Datos y Variables

Dato

Un dato es una representación simbólica (un número, letra, gráfico, entre otros) de una característica de un objeto. Un dato puede estar representado por una cifra, letra, palabra o conjunto de palabras que describen una característica de dicho objeto. Por ejemplo, “Florencia González” y “20” pueden representar el nombre y la edad de una persona respectivamente, donde el nombre y la edad son las características de esa persona.

Un solo dato no representa información, sino que la información está conformada por un conjunto de datos que, al ser procesados mediante algún mecanismo, constituyen un mensaje para incrementar el conocimiento de quien lo recibe.

Tipos de datos

Un tipo de dato corresponde a una clasificación que se hace para poder tratar cada dato de la forma más adecuada, según lo que se requiera. El tipo de dato le indica al dispositivo de procesamiento cuanto espacio de memoria debe reservar para almacenar el dato. Los tipos de datos más comunes son: alfanuméricos, numéricos y lógicos.

Datos alfanuméricos

Estos datos se componen de la combinación de todos los caracteres conocidos: letras del alfabeto, dígitos y caracteres especiales, incluyendo el espacio en blanco. Los datos alfanuméricos se dividen en carácter y cadena.

Caracter: se refiere a los datos que solo tienen un carácter, que puede ser una letra del alfabeto, un dígito del 0 al 9 o un carácter especial. Por ejemplo: ’a’, ’R’, ’2’, ’#’, ’@’

Los dígitos que son tratados como caracteres son diferentes a los valores numéricos, por lo tanto, no se deben realizar operaciones aritméticas con ellos. Los diferenciamos encerrándolos entre comillas.

Cadena: son los datos que están compuestos por un conjunto de letras del alfabeto, dígitos y caracteres especiales, incluyendo el espacio en blanco. Los datos de tipo cadena también deben encerrarse entre comillas. Son ejemplos de datos de tipo cadena los siguientes: “Módulo 45 # 23 – 68”, “León”, “Felipe Ferreyra”, “42478923”, “08004243433”.

Los dos últimos ejemplos podrían corresponder a un número de DNI y al número de una línea de teléfono: aunque estos datos están conformados por dígitos, no pueden ser utilizados para hacer cálculos numéricos.

Datos numéricos

Corresponden a los datos que están compuestos por solo números y signos (positivo y negativo), es decir, dígitos del 0 al 9, con los cuales se pueden realizar operaciones aritméticas. Estos tipos de datos se subdividen en enteros y reales.

Entero: son aquellos datos compuestos por números que no tienen punto decimal. Pueden ser números positivos, negativos o el cero. Ejemplos de este tipo de datos pueden ser: 20, -5, 200, 1500000.

Real: son datos con componente decimal, pueden ser positivos, negativos o cero. Ejemplos: 1.75, 4.5, 1800000.00, -234.00. Las unidades de mil no se deben separar con puntos o comas. Solamente se usa el punto para indicar la parte decimal

Datos lógicos o booleanos

Son aquellos datos que toman solo uno de los dos posibles valores booleanos: Verdadero o Falso. Estos valores son equivalentes a los dígitos del sistema binario: 1 corresponde a Verdadero y 0 a Falso.

Por ejemplo, supongamos que se necesita almacenar la respuesta suministrada por un paciente en una clínica sobre si fuma o no, o sobre si es alérgico o no a un tipo de medicamento. En ambos casos, se debería utilizar este tipo de datos para clasificar el dato suministrado por el paciente.

Identificador

Un identificador es el nombre que se le asigna a una variable, constante, función, procedimiento o algoritmo. Esto se hace para que se los pueda identificar claramente.

Existen una serie de reglas para asignar nombres a estos elementos, que van a depender del lenguaje de programación que vayamos a utilizar.

Algunas de las reglas:

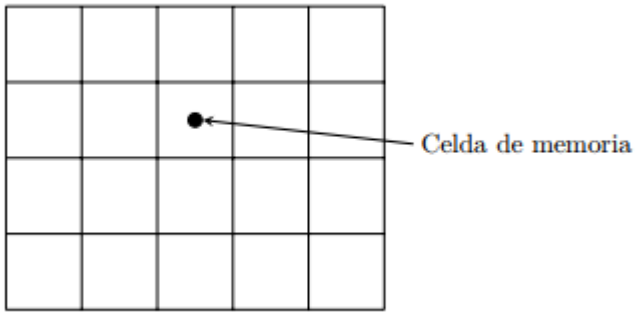
- Elegimos identificadores mnemotécnicos: que sean alusivos o explicativos del elemento al que están nombrando
- El primer carácter debe ser una letra
- No utilizar caracteres especiales, ñ, vocales tildadas o símbolos como $, #,?,!, etc.

- No dejar espacios en blanco
- No utilizar palabras propias del lenguaje o algoritmo = “Palabras reservadas” del lenguaje.
- Un identificador puede constar de varias palabras unidas o separadas por guion bajo.

Variables

Para almacenar los datos en una computadora, se utiliza su memoria, que se puede comparar con una serie de casilleros para guardar valores.

Cada cuadrito representa una dirección en la memoria, a la que le podemos asignar nombre, por medio de los identificadores, que acabamos de explicar.



Una variable es una posición o espacio de memoria en el cual se almacena un dato. Su valor puede cambiar en cualquier momento de la ejecución del algoritmo. Por ejemplo: durante la operación diaria, la variable que almacena el dato con el saldo de una cuenta bancaria, sufre modificaciones.

Una variable consta de tres elementos:

- el tipo de dato: carácter, cadena, entero, real, booleano/lógico
- el nombre o identificador: el nombre con que denominamos a ese espacio (cuadrito) de memoria
- contenido o dato en sí mismo: el valor almacenado.

Almacenando un dato en una variable

Podemos almacenar datos en una variable:

1. Leyendo el dato desde el exterior del programa. Ej: El sistema le pide al usuario que ingrese su DNI para luego guardarlo en una variable.

2. Por medio de una expresión de asignación, para lo que utilizamos el signo igual (=)

Su sintaxis es la siguiente: primero se pone el identificador de la variable, el signo igual y, por último, el valor que vamos a asignarle.

```
variable = valor
```

Por ejemplo:

```
dni = “42548965”
telefono = “45872563”
salario = 450000,00
edad = 27
es_fumador = Verdadero
estratoSocioeconomico = “3”
```

Las variables dni, teléfono y estratoSocioeconomico, si bien constan de dígitos, no están destinados a operaciones aritméticas, por lo que son tratados como cadenas de texto o caracteres, por lo que los encerramos entre comillas.

Los últimos dos identificadores constan de dos palabras unidas. Vemos que podemos mostrarlos de dos formas: unidos por el guión bajo, o diferenciando cada palabra, comenzándola con mayúscula.

Entonces, en este ejemplo, tenemos 6 variables que almacenan datos, a los que podemos individualizar, leer y operar, gracias a su identificador.

¿Cómo cambiamos el valor de una variable?

Veamos el siguiente ejemplo y tratemos de pensar qué está sucediendo:

```
a = 5
b = a
c = 3 * b
a = c – b
```

¿Cuánto vale a?

¡Atención!: Cada vez que una variable cambia de valor, el dato anterior se pierde.

Constantes

Una constante es un espacio de memoria donde se almacena un dato que permanece constante durante la ejecución de todo el algoritmo. También tienen tipo, identificador y valor. La sintaxis para la asignación es el mismo que para las variables, pero podemos diferenciar variables y constantes por medio de su identificador: en la constantes se suelen escribir en mayúsculas. Por ejemplo:

```
VALOR_PI = 3.1415926
IVA = 0.21
MAXIMO = 10
SALARIO_MINIMO = 67000.00
CATEGORIA = “B”
```

Fuente: ‘Introducción a la Lógica de programación’, Herrera Molares, Gutiérrez Posada, Pulgarín Giraldo, 2017

Lógica proposicional

La lógica proposicional es un sistema basado en proposiciones y conectores que permiten realizar operaciones sobre las proposiciones y crear otras proposiciones más complejas.

La proposición

La proposición no es más que un enunciado o afirmación. Su característica fundamental y distintiva, es que tiene un valor de verdad. Una expresión para ser considerada una proposición o enunciado, debe poder ser evaluada y declarada como verdadera o falsa.

Veamos algunos ejemplos para distinguir cuáles de las siguientes expresiones son proposiciones:

Expresión	¿Es proposición?
¿Cuánto stock queda?	NO
5 + 2 = 9	SI
Argentina está en América del Sur	SI
Ojalá mejore el clima	NO
4 + 2 = 6	SI
Alcanzame aquel libro	NO
París es la capital de Bolivia	SI
Hoy hizo calor	SI

La pregunta que debemos hacernos es: ¿Puedo evaluar si ese enunciado es VERDADERO o FALSO? Si la respuesta es “si”, entonces decimos que esa expresión **tiene un “valor de verdad”**, y estamos sin dudas ante una proposición.

¿Para qué sirven dentro del ámbito que estudiamos? Según el valor de verdad de una proposición, en un programa u aplicación se produce una bifurcación, en la que se toma una decisión y se comporta de distinto modo en uno u otro caso.

Ejemplos:

Si el sistema de solicitudes de licencias de conducir, detecta que la proposición que describe una situación del usuario:

- “tiene la edad mínima requerida” es FALSO, rechaza la solicitud.
- “tiene la edad mínima requerida” es VERDADERO, sigue adelante con el trámite.

Los principios lógicos supremos

La lógica es una disciplina científica que se fundamenta en tres principios lógicos supremos. Estos son: el principio de identidad, el principio de no-contradicción y el principio de tercero excluido.

- El principio de identidad se resume en que toda proposición es equivalente a sí misma.
- El principio de no contradicción es el fundamento tradicional de todas las verdades. Expresa que: ninguna proposición es al mismo tiempo verdadera y falsa. Una proposición y su negación no pueden ser verdaderas o falsas al mismo tiempo.
- El principio del tercer excluido afirma que una proposición es verdadera o falsa.No existen otras posibilidades.

Relaciones

En la lógica proposicional nos interesa conocer la relación que existe entre las proposiciones. Estas relaciones parten de evaluar un conjunto de proposiciones y enunciar reglas que se cumplirán para todas la proposiciones que se relacionen de la misma manera, sin importar el contexto ni de qué proposiciones se trate.

Conjunción

Evaluemos el siguiente ejemplo:

Proposición simple	**Proposición simple**	**Proposición compuesta (conjunción)**
p	**q**	**p y q**
Hace frío	*Llueve*	*Hace frío y llueve*
V	V	V
V	F	F
F	V	F
F	F	F

Tabla de verdad N.º 1

En este ejemplo tenemos las dos primeras proposiciones con un valor de verdad “verdadero” y una tercera proposición compuesta por una relación entre las dos anteriores. Esta relación esta dada por el conector “y”, llamado también “conjunción”.

En la tabla tenemos reflejados todos los posibles valores de verdad que pueden tener estas dos proposiciones, y su conjunción. La conjunción “exige” para ser verdadera, que ambas proposiciones que la componen sean verdaderas. Cualquier otra combinación, resulta en “p y q” falso.

Podemos simplificar esta relación entre dos proposiciones verdaderas, y una proposición compuesta, reemplazando las proposiciones simples por las letras p, q, quedando:

- Si p es verdadero, q es verdadero, entonces, p y q es verdadero.
- Si p es verdadero, q es falso, entonces, p y q es falso.
- Si p es falso, q es verdadero, entonces, p y q es falso.
- Si p es falso, q es falso, entonces, p y q es falso.

Esta manera de representar la relación entre dos proposiciones y todas las posibles combinaciones entre sus valores de verdad, se llama “tabla de verdad” y establece las reglas de esa relación, sin importar cuáles sean estas proposiciones, ni su contexto.

La utilidad de estas tablas radica en que muestra de manera abstracta todas las posibilidades con las que podemos encontrarnos. Puede parecer difícil encontrarle un valor práctico si nos quedamos con la obvia idea de “si p es verdadero, q es verdadero, entonces p y q es verdadero”. Pero en la vida real “p y q” no se manifiesta de manera tan literal.

A la hora de, por ejemplo, evaluar los riesgos para establecer la cuota de un seguro, nuestro sistema formado por p, q, p y q, podría tomar una forma concreta de la siguiente forma:

p: La propiedad cuenta con sistema de alarma, q: La propiedad tiene rejas instaladas, p y q:la propiedad es segura.

Aquí vemos que “p y q”, toma en la vida real un enunciado que implícitamente presupone a p y q.

Si creamos la tabla de verdad para este caso, quedaría:

p	q	p y q
La propiedad cuenta con sistema de alarma	La propiedad tiene rejas instalada	La propiedad es segura
V	V	V
V	F	F
F	V	F
F	F	F

Tabla de verdad N.º 2

Al comparar las tablas de verdad N.º 1 y N.º 2, podemos concluir que esta misma tabla de verdad va a aplicar (como una regla universal) para **todos los casos de conjunción entre dos proposiciones.**

Actividad

(actividad no entregable, no calificable)

Otro de los conectores más utilizados es el **“o”**, tambíen llamado **“disyunción”**. Al igual que el conector “y”, este conector une dos proposiciones, pero a diferencia de la conjunción, basta con que al menos una de las proposiciones simples (ya sea p, o la proposición q) sea evaluada como verdadera, para que la proposición compuesta “p o q” sea verdadera. La única manera de que esta proposición compuesta resulte falsa, es que ambas proposiciones simples sean falsas.

¿Te atreves a completar la tabla de verdad de la disyunción? Podés utilizar la forma abstracta (p, q, p o q), o apoyarte en proposiciones más literales.

No te preocupes si no lo logras, la respuesta la encontrarás en el siguiente punto.

p	q	p o q
...	...	...
V	V	?
V	F	?
F	V	?
F	F	?

Tabla de verdad N.º 3

Tablas de verdad

Si al ingresar a mi cuenta de homebanking ingreso usuario y clave:

- correcto (V), entro al sistema y me habilita a realizar operaciones.
- incorrecto (F), no accedo al sistema.

Si prestamos atención, nos habremos dado cuenta que en el segundo ejemplo necesitamos dos proposiciones con valor de verdad (V) para poder acceder al homebanking: uno que verifica el nombre de usuario, y otro que verifica la clave.

A medida que las operaciones se van complejizando, es posible que necesitemos “combinar” varias proposiciones para saber el valor de verdad final.

Para ello utilizamos las tablas de verdad y las tres básicas son NO, O e Y.

Proposiciones – Negación
Tabla de verdad

p = “es mayor de edad”	
p	no-p
V	F
F	V

En esta primera tabla rige el principio de no contradicción que estudiamos en el punto anterior. Una proposición es verdadera o falsa. No puede ser ambas cosas al mismo tiempo. Si “p” es verdadero, “no p” es falso. Y viceversa.

Proposiciones – Disyunción (O)
Tabla de verdad

p = “es mayor de edad” q = “viene con un mayor”		
p	q	p o q
V	V	V
V	F	V
F	V	V
F	F	F

Proposiciones – Conjunción (Y)

Tabla de verdad

p = “usuario correcto” q = “clave correcta”		
p	**q**	**p y q**
V	V	V
V	F	F
F	V	F
F	F	F

A partir de estas tablas, podemos ir resolviendo operaciones más complejas entre proposiciones (veremos algunos ejemplos en clase)

Sin embargo, existen algunas leyes de lógica proposicional que nos permiten simplificar la resolución de estas operaciones. Podemos nombrar como ejemplo:

- Ley de Composición
- Leyes de De Morgan (2)
- Leyes de Conmutatividad
- Leyes de Asociatividad
- Leyes Distributivas
- Ley de Doble negación