1. Resolución de problemas

> *Pablo tiene $50 en su bolsillo, y quiere comprar caramelos para 3 amigos y quedarse con al menos 1 para él. Sabiendo que los caramelos cuestan $1.25, ¿Cuántos caramelos puede comprar? ¿Cuántos caramelos recibirá cada uno de sus amigos? ¿Cuántos caramelos se quedará Pablo?*

Este es un típico **problema** al que suele enfrentarse todo estudiante que haya cursado matemática, pero *¿es* realmente un problema? Quizás para algunos lo sea y para otros es simplemente un ejercicio más en una guía. Quizás alguna vez se enfrentó a una *situación problemática* similar a esta, en la cual debió aplicar su capacidad de compresión, traducción o transformación de la información. A continuación, enunciaremos procedimientos o herramientas para enfrentar de forma *ordenada* estas situaciones.

La resolución de un problema es el punto donde suponemos que se ha alcanzado la conclusión de un proceso más amplio. Tiene, como pasos previos, la identificación del problema y *su modelado*. Por **problema** se entiende un asunto del que se espera una solución que no es obvia a partir del planteamiento inicial. El matemático Wheatley (1984) lo definió de la siguiente manera: "la resolución de problemas es lo que hacés cuando no sabes qué hacer".

Sobre este tema se encuentran muchos textos, pero haremos hincapié en lo que plantea el matemático Polya (1994) y su metodología de 4 pasos para resolver problemas:

1.1. Entender el problema

Los problemas suelen presentarse en lenguaje natural de forma oral o escrita, por lo tanto, debemos interpretarlos para luego poder generar una **abstracción**. En esta etapa identificaremos los *datos* del problema, los cuales influyen en el resultado final. Por ejemplo, en el caso de Pablo, lo que justamente no interesa es su nombre, podría llamarse Darío y la *esencia* del problema quedaría intacta. Podemos *abstraernos* del nombre propio y buscar otra información que sea *sustancial* para responder a las preguntas que se plantean.

Al momento de la interpretación debería poder contestar todas o algunas de las siguientes preguntas:

- ¿Entendés todo lo que dice?
- ¿Podés replantear el problema en tus propias palabras?
- ¿Distinguís cuáles son los datos?
- ¿Sabés a qué querés llegar?
- ¿Hay suficiente información?
- ¿Hay información extraña o de sobra?
- ¿Es este problema similar a algún otro que hayas resuelto antes?

En el caso de Pablo, entendiendo que es un problema dentro de un contexto matemático, podemos suponer que debemos extraer los datos numéricos:

- Dinero disponible = \$50
- Cantidad de amigos = 3
- Costo de los caramelos = \$1.25
- P. se queda con *al menos* 1 caramelo.

1.2. Configurar un plan

Una vez extraídos los datos o información relevante, debemos empezar a buscar qué hacer con estos. Qué relación existen entre ellos y decidir cómo proceder o manipularlos. Para ello se le proponen algunas herramientas que suelen ser utilizadas como estrategia para empezar a *"encarar"* el problema.

1. Ensayo y error (Conjeturar y probar la conjetura).
2. Usar una variable.
3. Buscar un patrón.
4. Hacer una lista.
5. Resolver un problema similar más simple.
6. Hacer una figura.
7. Hacer un diagrama.
8. Usar razonamiento directo.
9. Usar razonamiento indirecto.
10. Usar las propiedades de los números.
11. Resolver un problema equivalente.

12. Trabajar hacia atrás (empezar desde la solución).
13. Usar casos.
14. Resolver una ecuación.
15. Buscar una fórmula.
16. Usar un modelo.
17. Usar análisis dimensional.
18. Identificar sub-metas.
19. Usar coordenadas.
20. Usar simetría.

Por ejemplo, la consigna del problema de Pablo tiene 3 preguntas, por lo tanto seria una posibilidad tomar cada una como un sub-problema, e ir resolviendo del más sencillo al más difícil.

1.3. Ejecutar el Plan

Llega la hora de *ensuciarse* las manos, debemos implementar la o las estrategias que se escogieron hasta solucionar completamente el problema o hasta que la misma acción sugiera tomar un nuevo curso. Concederse un tiempo razonable para resolver el problema. Si no hay éxito, solicitar una sugerencia o hacer el problema a un lado por un momento (¡puede que "se prenda el foco" cuando menos se espera!). No tener miedo de volver a empezar. Suele suceder que un comienzo fresco o una nueva estrategia conduzcan al éxito.

Más adelante en el apunte resolveremos el planteo inicial.

1.4. Retroalimentación (mirar hacia atrás)

Una vez alcanzados resultados, debemos *revisar* que sean los adecuados. Para eso se sugiere que se haga las siguientes preguntas:

¿Es la solución correcta? ¿la respuesta satisface lo establecido en el problema?

¿Se advierte una solución más sencilla?

¿Puede verse cómo extender la solución a un caso general?

Es importante **verificar** los resultados obtenidos, ya que luego de un tiempo razonando o elaborando la solución pudo habernos llevado a un error, o incluso podríamos darnos cuenta de que hicimos pasos de más o podríamos simplificar algún *sub-proceso* utilizado.

También podríamos encontrar una pequeña modificación que logre llevar nuestra solución puntual a una solución que resuelve una situación similar o más general.

2. Definición

Como se mencionó, comúnmente los problemas se enuncian en palabras. Así, para resolver un problema, luego de interpretarlo uno traslada las palabras a una forma equivalente del problema, es decir que la **modeliza**, resuelve esta modelización e interpreta la respuesta. Por ejemplo, el problema de Pablo se puede traducir al lenguaje matemático, aplicando variables y operadores tales como la suma, la multiplicación, la división, etc.

Un algoritmo se puede definir como **una secuencia de instrucciones que representan un *modelo de solución* para determinado tipo de problema**. O bien como un conjunto de instrucciones que realizadas en orden conducen a obtener la solución de un problema.

La programación –en el campo de la computación– es la técnica de traducir un algoritmo a algún lenguaje o gráfico que pueda ser, a su vez, transformado a un lenguaje que pueda ejecutar una máquina[1].

Es necesario primeramente diseñar el algoritmo y una vez definido, expresarlo en algún lenguaje de programación, a este proceso se lo conoce comúnmente como **codificar** o **programar**. Luego, se **depura**, el programa es ejecutado por la máquina y debe validar que la salida sea las esperada. Luis Joyanes, autor de muchos libros acerca de lógica y programación dice: "(...) en la programación, los algoritmos son más importantes que los lenguajes de programación o las computadoras. Un lenguaje de programación es sólo un medio para expresar un algoritmo y una computadora es sólo un procesador para ejecutarlo" (Joyanes Aguilar y Zahonero Martínez, 2010, p. 15).

Los algoritmos son independientes de los lenguajes de programación. En cada problema el algoritmo puede escribirse y luego ejecutarse en un lenguaje diferente de programación. El algoritmo es la infraestructura de cualquier solución, escrita luego en cualquier lenguaje de programación.

[1] Una computadora no solo es una PC, hoy en día los teléfonos celulares, calculadoras, tablets, entre otros dispositivos que son computadoras o las poseen para su funcionamiento.

3. Características

Los algoritmos tienen las siguientes características esenciales:

1. **Preciso**. No da lugar a ambigüedades, su descripción es única.
2. **Definido**. Si se utiliza el mismo algoritmo bajo las mismas condiciones siempre se obtiene el mismo resultado.
3. **Finito**. El algoritmo tiene un comienzo, un determinado número de pasos y un final.
4. **Parámetros**. Opcionalmente puede recibir datos de *entrada* que alteran el resultado.
5. **Salida**. Siempre genera una solución o resultado, que es el propósito del algoritmo.

Si tuviera que resolver una cena con menú italiano, podría seguir una receta de cocina, p.e. canelones de espinaca y ricota con salsa a los 4 quesos. Y siguiendo su preparación al pie de la letra, controlando las variables (cantidades exactas, temperatura del horno, dimensiones de las cazuelas, etc.), todas la veces que *ejecute* esta receta llegaría siempre al mismo resultado. En este caso, la receta (precisa y definida), especifica ingredientes (parámetros de entrada), que resuelve la cena italiana (problema), la cual termina (finita) cuando el plato está listo (salida).

4. ¿Cualitativos o cuantitativos?

Un algoritmo es **cualitativo** cuando en sus pasos o instrucciones no están involucrados cálculos numéricos. Los pasos para cebar un mate, para desarrollar una actividad física o armar un modelo a escala, son ejemplos de algoritmos cualitativos.

Los algoritmos **cuantitativos** involucran cálculos numéricos. Por ejemplo, calcular un factorial, resolver una ecuación de segundo grado, encontrar la intersección entre dos rectas, etc.

¿Qué tipo de algoritmo se necesita diseñar para el problema de Pablo? ¿Y el de la cena italiana?

5. Representación

Un algoritmo puede ser representado de muchas maneras, las más usuales suelen ser las siguientes:

- Lenguaje natural (español, inglés, francés, etc.)
- Lenguaje matemático
- Pseudocódigo
- Diagramas (Nassi-Shneiderman (N-S), de flujo)

El manejo promedio que tenemos del lenguaje natural puede no resultarnos lo suficientemente preciso, permitiendo ambigüedades, obteniendo una descripción no del todo satisfactoria del algoritmo. Las ecuaciones, propias del lenguaje matemático, son un buen sistema de representación, y muy eficientes para lenguajes de programación del *paradigma funcional*. En este curso nos centraremos en otro paradigma de la programación, conocido como *paradigma imperativo y procedural*, es decir, orientado a procesos. Por ello, los dos últimos mencionados son más adecuados para describir los algoritmos que implementaremos, aunque siempre vendrá bien saber matemática.

5.1. Lenguaje matemático

Resolvamos el problema de Pablo y dejemos planteados los algoritmos (ecuaciones) necesarios para contestar a cada una de las preguntas del enunciado. Cada una de ellas tendrá su algoritmo y serán necesarias las salidas de los anteriores para responder (modelizar) las siguientes preguntas.

¿Cuántos caramelos puede comprar?

c: cantidad de caramelos que puede comprar Pablo.

$$c \in \mathbb{N}_0 / c = \$50 \div \$1{,}25$$

Para los que no están familiarizados con la simbología matemática, la traducción en lenguaje natural sería: para c (variable) perteneciente al conjunto de los Naturales y el cero tal que c es \$50 dividido \$1,25. Es decir, el dinero que posee Pablo, dividido el precio unitario de cada caramelo. Es importante aclarar que c es un número natural ya que en este caso es imposible que compre caramelos por fracción (medio caramelo, un cuarto de caramelo, etc.).

¿Cuántos caramelos recibirá cada uno de sus amigos?

a: cantidad de caramelos que recibe cada amigo.

$$a \in \mathbb{N}_0 / a = (c - 1) \div 3$$

En este caso, tomamos el resultado (salida) del algoritmo anterior, le restamos un caramelo que se queda Pablo, y lo dividimos por la cantidad de amigos.

¿Cuántos caramelos se quedará Pablo?

p: cantidad de caramelos que se queda Pablo.

$$p \in \mathbb{N}_0 / p = c - (a \times 3)$$

Por último, calculamos la cantidad de caramelos que repartió Pablo entre sus amigos $(a \times 3)$ y se lo restamos a la cantidad de caramelos comprados (c), que nos deja el resto de caramelos no repartidos, en otras palabras, que se quedó Pablo.

Cabe destacar que no es la única secuencia o posible camino hacia la solución. ¿Se le ocurre otra manera de resolver este problema?

5.2. Pseudocódigo

El pseudocódigo es un lenguaje informal interpretable por seres humanos, que describe un algoritmo valiéndose del lenguaje natural. Éste toma su formato de las estructuras más usuales de los lenguajes de programación procedurales. Un ejemplo podría ser el siguiente:

Programa 1: Cálculo del cuadrado

```
Algoritmo calcularCuadrado
  Escribir "Ingrese un número: "
  Leer N
  Escribir "El cuadrado de ", N, " es ", N*N
Fin Algoritmo
```

Hay ciertas reglas que debemos cumplir. En pseudocódigo cada bloque tiene un principio y un final explícito y el algoritmo posee un nombre único. Observe que las acciones o instrucciones son escritas en infinitivo y los textos o frases que deseamos que la computadora escriba deben ir entre comillas. Si deseamos intercalar valores de variables con textos podemos separarlos entre comas.

Una posible salida de este algoritmo sería el siguiente:

```
Ingrese un número: 3
El cuadrado de 3 es 9
```

5.3. Diagramas Nassi-Shneiderman y de flujo

Los diagramas son representaciones gráficas de los algoritmos que permiten una rápida visualización y seguimiento de los pasos o instrucciones que componen la solución que modeliza. En este apunte no describiremos todos los posibles elementos presentes en los mismos, los iremos descubriendo a medida que avancemos en el curso. Así como sus características y utilidad al momento de diseñar un algoritmo.

Veamos el siguiente pseudocódigo y sus representaciones gráficas tanto en un diagrama N-S como en diagrama de flujo.

Programa 2: Determinar si un número es positivo

```
Algoritmo numeroPositivo
  Escribir "Ingrese un número:"
  Leer N
  Si N Es Mayor Que 0 Entonces
    Escribir "El número es positivo"
  SiNo
    Escribir "El número no es positivo"
  Fin Si
Fin Algoritmo
```

Figura 1: Diagrama de flujo del algoritmo

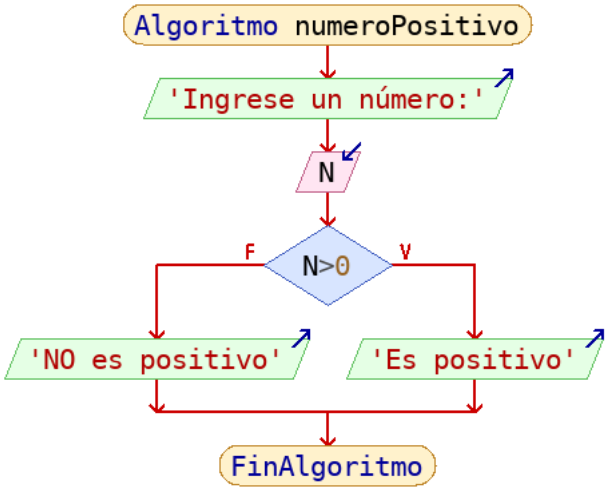
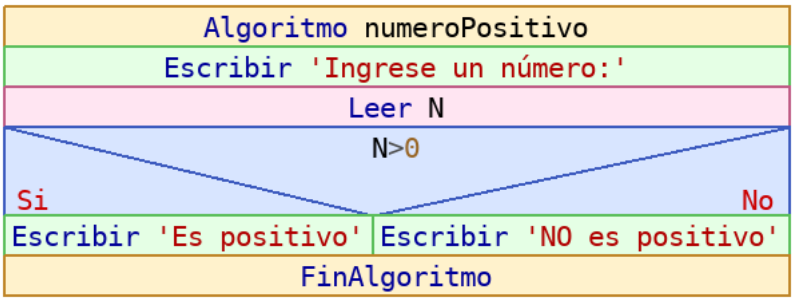


Figura 2: Diagrama N-S del algoritmo



El diagrama de flujo (Figura 1), como lo indica su nombre tiene un *camino* único señalizado por flechas. Podemos observar una bifurcación por medio de un rombo que toma dos posibles caminos, pero solo puede salir por uno, ***V*** en caso de que la **proposición lógica** sea *verdadera* o ***F*** en caso contrario.

Los diagramas N-S (Figura 2) están compuestos por *cajas* donde observamos una similitud casi idéntica al pseudocódigo. La condición está en un recuadro que abre dos columnas encabezadas por un **Sí** y un **No** indicando por cual debe seguir el algoritmo.

Ahondaremos más en las **sentencias de control** como el *condicional* en los próximos apuntes.

6. Ejecitación

Clasifique en cualitativos o cuantitativos e intente diseñar los algoritmos de los siguientes casos:

- Hallar el valor absoluto o módulo de un número entero.
- Cebar un mate.
- Realizar una llamada telefónica.
- Calcular un promedio de notas trimestrales.
- Cambiarle los pañales a un bebe.
- Preparar arepas.
- Calcular el sueldo mensual de un operario.

Referencias

Joyanes Aguilar, L., y Zahonero Martínez, I. (2010). *Programación en c: metodología, algoritmos, estructura de datos y objetos*. Madrid: McGraw-Hill.

Polya, G. (1994). *Cómo plantear y resolver problemas*. México D.F.: Trillas.

Wheatley, G. H. (1984). *Problem solving in school mathematics* (inf.téc. MEPS n.º 84.01). West Lafayette, Indiana: Purdue University, School of Mathematics and Science Center.