual a otro, o si comparten alguna propiedad (como ser pares, por ejemplo), determinación que también se utiliza para resolver problemas.

En el primer caso se están realizando **operaciones** y en el segundo se están utilizando **relaciones**.

Definición: Dada una colección de objetos, una **OPERACIÓN** definida sobre la misma es una función (un algoritmo) que toma como argumentos (entrada) algunos de esos objetos y produce como resultado (salida) un único objeto, no necesariamente de la misma colección.

Si el resultado de una operación en todos los casos pertenece a la misma colección, se dice que la operación es **cerrada**. Así, la adición de números naturales es una operación cerrada ya que la suma de dos números naturales cualquiera es otro número natural. La división, por otro lado, no es una operación cerrada entre números naturales, porque no siempre la división de dos naturales tiene como resultado otro número natural. Los ejemplos son tan evidentes, que le dejamos al lector pensarlos y construirlos.

Por otro lado, si la operación se aplica sobre sólo un objeto de la colección se dice que es **unaria**, si se aplica sobre dos de ellos se dice que es **binaria**, y así sucesivamente.

Es oportuno recordad aquí la diferencia entre una operación aritmética, el símbolo que se utiliza para denotarla y su resultado; a veces se usan como sinónimos y eventualmente lo haremos, pero debe tenerse en cuenta que **sobre los números naturales**:

- $a + b = c$ La **ADICIÓN** es una operación binaria cerrada que se representa por el símbolo $+$, los números **a** y **b** se denominan **sumandos** y su resultado **c** es la **suma** de **a** y **b**. Se lee "**a más b**".
- $a - b = c$ La **SUSTRACCIÓN** es una operación binaria no cerrada que se representa con el símbolo $-$, el número **a** se denomina **minuendo**, **b** se llama **sustraendo** y el resultado **c**, si existe, recibe el nombre de **resta** entre **a** y **b**. Se lee "**a menos b**".
- $a \times b = c$ La **MULTIPLICACIÓN** es una operación binaria cerrada que representamos con el símbolo \times , los números **a** y **b** se llaman **factores** o multiplicandos y el resultado **c** recibe el nombre de **producto** entre **a** y **b**. Se lee "**a por b**".
- $a \mid b = c$ La **DIVISIÓN** es una operación binaria no cerrada que representamos con el símbolo \mid , **a** recibe el nombre de **divisor**, **b** se llaman **dividendo** y el resultado **c**, si existe, se denomina **cociente** entre **b** y **a**. Se lee "**a divide a b**".

Debe notarse que para la división se ha utilizado una barra vertical (\mid), y no la notación de colocar una barra horizontal con el divisor debajo de ella y el dividendo arriba ($\frac{b}{a}$) ya que esto es un número racional y no es correcto utilizarla si hablamos de números naturales.

Por otro lado, cuando se *comparan* dos números naturales lo que se está queriendo hacer es determinar si *determinada condición* se cumple o no se cumple entre ellos; el resultado de esta comparación NO es otro número en ningún caso, sino la verificación de si existe la condición enunciada entre ellos o no; por ello la comparación dará como resultado SI o NO, VERDADERO o FALSO, SE CUMPLE o NO SE CUMPLE. En el caso que efectivamente se verifique, se dice que los objetos están **relacionados** por la condición enunciada; en caso de no cumplirse, los objetos no estarán relacionados por la misma. Esencialmente, se trata de lo que en computación se denomina *un problema de decisión*.

Ejemplo: Al preguntar ¿**7** es mayor que **2**?, estamos queriendo saber si se cumple la condición "*ser mayor que*" entre **7** y **2**; si la condición se cumple, como en este ejemplo, la respuesta a la pregunta es **SI** y decimos que **7** *ESTÁ RELACIONADO CON* **2** por la condición "*ser mayor que*"; lo simbolizamos con $7 > 2$. Por otro lado, si preguntamos ¿es tres igual a cuatro?, la respuesta es **NO** y decimos que los números tres y cuatro no están relacionados por la condición de igualdad; lo simbolizamos con $3 \neq 4$. **■**

¿Cómo se determina si los objetos cumplen o no determinada condición entre ellos? La condición puede ser muy sencilla (como la de "*ser mayor que*" entre números naturales) o complicada (dos números naturales estarán relacionados entre ellos si la suma

de sus cuadrados es un número primo); por ello, la determinación puede ser trivial, haciéndose por simple inspección de los objetos, o puede requerir que se defina un algoritmo para verificarla.

Luego tendremos una mejor definición de relación cuando se discuta en el texto el tema específico, pero podemos aquí ir adelantando un primer intento:

Definición: Dada una colección de objetos y una condición enunciada entre dos cualesquiera de ellos, se llama **RELACIÓN BINARIA** a la colección de *pares de objetos* de la colección dada que verifican la condición.

Así, la condición de “ser menor que” entre los números naturales, dará como resultado la colección infinita de pares $(0, 1)$, $(0, 2)$, $(0, 3)$, ..., $(1, 2)$, $(1, 3)$,

Lo que debería quedar claro luego de esta breve discusión, es que *operación* y *relación* son cosas completamente distintas.

EJERCICIOS Y PREGUNTAS DE REPASO

1. Busque en Internet definiciones del concepto de *operación* y de *relación*, escribálas y discuta brevemente sus similitudes y diferencias (en caso de encontrar distintas explicaciones del mismo concepto).
2. Explique por qué la sustracción de números naturales NO es una operación cerrada.
3. Realice una definición directa de la operación de *potenciación* y de *radicación*, sobre los números naturales. Diga si estas operaciones son cerradas e indique los nombres de cada una de sus partes constituyentes.
4. Suponga que se tiene una colección de programas escritos en distintos lenguajes (Java, Python, C, FORTRAN, etc.) y se define el concepto de “*equivalencia*” entre ellos diciendo que *dos programas son equivalentes si ante las mismas entradas entregan al funcionar las mismas salidas*. La equivalencia entre programas ¿es una operación o una relación?
5. Escriba en castellano el procedimiento necesario para efectuar la división de dos números naturales cualesquiera. Pruebe luego su procedimiento con números de distinto tamaño generando todos los casos posibles: números iguales, el primero mayor que el segundo, el segundo mayor que el primero, números que son divisibles entre ellos y que no lo son. Piense también qué pasa si uno de esos números es el cero.
6. Compilar un programa de computación escrito en un lenguaje de alto nivel como C, llamado programa fuente, es un proceso que verifica si está bien escrito (cumple las especificaciones del lenguaje) y, en caso de estar correcto, genera un programa equivalente en un lenguaje de más bajo nivel denominado programa objeto. ¿Llamaría a la compilación de un programa una operación o una relación?

0.5. Resumen

En estas *Nociones Básicas* se han discutido conceptos generales que son comunes a todas las ciencias y en particular, a la matemática y la computación.

Definición	Explicación clara y sin ambigüedades de un concepto, utilizando términos técnicos y comunes con significado conocido.
Tipos de Definición	<u>Intuitiva</u> : definición informal de un concepto o término primitivo. <u>Directa</u> : definición usando sólo términos ya conocidos. <u>Recursiva</u> : definición usando términos conocidos y el mismo término que se está definiendo.

Constante	Objeto que no cambia su significado o valor a lo largo de un contexto.
Variable	Objeto que cambia su significado o valor durante la discusión de un tema.

Procedimiento	Secuencia finita de acciones simples prescriptas en un cierto orden.
Algoritmo	Procedimiento bien definido que siempre termina con cualquier entrada.

Operación	Función o algoritmo que permite obtener un resultado desde datos.
Relación	Colección de pares de objetos que verifican una condición.

0.6. Bibliografía

- Alcalde, E., & García, M. (1987). *Metodología de la Programación*. Madrid, España: McGraw-Hill.
- Euclides. (2007). *ELEMENTOS*. Barcelona, España: Gredos.
- Frittelli, V. (2001). *Algoritmos y Estructuras de Datos*. Córdoba, Argentina: Universitas.
- Giró, J. F., Vázquez, J. C., Meloni, B. E., & Constable, L. E. (2015). *Lenguajes Formales y Teoría de Autómatas*. Buenos Aires, Argentina: Alfaomega.
- Knuth, D. (2011). *The art of computer programming*. Reading, Massachusetts, USA: Addison-Wesley.
- Masini, G. (1980). *El Romance de los Números*. Barcelona, España: Círculo de Lectores.
- Ruiza, M., Fernández, T., & Tamaro, E. (15 de 01 de 2020). *Biografías y Vidas*. Obtenido de Biografia de Al-Khwarizmi: <https://www.biografiasyvidas.com/biografia/k/khwarizmi.htm>