UNLP

FACULTAD DE INFORMÁTICA

TALLER DE LECTO COMPRENSIÓN Y TRADUCCIÓN EN INGLÉS

ADJ. ANALIA NAPOLITANO
ADJ. SANDRA PERALTA

AÑO 2018

GUIA DE TRABAJOS PRACTICOS

AYUDANTES DIPLOMADOS:
Gabriela FRAGA
Florencia REGUERAL
María Paula GAVAGNIN

AYUDANTE ALUMNO:
Araceli BUFFONE

What do computer engineers do?

Computer hardware engineers "research, design, develop and test computer systems and components such as processors, circuit boards, memory devices, networks and routers," according to the U.S. Bureau of Labor Statistics (BLS). Computer hardware includes:

- microprocessors;
- memory chips: random-access memory (RAM), read-only memory (ROM) and nonvolatile rewritable flash memory;
- data storage devices: hard disks, solid-state drives and optical drives;
- input devices: keyboards, mice, joysticks and gaming controllers, cameras, microphones, scanners, touch screens and remote sensors;
- output devices: printers, monitors, audio devices and remote controls; and
- networking components: adapters, modems, switches and routers.

An important function of computer engineers is to integrate **these** components into computer and network systems. **This** all requires a good working knowledge of electrical engineering.

Another important aspect of computer engineering is software development. Computer software includes:

- operating systems;
- applications: word processing, spreadsheets, accounting, database management, graphics, computer-assisted design (CAD), computer-assisted manufacturing (CAM), audio, video, media and games;
- networking and communications: World Wide Web (WWW), voice over Internet Protocol (VOIP), instant messaging and email;
- utilities: file handling, disk management, device drivers, archiving and backup systems;
- programming languages: editing, compiling and debugging; and
- security: antivirus, firewalls, encryption and user authentication.

EJERCICIO 1

A. Lea solo el título. ¿En qué párrafo o párrafos el texto responde esta pregunta?

_____

_____

B. Señale en qué partes del texto se mencionan estos temas:

Elementos que componen el software

Elementos que componen el hardware

EJERCICIO 2

A. ¿A qué se refieren las siguientes palabras en el texto? Elija la opción correcta.

a. **These (l.12):** 1. Adapters, modems, switches and routers.

2. microprocessors, memory chips, data storage devices, input and output devices and networking components.

b. **This (l.13**): 1. To integrate these components into computer and network systems.

2. A good working knowledge of electrical engineering.

B. ¿Qué traducción le daría a cada referente?

_____

EJERCICIO 3

A. Formación de palabras: Las siguientes palabras extraídas del texto tienen en común la presencia de …….

processors – rewritable – storage – optical – development – user

Complete el cuadro:

	Palabra raíz	**Afijo**
nonvolatile		
rewritable		
storage		
optical		
electrical		
development		
management		
authentication		

¿Qué cambio introducen los prefijos?

_____

¿Qué cambios introducen los sufijos?

_____

B. Encuentre en el texto palabras que terminan en estos sufijos:

-OR

¿A qué categoría corresponden?

_____

¿De qué categoría provienen?

_____

-ER

C. Elija seis palabras que terminan en el sufijo –ing y transcríbalas. ¿Se pueden traducir como *verbos* en algún caso?

D. ¿En qué se diferencian estos pares de palabras? ¿Qué tienen en común?

Engineer / engineering **Drive / driver**

_____

_____

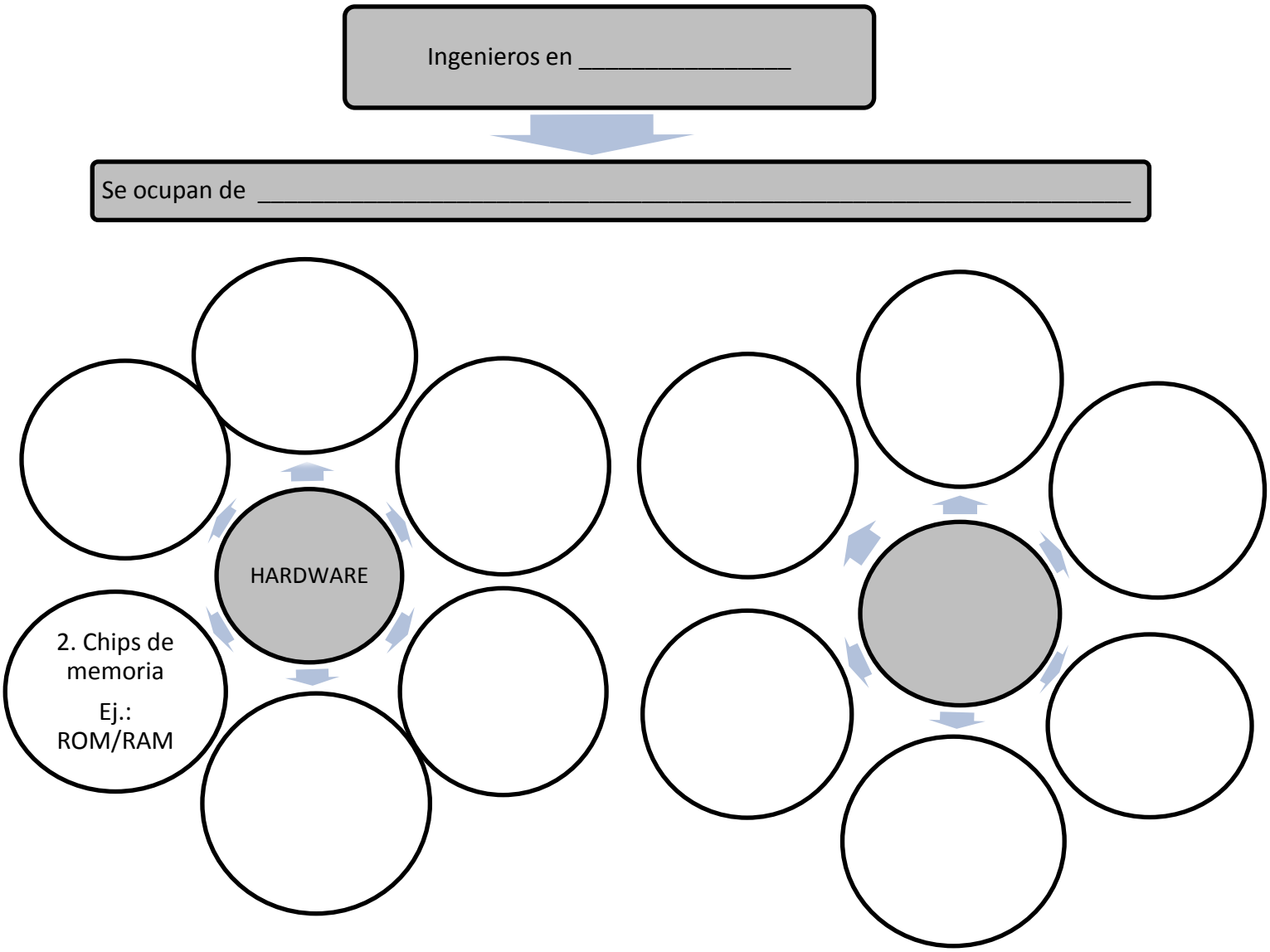
EJERCICIO 4

Subraye el núcleo o núcleos de las siguientes frases nominales y tradúzcalas.

a. Computer systems ……
b. Memory devices ……..
c. Hard disks ……
d. Nonvolatile rewritable flash memory ……
e. Audio devices and remote controls ……
f. A function of computer engineers ……
g. Another important aspect of computer engineering ……

EJERCICIO 5

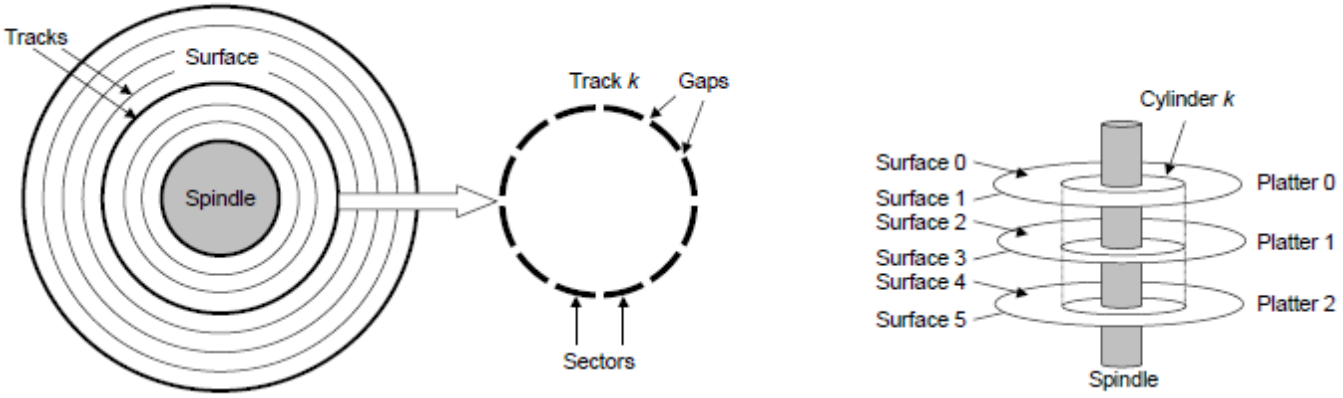
Lea el texto con atención y complete este resumen esquemático en español.



DISK GEOMETRY

Disks are constructed from *platters*. Each platter consists of two sides, or *surfaces*, **that** are coated with magnetic recording material. A rotating *spindle* in the center of the platter spins the platter at a fixed *rotational rate*, typically between 5400 and 15,000 *revolutions per minute (RPM)*. A disk will typically contain one or more of **these platters** encased in a sealed container.

Figure 6.9 (a) shows the geometry of a typical disk surface. Each surface consists of a collection of concentric rings called *tracks*. Each track is partitioned into a collection of *sectors*. Each sector contains an equal number of data bits (typically 512 bytes) encoded in the magnetic material on the sector. Sectors are separated by *gaps* where no data bits are stored. Gaps store formatting bits that identify sectors.



(a) Single-platter view. (b) Multiple-platter view.

Figure 6.9: **Disk geometry.**

A disk consists of one or more platters stacked on top of each other and encased in a sealed package, as shown in Figure 6.9 (b). The entire assembly is often referred to as a *disk drive*, although we will usually refer to **it** as simply a *disk*. We will sometime refer to disks as *rotating disks* to distinguish **them** from flash-based *solid state disks (SSDs)*, which have no moving parts.

Disk manufacturers describe the geometry of multiple-platter drives in terms of *cylinders*, where a cylinder is the collection of tracks on all the surfaces that are equidistant from the center of the spindle. For example, if a drive has three platters and six surfaces, and the tracks on each surface are numbered consistently, then cylinder k is the collection of the six instances of track k.

DISK OPERATION

Disks read and write bits stored on the magnetic surface using a *read/write head* connected to the end of an *actuator arm*, as shown in Figure 6.10 (a). By moving the arm back and forth along its radial axis, the drive can position the head over any track on the surface. This mechanical motion is known as a *seek*. Once the head is positioned over the desired track, then as each bit on the track passes underneath, the head can either sense the value of the bit (read the bit) or alter the value of the bit (write the bit). Disks with multiple platters have a separate read/write head for each surface, as shown In Figure 6.10 (b). The heads are lined up vertically and move in unison. At any point in time, all heads are positioned on the same cylinder.

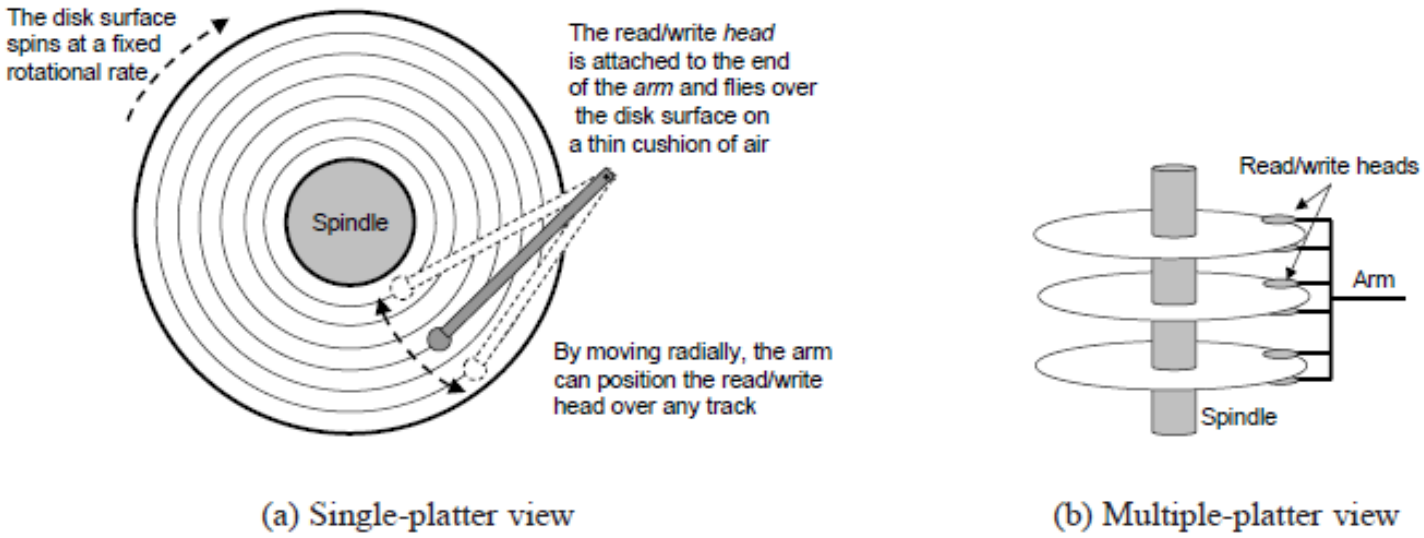


Figure 6.10: **Disk dynamics.**

EJERCICIO 1

A. Antes de leer el texto preste atención a los dos títulos, ¿sobre qué cree que tratará cada parte? ¿Qué otro elemento nos permite anticipar el contenido del texto?

_____

_____

_____

B. Indique en cuál de las dos partes del texto cree que se encontrarán estas ideas. Una con flechas.

	Definición de cilindro.	
	Movimientos del cabezal.	
DISK GEOMETRY	Concepto de pistas y sectores.	DISK OPERATION
	Composición física de un disco.	
	Mecanismos de lectura y escritura en el disco.	

C. Ahora lea todo el texto y corrija o confirme lo anterior.

EJERCICIO 2

A. Ubique estas palabras junto a sus equivalentes en inglés en las Figuras 6.9 y 6.10:

sectores - brazo - espacios – eje – superficie – cabezales de lectura y escritura – pista – platillo - cilindro

B. ¿Qué dos vistas nos presenta cada figura?

_____ _____

Disk Geometry

EJERCICIO 3

A. Elija la o las traducciones correctas para las siguientes oraciones extraídas del texto. ¿En qué voz y en qué tiempo se encuentran los verbos?

1. Disks are constructed from platters

- Los discos están compuestos de platillos.
- Los discos conforman platillos.
- Los discos se componen de platillos.

2. Each track is partitioned into a collection of sectors

- Cada pista divide al grupo en sectores.
- Cada pista se divide en un grupo de sectores.
- Cada pista está dividida en un grupo de sectores.

B. Busque en el texto otros ejemplos con esta misma estructura. ¿Cuál sería la versión en español para cada uno?

…….

…….

…….

…….

…….

…….

EJERCICIO 4

Lea el texto detenidamente y elija la opción correcta para las palabras resaltadas en negrita.

1. ¿A qué refiere "that"?
a. disks
b. platters
c. two sides/surfaces

2. ¿A qué refiere "these platters"?
a. surfaces
b. platters
c. sides

3. ¿A qué refiere "it"?
a. disk drive
b. one or more platters
c. disk

4. ¿A qué refiere "them"?
a. rotating disks
b. flash-based solid state disks
c. moving parts

EJERCICIO 5

Las siguientes frases nominales fueron extraídas del texto. Elija la traducción correcta.

1. A rotating spindle in the center of the platter

a. un eje giratorio en el centro del platillo

b. un platillo en el centro del eje giratorio

2. a fixed rotational rate

a. una velocidad de rotación fija

b. una rotación de velocidad fija

3. rotating disks

a. rotación de discos

b. discos giratorios

4. flash-based solid state disks

a. memoria flash basada en los discos de estado sólido

b. SSD basados en memoria flash

5. the collection of tracks on all the surfaces

a. todas las superficies en el conjunto de pistas

b. el conjunto de pistas en todas las superficies

EJERCICIO 6

A. ¿Qué cuatro elementos <u>se definen</u> en el texto? Complete los cuadros con las definiciones en español.

1. _____	***2.*** _____
3. _____	***4.*** _____

B. Ahora indique en español <u>la función</u> de estos otros elementos:

SPINDLE **SECTORS** **GAPS**

……..
……..
……..
……..
……..
……..

Disk Operation

<u>EJERCICIO 7</u>

Lea la segunda parte del texto y <u>explique</u> lo siguiente:

a. En qué consiste una búsqueda.
……..
……..

b. En qué procesos interviene un cabezal y cómo lo hace.
……..
……..

c. Qué caracteriza a los discos con platillos múltiples.
……..
……..

d. Si la Figura 6.10 agrega información a la ya presentada en el texto.
……..
……..

Fundamental Data Types

An abstract data type is a collection of well-defined operations that can be performed on a particular structure. The operations define what the data type does, but not how **it** works. Abstract data types are black boxes that we dare not open when we design algorithms that use **them**.

For each of the most important abstract data types, several competing implementations, or *data structures*, are available. Often, alternative data structures realize different design tradeoffs that make certain operations (say, insertion) faster at the cost of other operations (say, search). In some applications, certain data structures yield simpler code than other implementations of a given abstract data type, or have some other specialized advantages.

Specialized Data Structures

The basic data structures thus far described all represent an unstructured set of items so as to facilitate retrieval operations. **These data structures** are well known to most programmers. Not as well known are high-powered data structures for representing more structured or specialized kinds of objects, such as points in space, strings, and graphs.

The design principles of these data structures are the same as for basic objects. There exists a set of basic operations we need to perform repeatedly. We seek a data structure that supports these operations very efficiently. These efficient, specialized data structures are as important for efficient graph and geometric algorithms as lists and arrays are for basic algorithms, so one should be aware of **their** existence.

- *String data structures* - Character strings are typically represented by arrays of characters, with perhaps a special character to mark the end of the string. Suffix trees/arrays are special data structures **that** preprocess strings to make pattern matching operations faster;
- *Geometric data structures* - Geometric data typically consists of collections of data points and regions. Regions in the plane are usually described by polygons, where the boundary of the polygon is given by a chain of line segments. Polygons can be represented using an array of points $(v_1, \ldots, v_n, v_1)$, such that (v_i, v_{i+1}) is a segment of the boundary. Spatial data structures such as *kd*-trees organize points and regions by geometric location to support fast search;
- *Graph data structures* - Graphs are typically represented by either adjacency matrices or adjacency lists. The choice of representation can have a substantial impact on the design of the resulting graph algorithms, as discussed in Chapter 3. Implementation aspects of graph data structures are presented in Section 5;
- *Set data structures* - Subsets of items are typically represented using a dictionary, to support fast membership queries.

(from: The Algorithm Design Manual by Steven S. Skiena, Department of Computer Science State University of New York, 1997)

EJERCICIO 1

A. ¿En cuántos párrafos podríamos dividir este texto? Márquelos con llaves.

B. ¿En cuál de estos párrafos podemos encontrar la respuesta a las siguientes preguntas? Indique el número de párrafo.

1. What is the difference between specialized data structures and basic data structures?

Párrafo ___

2. How are certain data structures usually represented? Párrafo___

3. What do we understand by an *abstract data type*? Párrafo ___

4. What´s the relationship between abstract data types and data structures? Párrafo ___

5. Why are basic objects and certain data structures related? Párrafo ___

EJERCICIO 2

Una las palabras resaltadas en cada frase con la idea o concepto a la cual hacen referencia

a.… but not how **it** works… (line 2) ……*data type*……..……

b.…when we design algorithms that use **them**…(line 3) ……..…….……

c. **These data structures**… (line 11) ……..……

d.…so one should be aware of **their** existence…(line 17) ……..……..

e. …special data structures **that** preprocess strings…(line 21) ……

1. these efficient, specialized data structures
2. data type
3. special data structures
4. abstract data types
5. the basic data structures

Para asegurarse de que los antecedentes sean correctos, traduzca la frase completa como en el ejemplo.

Ej: a. …pero no cómo funciona el tipo de dato….

b.……..

c.…….

d.……..

e.……..

EJERCICIO 3

Busque en el diccionario el significado de las palabras resaltadas y luego escriba la traducción de las siguientes frases del texto que las contienen.

a.… we **dare** not open…:……..……

b. different design **tradeoffs**…:……

c. The basic data structures **thus far** described…:……..……

d.…the **boundary** of the polygon… :……..……

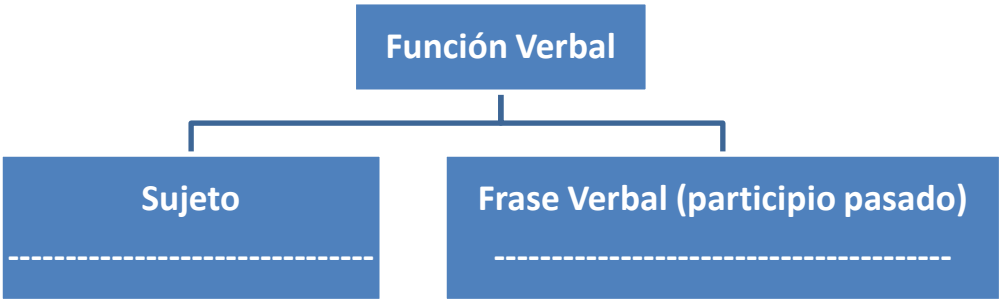
e.…**either** adjacency matrices **or** adjacency lists…:……

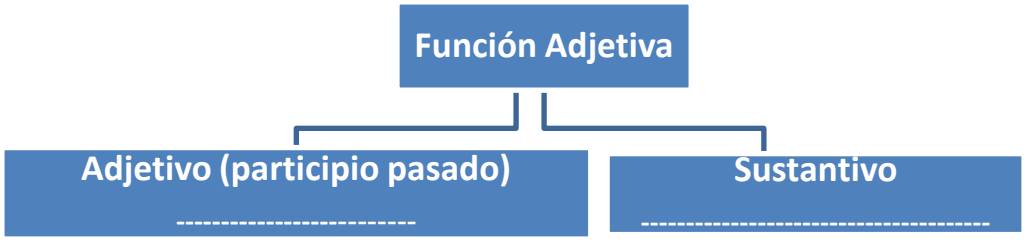
EJERCICIO 4

A. Miremos las palabras subrayadas en los párrafos 1 y 2. ¿Qué clase de palabras son?

- **¿Cuál de ellas funciona como verbo? _____**
- **¿Y el resto? _____**

B. Ahora analicemos los ejemplos anteriores en estos esquemas:





C. Busque más ejemplos en el resto del texto y decida si están funcionando como <u>verbos</u> o como <u>adjetivos</u>. Puede agregar más burbujas a los esquemas anteriores.

EJERCICIO 5

Encuentre en el texto las respuestas para las siguientes preguntas. Tradúzcalas.

1. ¿Para qué se puede utilizar un carácter especial en los arreglos de cadenas de caracteres?

……

2. ¿Para qué preprocesan cadenas algunas estructuras como el árbol de sufijos?

……

3. ¿Para qué organizan puntos y regiones mediante ubicación geométrica las estructuras de datos espaciales?

……

4. ¿Para qué se utiliza el diccionario?

……

- **¿Qué tienen en común las respuestas?**

  ……

  ……

- **Compare las respuestas encontradas con la siguiente frase del texto:**

  *There exists a set of basic operations we need **to perform** repeatedly.*

¿Cuál es la similitud y/o diferencia entre este último ejemplo y los anteriores?

……

……

EJERCICIO 6

A. Corrija estas ideas con información del texto:

1. Un tipo de datos abstracto está compuesto de operaciones que indican la forma en que el tipo de datos funciona.

……

……

2. Existe una estructura de datos específica para cada tipo de datos abstracto.

……

……

3. Las estructuras de datos básicas y las más complejas representan objetos similares.

……

……

4. Los principios de diseño de ambas clases de estructuras difieren enormemente.

……

……

5. Los algoritmos básicos necesitan de estructuras de datos especializadas.

……

……

B. Ahora complete el siguiente esquema con información sobre las diferentes estructuras de datos:

TIPO DE ESTRUCTURA DE DATOS	⟹	**FORMA DE REPRESENTACIÓN**
……..	⟹	……..……
…….	⟹	……..……
….....……..……	⟹	……..……
……..……..	⟹	…….. ……..

1.5 Typical roles and career path for database professionals

Like any other work profile, the database domain has several roles and career paths associated with it. The following is a description of some of **these roles.**

1.5.1 Data Architect

A data architect is responsible *for designing* an architecture **that** supports the organization's *existing* and future needs for data management. The architecture should cover databases, data integration and the means to get to the data. Usually the data architect achieves **his** goals *by setting* enterprise data standards. A Data Architect is also referred to as a Data Modeler. **This** is in spite of the fact that the role involves much more than just *creating* data models.
Some fundamental skills of a Data Architect are:

- Logical Data *modeling*
- Physical Data *modeling*
- Development of a data strategy and associated policies
- Selection of capabilities and systems to meet business information needs

1.5.2 Database Architect

This role is similar to a Data Architect, though constraints more towards a database solution. A database architect is responsible for the *following* activities:

- Gather and document requirements from business users and management and address **them** in a solution architecture.
- Share the architecture with business users and management.
- Create and enforce database and application development standards and processes.
- Create and enforce service level agreements (SLAs) for the business, specially addressing high availability, backup/restore and security.
- Study new products, versions compatibility, and deployment feasibility and give recommendations to development teams and management.
- Understand hardware, operating system, database system, multi-tier component architecture and interaction between **these components.**
- Prepare high-level documents in-line with requirements.
- Review detailed designs and implementation details.

It is critical for a database architect to keep pace with the various tools, database products, hardware platforms and operating systems from different vendors as they evolve and improve.

1.5.3 Database Administrator (DBA)

A database administrator (DBA) is responsible for the maintenance, performance, integrity and security of a database. Additional role requirements are likely to include *planning*, development and *troubleshooting.*
The work of a database administrator (DBA) varies according to the nature of the *employing* organization and the level of responsibility associated with the post. The work may be pure maintenance or it may also involve *specializing* in database development. Typical responsibilities include some or all of the following:

- *Establishing* the needs of users and monitoring user access and security;
- *Monitoring* performance and managing parameters to provide fast query responses to front-end users;
- *Mapping* out the conceptual design for a planned database in outline;
- Take into account both, back-end organization of data and front-end accessibility for end users;
- *Refining* the logical design so that **it** can be translated into a specific data model;

- Further *refining* the physical design to meet system storage requirements;
- *Installing and testing* new versions of the database management system (DBMS);
- *Maintaining* data standards, *including* adherence to the Data Protection Act;
- *Writing* database documentation, including data standards, procedures and definitions for the data dictionary (metadata);
- *Controlling* access permissions and privileges*;*
- *Developing, managing and testing* backup and recovery plans;
- *Ensuring* that storage, *archiving*, backup and recovery procedures are *functioning* correctly;
- Capacity *planning;*
- *Communicating* regularly with technical, applications and operational staff to ensure database integrity and security;
- *Commissioning and installing* new applications.

Because of the *increasing* levels of hacking and the sensitive nature of data stored, security and recoverability or disaster recovery have become increasingly important aspects.

(From: Database Fundamentals by Neera Sharma. Canadá, 2010)

EJERCICIO 1

A. Lea los títulos de este texto. En el primero aparece la expresión "*profesionales de bases de datos", ¿*qué profesionales se mencionan en los otros títulos?

_____

B. Explorando el texto sin leerlo en detalle, indique cuáles de estos aspectos se describen para cada uno de los profesionales:

- Lugar de trabajo
- Tareas a cargo
- Formación profesional
- Habilidades que deben poseer
- Relación con el avance tecnológico de software y hardware
- Sueldos aproximados

EJERCICIO 2

Lea el texto hasta 1.5.2 inclusive y marque V o F. Corrija las ideas falsas.

1. El texto describirá los diferentes avances tecnológicos del campo de las bases de datos.

2. "Modelador de datos" es otro nombre que recibe el arquitecto de datos.

3. El trabajo de arquitecto de datos consiste solo en la creación de modelos de datos.

4. Trabajar como arquitecto de base de datos implica tener en cuenta a usuarios y administradores.

5. El arquitecto de base de datos debe estar al tanto de los últimos productos de software y hardware.

……....

……..……

……...

……..

EJERCICIO 3

A. Explique de qué estamos hablando en cada caso.

- ✓ some of "**these roles**", ¿qué roles?
  ……...
- ✓ "**This**" is in spite of … ¿qué cosa es así a pesar de que el rol involucra mucho más que crear modelos de datos?
  …….
- ✓ "**This role**" is similar to …, ¿cuál rol?
  ……
- ✓ Interaction between "**these components**", ¿qué componentes?
  ……..

B. Elija la opción correcta para decir a qué se refieren las siguientes palabras y frases en el texto.

1. **that**: a. an architecture
   b. the organization's …needs
2. **his** (goals): a. data standards
   b. the data architect
3. **them**: a. requirements
   b. business users
4. **it** a. a specific data model
   b. the logical design

C. En la siguiente oración:

It *is critical for a database architect to keep pace with the various tools, database products, hardware platforms and operating systems from different vendors as they evolve and improve*

¿Qué diferencia al "It" de los otros referentes analizados en el texto?

……..

¿Qué es importante para un arquitecto de BD?

……..

EJERCICIO 4

Lea *Data Architect* y complete con información del texto prestando especial atención a las palabras subrayadas.

1. El arquitecto de datos se ocupa de ……

……..

2. La arquitectura que diseña es compatible con las necesidades ……

……..

3. Normalmente logra sus objetivos ……...

4. Su rol involucra mucho más que ……..…….

5. Entre las habilidades esenciales de un arquitecto de datos se encuentran el ……..

……..

¿Se pudo traducir alguna forma en –ing como una palabra en –ando o –endo (gerundio español)?

EJERCICIO 5

Lea *Database Architect* y responda lo siguiente:

A. ¿Qué hace el arquitecto de base de datos con …

… los requerimientos de los usuarios y administradores? _____

… la arquitectura que diseña? _____

…los acuerdos sobre el nivel del servicio? _____

B. ¿Qué cosa el arquitecto de base de datos…

…debe estudiar? _____

…debe comprender? _____

…debe preparar? _____

… debe revisar? _____

EJERCICIO 6:

Lea *Database Administrator* y marque en el texto las líneas en que se desarrollan estas ideas:

a. Aspectos relacionados con la seguridad de los datos.
b. Responsabilidades del ABD.
c. Factores que determinan el tipo de trabajo de un ABD.

EJERCICIO 7:

A. Lea el texto y preste atención a las palabras subrayadas. ¿Cuál de ellas es un verbo que describe una acción en progreso?

_____

B. Elija la opción correcta en español para estas frases nominales según su uso en el texto.

1. the nature of the employing organization:

- la naturaleza de emplear una organización
- la naturaleza de la organización empleadora

2. capacity planning:

- planeamiento de la capacidad
- capacidad de planeamiento

3.Refining the logical design:

- Refinando el diseño lógico
- Refinar el diseño lógico

C. Ahora encuentre en el texto algunos ejemplos similares, transcríbalos y tradúzcalos.

Ejemplos en el texto	***Versión en español***

D. Complete con información del texto.

- Los factores que determinan la clase de trabajo que hará el ABD son ……
……..
- La seguridad de un sistema informático es importante debido a ……
……..

1.2 _____

While a database is a repository of data, a ***database management system***, or simply DBMS, is a set of software tools that control access, organize, store, manage, retrieve and maintain data in a database. In practical use, the terms database, database server, database system, data server, and database management systems are often used interchangeably.
Why do we need database software or a DBMS? Can we not just store data in simple text files for example? The answer lies in the way users access the data and the handling of corresponding challenges. First, we need the ability to have multiple users insert, update and delete data to the same data file without "stepping on each other's toes". We also need to have a standard interface for data access, tools for data backup, data restore and recovery, and a way to handle other challenges such as the capability to work with huge volumes of data and users. Database software has been designed to handle all of these challenges.

The most mature database systems in production are relational database management systems (RDBMS's). RDBMS's serve as the backbone of applications in many industries including banking, transportation, health, and so on.

1.2.1 _____

In the 1960s, network and hierarchical systems such as CODASYL and IMSTM **were** the state-of-the-art technology for automated banking, accounting, and order processing systems enabled by the introduction of commercial mainframe computers. While these systems **provided** a good basis for the early systems, their basic architecture **mixed** the physical manipulation of data with its logical manipulation. When the physical location of data changed, such as from one area of a disk to another, applications had to be updated to reference the new location.
A revolutionary paper by E.F. Codd, an IBM San Jose Research Laboratory employee in 1970, **changed** all that. The paper titled *"A relational model of data for large shared data banks"* **introduced** the notion of data independence, which separated the physical representation of data from the logical representation presented to applications. Data could be moved from one part of the disk to another or stored in a different format without causing applications to be rewritten. Application developers were freed from the tedious physical details of data manipulation, and could focus instead on the logical manipulation of data in the context of their specific application.

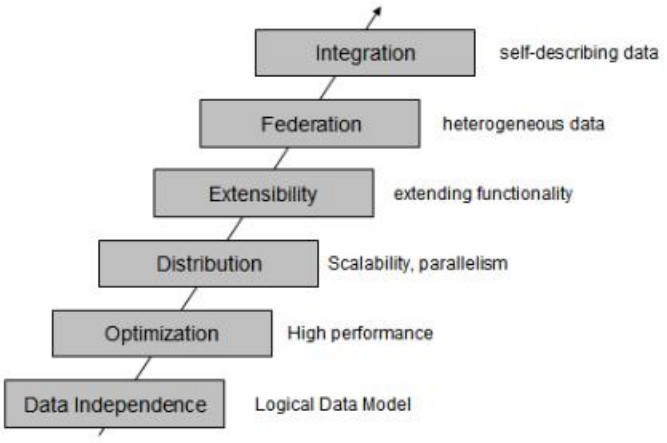


Figure 1.1 Evolution of database management systems

IBM's System R was the first system to implement Codd's ideas. System R was the basis for SQL/DS, which later became DB2. It also had the merit to introduce SQL, a relational database language used as a standard today, and to open the door for commercial database management

systems. Originally, the language was termed “Structured English Query Language” or SEQUEL, but it was later changed to SQL as SEQUEL was a registered trademark of a UK based company. SQL was adopted as a standard language in 1986 by the ***American National Standards Institute*** (***ANSI***) and by the ***International Standards Organization*** (***ISO***) in 1987. Since its standardization, SQL standards have been updated six times. The last update was in 2008 and is popularly referred as SQL:2008.

Today, relational database management systems are the most used DBMS's and are developed by several software companies. IBM is one of the leaders in the market with DB2 database server. Other relational DBMS's include Oracle, Microsoft SQL Server, INGRES, PostgreSQL, MySQL, and dBASE.

(from: Database Fundamentals by Neera Sharma. Canadá, 2010)

EJERCICIO 1

Antes de leer el texto.

Estos son los dos títulos del texto anterior:

The evolution of database management systems

What is a database management system?

- ✓ **¿Qué información espera que contenga cada parte?**

……

……

- ✓ **¿En qué tiempo estarán los verbos en cada una? ¿En presente, pasado o futuro?**

……

……

EJERCICIO 2

Lea el texto, corrija o confirme lo anterior y coloque los títulos donde corresponda.

EJERCICIO 3

Lea el punto 1.2 e indique en qué parte el texto presenta lo siguiente:

1. Una explicación	¿Sobre qué?
2. Definiciones	¿De qué?
3. Información específica	¿Sobre qué?
4. Sinónimos de un término	¿De cuál?

EJERCICIO 4

Busque la siguiente información en el texto y desarróllela en español.

a. Ámbitos en que se usan las bases de datos relacionales: _____

_____

b. Términos que se utilizan en forma alternativa: _____

_____

c. Funciones del sistema de gestión de bases de datos: _____

_____

d. Algunos desafíos que el software de base de datos puede manejar: _____

_____

_____

EJERCICIO 5

Lea el punto 1.2.1 y agrupe esta información de acuerdo a las fechas en la burbuja correspondiente.

Década de 1960

1970

Década de 1980

Uso habitual de sistemas relacionales.
Concepto revolucionario de independencia de los datos.
Estandarización del lenguaje SQL.
Sistemas jerárquicos y de redes.
Creación del lenguaje relacional SQL.
Mayor libertad para los desarrolladores de aplicaciones.
Mezcla de manipulación lógica y física de la información.

2010

Ahora responda:

- ✓ **¿Qué elementos nos indican que el texto trata sobre un tiempo pasado?**

_____

- ✓ **¿Qué palabra nos trae nuevamente al presente?**

_____

EJERCICIO 6

Preste atención a los verbos en pasado subrayados en el texto y elija en esta síntesis la opción correcta.

En 1960 CODASYL e IMSTM **fueron / eran** la tecnología de vanguardia, **ofrecieron / ofrecían** una buena base para los primeros sistemas pero **mezclaron / mezclaban** la manipulación física y lógica de los datos.

Pero en 1970 el trabajo de Codd **cambió / cambiaba** todo esto e **introdujo / introducía** un concepto revolucionario.

- ✓ **¿Qué diferencia hay entre una forma de pasado y la otra?**

_____

EJERCICIO 7

Lea el texto nuevamente y explique lo siguiente:

a. Antes del modelo relacional, ¿qué sucedía ante un cambio de ubicación física de los datos?

……

……

b. ¿Qué cambio introdujo la noción de independencia de los datos? ¿Qué se podía hacer con ellos?

……

……

……

c. ¿Qué significó este cambio para los desarrolladores de aplicaciones?

……

……

d. ¿Qué tres datos importantes nos da el texto sobre el Sistema R?

……

……

e. ¿Qué sucedió con el nombre del lenguaje SQL?

……

……

……

f. ¿Qué sucedió en 1986 y 1987?

……

……

EJERCICIO 8

Complete con información específica sobre lo siguiente:

a. CODASYL e IMSTM se utilizaban en …….....

b. E.F. Codd era un ……

c. SQL es …….

d. Oracle, INGRES, y dBASE, entre otras son ejemplos de ……..

EJERCICIO 9

Luego de haber leído todo el texto, que es parte de un texto más largo, observe la figura que lo acompaña. ¿Qué etapa o etapas de esta figura se describen en el texto?

_____

Chapter 6
The Memory Hierarchy

To this point in our study of systems, we have relied on a simple model of a computer system as a CPU that executes instructions and a memory system that holds instructions and data for the CPU. In our simple model, the memory system is a linear array of bytes, and the CPU can access each memory location in a constant amount of time. While this is an effective model as far as it goes, **it** does not reflect the way that modern systems really work.

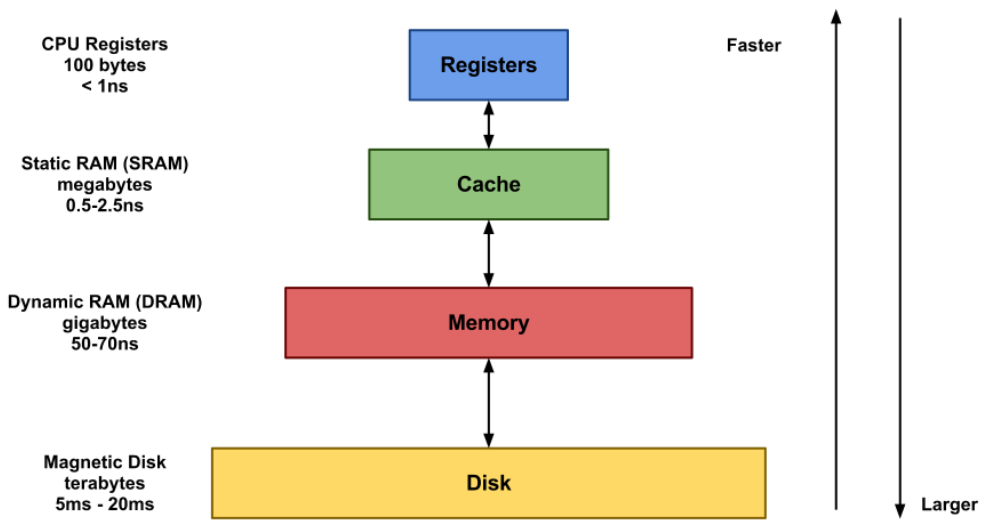
In practice, a memory system is a hierarchy of storage devices with different capacities, costs, and access times. CPU registers hold the most frequently used data. Small, fast *cache memories* nearby the CPU act as staging areas for a subset of the data and instructions stored in the relatively slow main memory. The main memory stages data stored on large, slow disks, which in turn often serve as staging areas for data stored on the disks or tapes of other machines connected by networks.

Memory hierarchies work because well-written programs tend to access the storage at any particular level more frequently than **they** access the storage at the next lower level. So the storage at the next level can be slower, and thus larger and cheaper per bit. The overall effect is a large pool of memory that costs as much as the cheap storage near the bottom of the hierarchy, but that serves data to programs at the rate of the fast storage near the top of the hierarchy.

As a programmer, you need to understand the memory hierarchy because it has a big impact on the performance of your applications. If the data your program needs are stored in a CPU register, then they can be accessed in zero cycles during the execution of the instruction. If stored in a cache, 1 to 30 cycles. If stored in main memory, 50 to 200 cycles. And if stored in disk tens of millions of cycles!

Here, then, is a fundamental and enduring idea in computer systems: If you understand how the system moves data up and down the memory hierarchy, then you can write your application programs so that **their** data items are stored higher in the hierarchy, where the CPU can access **them** more quickly.

This idea centers around a fundamental property of computer programs known as locality. Programs with good locality tend to access the same set of data items over and over again, or they tend to access sets of nearby data items. Programs with good locality tend to access more data items from the upper levels of the memory hierarchy than programs with poor locality, and thus run faster. For example, the running times of different matrix multiplication kernels that perform the same number of arithmetic operations, but have different degrees of locality, can vary by a factor of 20!



It can be noted that the larger the memory, the slower the data access.

6.1 Storage Technologies

Much of the success of computer technology stems from the tremendous progress in storage technology. Early computers had a few kilobytes of random-access memory. The earliest IBM PCs did not even have a hard disk. **That** changed with the introduction of the IBM PC-XT in 1982, with its 10-megabyte disk. By the year 2010, typical machines had 150,000 times as much disk storage, and the amount of storage was increasing by a factor of 2 every couple of years.
Glosario: To stage: *montar / manipular*

EJERCICIO 1

A. Lea el texto completo con atención e indique en qué párrafo se mencionan las siguientes ideas (Parte 6):

- Ciclos requeridos para acceder a la información _____
- Rol de la memoria principal. _____
- Importancia de la *localidad*. _____

B. Marque con un ✓ los conceptos que están definidos en el texto.

- ***Computer system*** ___
- ***Memory system*** ___
- ***Caché memory*** ___
- ***Locality*** ___
- ***Storage devices*** ___

EJERCICIO 2

Lea el texto nuevamente y decida si las siguientes afirmaciones son correctas.

a. Toda la información que es frecuentemente usada se encuentra en los registros de la CPU. SI / NO

b. La primera Computadora Personal IBM consistía en un disco duro. SI / NO

c. Para acceder a los datos almacenados en un disco, se necesitan más de un millón de ciclos. SI / NO

d. Un programador debe estar familiarizado con la jerarquía de memoria. SI / NO

e. Los adelantos en la tecnología de almacenamiento han beneficiado el desarrollo de la tecnología de las computadoras. SI / NO

EJERCICIO 3

Encierre en un círculo a qué refieren las palabras resaltadas en el texto.

It	CPU / model
they	hierarchies / programs
their	application programs / data items
them	application programs / data items
That	a few kilobytes of ….. memory / The earliest IBM PCs did not …a disk

EJERCICIO 4

Las expresiones subrayadas a lo largo del texto tienen en común que expresan:

A. ¿Podría agruparlas? Complete.

Comparación de adjetivos:
……
……

Comparación de adverbios:
……
……

Adjetivos superlativos:
……
……

Otras formas de comparación:
……
……

B. Ahora traduzca las frases completas al español.

1. the most frequently used data

……..

2. programs tend to access the storage at any particular level more frequently than….

……..

3. the next lower level

……..

4. the storage at the next level can be slower, and thus larger and cheaper per bit…

……..

5. their data items are stored higher in the hierarchy

……..

6. the CPU can access them more quickly

……..

7. Programs with good locality tend to access more *data** items from the upper levels of the memory hierarchy than programs with poor locality, and thus run faster.

……

……

8. The earliest IBM PCs

……

C. ¿Qué indica more en *more data* *? ¿Qué valor tiene?

_____

EJERCICIO 5

Complete brevemente con información del texto.

En la actualidad, la manera en que funcionan los sistemas modernos no ……..
……
……

Los programas con buena localidad son más rápidos debido a que ……...
……
……

A partir de 1982, con la PC de IBM ……
……
……

EJERCICIO 6

¿Causa o consecuencia?
Complete las oraciones de la grilla con información del texto. Luego analice

C……	**C……**
…… ……	*puede acceder a la información en tus programas sin usar ciclos.*
……. …….	…….. *necesitará de 0 a 15 ciclos.*
Si la información está guardada en la memoria principal,	entonces ……... ……
……..…… …….	…… *millones de ciclos.*

EJERCICIO 7

Observe nuevamente la figura.

a. ¿Es posible comprender la figura sin leer el texto?

……
……

b. Explique qué información suministra la pirámide.

……
……
……

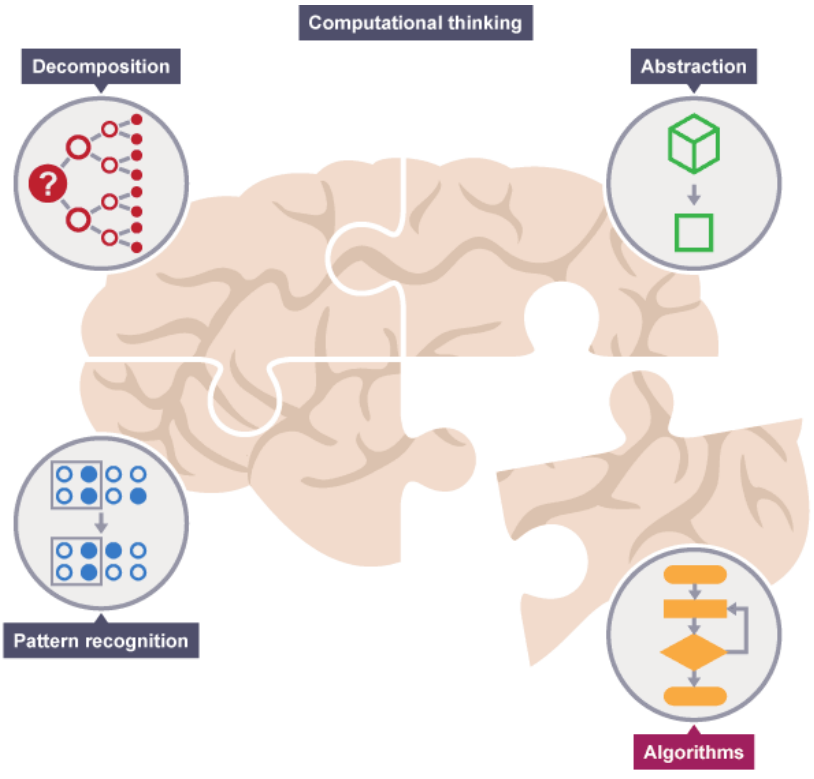
ALGORITHMS

When learning your first programming language, it is easy to get the impression that the hardest part of solving a problem on a computer is translating your ideas into the specific language that will be fed into the computer. This definitely is not the case. The most difficult part of solving a problem on a computer is coming up with the method of solution. After you have developed a method of solution, it is routine to translate your method into the required language, be it Pascal or some other programming language. When solving a problem with a computer, it is therefore helpful to temporarily ignore the computer programming language and to concentrate instead on formulating the steps of the solution and writing them down in plain English, as if the instructions were to be given to a human being. A set of instructions expressed in this way is frequently referred to as an "*algorithm*".

An algorithm is a plan, a set of step-by-step instructions that leads to a solution. Some approximately equivalent words are "recipe", "method", "directions", and "routine". If you can tie shoelaces, make a cup of tea or prepare a meal, then you already know how to follow an algorithm.

In an algorithm, each instruction is identified and the order in which they should be carried out is planned. An algorithm expressed in a language that a computer can understand is called a *program*, which explains why computer languages are called *programming languages.* Instructions may be expressed in a programming language or a human language, like English or French.

The word "algorithm" has a long history, but its meaning has recently taken on a new character]. The word itself derives from the name of the ninth-century Arabic mathematician and astronomer Al-Khowarizmi, who wrote an early and famous textbook on the manipulation of numbers and equations entitled Kitab al-jabr w'almuqabala. The similar-sounding word "algebra" was derived from the Arabic word "al-jabr", which appears in the title of the text and is often translated as "reuniting" or "restoring". The entire title can be translated as "Rules for Reuniting and Reducing". The meanings of the words "algebra" and "algorithm" used to be much more intimately related than they are now. Indeed, until very recently the word "algorithm" usually referred to algebraic rules for solving numeric equations.



Making a plan

It is important to plan out the solution to a problem to make sure that it will be correct. Using computational thinking and decomposition we can break down the problem into smaller parts and then we can plan out how they fit back together in a suitable order to solve the problem. This order can be represented as an algorithm. An algorithm must be clear. It must have a starting point, a finishing point and a set of clear instructions in between.

EJERCICIO 1

A. Lea el texto con atención y formule como mínimo dos preguntas que se puedan responder con el contenido de cada párrafo, como el ejemplo.

Párrafo 1: a. ¿……?

b. ¿……..?

Párrafo 2: a. *¿Qué es un algoritmo?*

b. ¿……..?

Párrafo 3: a. ¿……?

b. ¿……..?

Párrafo 4: a. ¿……?

b. ¿……..?

B. ¿Con qué párrafo/s se relaciona más la imagen y el epígrafe que la acompaña?

……..……

…….

EJERCICIO 2

Con la información de todo el texto, arme una definición bien completa de *algoritmo*.

Definición:

……...

…….

……..……

✓ **¿Qué otro término se define en el texto?**

……

EJERCICIO 3

Indique si estas ideas son correcta o no. Corrija con sus palabras las segundas, utilizando información del texto.

a. Traducir muestras ideas a un lenguaje de programación constituye la tarea más complicada de la resolución de un problema.

b. El texto nos sugiere formular primero el algoritmo en nuestra lengua y traducirlo luego al lenguaje de programación elegido.

c. Las instrucciones de un algoritmo no tienen un orden determinado.

d. Las palabras álgebra y algoritmo tienen un origen común.

e. La descomposición nos permite dividir un problema en partes menores y luego reordenarlas para lograr la solución.

EJERCICIO 4

Preste atención a estas oraciones extraídas del texto. ¿Qué tienen en común?

Estructura común	
	When learning your first programming language, …
	When solving a problem with a computer, …
	it is easy to get the impression that …
	it is routine to translate your method into the required language…
	it is therefore helpful to temporarily ignore the computer …
	It is important to plan out the solution to a problem …

- ✓ **Indique cuál sería el equivalente en español de cada estructura.**

a. ……

b. ……

- ✓ **Ahora observe los dos "it" subrayados en el texto. ¿Tienen un antecedente en el texto? ¿O tienen a la estructura del ejercicio anterior?**

EJERCICIO 5

Lea el texto nuevamente y responda.

1. ¿Por qué sería acertado decir que los algoritmos son parte de nuestra vida diaria?

……
……
……

2. ¿Quién fue y qué hizo Al-Khowarizmi?

……
……
……

3. ¿Qué cambio sufrió el significado de la palabra *algoritmo*?

……
……
……

Textos para Traducción

Texto A

The Challenge of Programming Language Design

A programming language is, by its very nature, a complex tool to design. The main source of this complexity stems from the fact that it must satisfy two diverse needs.
On the one hand, there is the need for a notation, which is natural for the programmer to use. On the other hand, the language must permit the programmer to create efficient programs. That implies that there must be adequate means for translating all possible uses of the language's notation into efficient machine code.

.....

.....

.....

.....

.....

.....

.....

.....

Texto B

Software

Software is a program that enables a computer to perform a specific task, as opposed to the physical components of the system (hardware). Computer software is so called in contrast to computer hardware, which encompasses the physical interconnections and devices required to store and execute the software.
Practical computer systems divide software into three major classes: system software, programming software and application software, although the distinction between them is arbitrary, and often blurred.

.....

.....

.....

.....

.....

.....

.....

.....

Texto C

The Microcomputer Program

Assembly language is important not only for controlling hardware devices of the microcomputer system, but also when performing pure software operations. For example, applications frequently require the microcomputer to search through a large table of data in memory, looking for a special string of characters,eg: person's name. This kind of operation can easily be performed by writing a program in a high-level language; however, for very large tables of data the search will take very long.

.....

.....

.....

.....

.....

.....

.....

.....

Texto D

CAD - computer-aided design

A CAD system is a combination of hardware and software that enables engineers and architects to design everything from furniture to airplanes. CAD systems allow an engineer to view a design from any angle with the push of a button and to zoom in or out for close-ups and long-distance views. In addition, the computer keeps track of design dependencies so that when the engineer changes one value, all other values that depend on it are automatically changed accordingly.

.....

.....

.....

.....

.....

.....

.....

.....

Texto E

Granular Content

A learning object can be as large as a year-long course of study in a subject, or as small as a single image file. It is important that institutional repositories have the means to describe their objects both at the highest level of granularity – the document level – and at the lowest level for each constituent part which, for a variety of reasons, requires independent description. Repository objects are, therefore, often hierarchies in themselves, sharing the character of archival records more than individual object catalogue records.

.....

.....

.....

.....

.....

.....

.....

.....

Texto F

Digital libraries in a digital world

In the past ten years, the concept of the 'digital library' (or the 'electronic library') has been increasingly used, and now crops up relentlessly in the professional literature. This is not surprising, as the combination of low-cost computing and high-speed networking now affects all areas of life in the developed world. 'Digital banking', 'online shopping' and 'digital television' are transforming the ways in which we transact our daily business and consume entertainment. We also book holidays online, gamble on the Internet and conduct hundreds of other activities online. (easy, to be one of the first).

.....

.....

.....

.....

.....

.....

.....

.....

Texto G

What role do institutional repositories play?

In pre-digital times, when researchers wrote up their results for publication, they would have been posted, hand-written or in typescript, to a publisher – the only agent with the technology to present the finished paper in a pleasing form, and to reproduce it in multiples sufficient to meet the likely demand across the world, in their journals. Publishers also managed a third very important process – that of verification that the research was of a quality which made it valuable to other researchers.

.....

.....

.....

.....

.....

.....

.....

.....

.....

Textos de Revisión

Texto de revisión 1

THE JOB OF THE COMPUTER PROGRAMMER

A computer programmer is first and foremost an interpreter. He has to break down a problem described in a natural language (such as English or Spanish) into logical steps and then translate the steps into a language understood by the computer.

What sort of problems do programmers have to deal with? Basically, three types. Firstly, scientific things like weather forecasting, statistical analysis, integration - anything related to sciences such as physics, biology, mathematics, astronomy, or the technologies used in industry.

Many scientific programmers work in universities or research centers. Some of them are scientists or technologists first and programmers second. They only know enough programming to solve their own particular problems. But it is different with commercial programmers. They may write programs to handle invoicing, or share registration, or stock control.

Systems programmers, instead, work for a computer manufacturer. They write software that acts as a buffer between scientific and commercial programs, and the machine.

But whatever type of program they write, scientific, commercial, or system software, all programmers have three main objectives. They must write programs that work. They must write them on time. And they must leave them fully documented so other programmers can easily take over, maintain and amend the programs

Texto de revisión 2

DATA FLOW SYMBOLS

A data flow is a path for data to move from one part of the information system to another. A data flow in a data flow diagram (DFD) represents one or more pieces of data. For example, a data flow could represent a single data item, such as a student identification (ID) number, or a data flow could represent a set of data, such as a class list with the students' ID numbers and names for a specific class. The DFD does not show the structure and detailed contents of a data flow, these elements are defined in the data dictionary.

The symbol for a data flow is a line with an arrowhead that shows the directions in which the data flows. The data flow name, which should identify the data it represents, is placed

above, below, or alongside the line. A data flow name consists of a singular noun and an adjective, if needed. Exceptions to a singular name rule are data flow names, where a singular name could mislead you to think a single parameter or single item of data exists.

Texto de revisión 3

PARALLEL COMPUTING SYSTEMS

A parallel computing system is a computer with more than one processor for parallel processing. In the past, each processor of a multiprocessing system always came in its own processor packaging, but recently introduced multicore processors contain multiple logical processors in a single package.

There are many different kinds of parallel computers. They are distinguished by the kind of interconnection between processors (known as "processing elements" or PEs) and memory. One major way to classify parallel computers is based on their memory architectures. Shared memory parallel computers have multiple processors accessing all available memory as global address space. They can be further divided into two main classes based on memory access times: Uniform memory access (UMA), in which access times to all parts of memory are equal, or Non-Uniform memory access (NUMA), in which they are not. Distributed memory parallel computers also have multiple processors, but each of the processors can only access its own local memory; no global memory address space exists across them

Parallel computing systems can also be categorized by the numbers of processors in them. Systems with thousands of such processors are known as massively parallel. Subsequently there are what are referred to as "Large scale" vs "Small scale" parallel processors. This depends on the size of the processor, e.g. a PC based parallel system would generally be considered a small scale system.

Parallel processor machines are also divided into symmetric and asymmetric multiprocessors, depending on whether all the processors are the same or not.

A variety of architectures have been developed for parallel processing. For example, a Ring architecture has processors linked by a ring structure. Other architectures include Hypercubes, Fat trees, systolic arrays, and so on.

- **Translate the following lists of noun phrases.**

✓ ***Exercise 1***

1. circuit board……..……..
2. storage controllers …..……
3. electronic components……..……....
4. semiconductor devices……...…..……...……...
5. high speed.....……
6. inkjet printers……..……..……
7. field attributes……
8. mouse sensitivity ……..…….……..
9. peripheral interfaces……..……..
10. physical and environmental events…….……..
11. hardware and software inventory…….……....
12. the user's particular needs…..……
13. two main advantages……..……
14. digital memory chips……...……
15. random access memory……..……
16. computer bus controllers……..……...
17. a single output operation……..……...
18. application program functions……..……....
19. memory access models……..
20. machine language instructions…….……....
21. dynamic memory allocation……..……....
22. a procedural programming paradigm……
23. source code file inclusion……..……....
24. electrical, electronic, or electro-mechanical device……...
25. your computer platform, operating system, programming skills, and application type ……..
    ……..
26. variables, arrays and complex arithmetic or boolean expressions……..……
    ……

✓ ***Exercise 2***

1. the movement of the pointer on the screen ..……
2. the first letter in the first word of the paragraph ..…….
3. a summary of the basic mouse techniques…….
4. a circular area 5 inches in diameter..……
5. motherboard or system board with slots for expansion cards…….....

6. the transfer of business data via intranets……
7. the differences between compilers and interpreters……...
8. languages for data analysis and statistics...…….
9. the load on very busy processors...…….....
10. a physical location on the screen.....…….....
11. external components of a computer system……..…….....
12. the abstractions of functions, variables and expression evaluation…….....

…….....

13. rudimentary support for modular programming……...
14. slots for expansion cards and holding parts…….....
15. spreadsheets with additional functions…….
16. the major difference between the two products……...
17. a great influence on many other popular languages…….....
18. a wide class of software and hardware implementations…….....
19. extensions of the underlying virtual memory architecture……..
20. an interesting demonstration of the remarkable interchangeability of pointers and arrays……..

…….....

✓ ***Exercise 3***

1. long-term storage…….....
2. medium-term storage……...
3. dial-up Internet access…….
4. non-physical item...…….....
5. a byte-code format……
6. in-depth knowledge …….....
7. non-sequential language ……
8. run-time support……
9. non-volatile storage……..
10. flat-panel displays……
11. fourth-generation languages ……..
12. first-time programmers..……..
13. Basic Input-Output System (BIOS) ……
14. read-only memory (ROM) ……
15. high-level programming language……..
16. a general-purpose, procedural, imperative computer programming language..……

…….

✓ ***Exercise 4***

1. compiled languages..……..
2. derived types……
3. designated initializers……..
4. a monolithic integrated circuit…….....
5. printed circuit board…….....
6. miniaturized electronic circuit…….....
7. the shared memory region ……..
8. distributed shared memory (DSM)…….
9. the uncompressed storage capacity of any medium…….....
10. four interrelated technologies…….
11. a few pre-defined tasks…….....
12. specialized systems for routine analyses…….
13. objects of unknown type……..
14. large blocks of data known as data streams…….
15. shared parts distributed among nodes and main memory…….
16. the prototypes of the functions contained within the library……...
17. any program written only in Standard C……
18. mainframe-based applications…….....
19. object-oriented language features..……..
20. the rubber-coated ball ..…….....
21. a graphics-based program…….
22. mass-produced consumer embedded systems……...
23. an implementation-defined search strategy…….....
24. the instructions provided in Chapters 1 through 3 in this guide. …….....

……..

25. problem-oriented languages...…….....
26. windows-based systems……..……..
27. built-in types for integers of various sizes…….
28. applications specifically targeted for Unix and Unix-like systems…….
29. any connected device added to the three base components.....……..

……..

30. any device attached to a computer in order to expand its functionality……...

✓ ***Exercise 5***

1. quicker text input……..
2. more practical uses ……..

3. more graphically oriented interfaces……
4. higher-level languages ……..…….
5. the most common type of removable media…….....
6. the most productive and efficient interface……..
7. the most widely used programming languages……...
8. the earliest forms of non-volatile read-only memory……..

✓ ***Exercise 6***

1. a blinking box……
2. rotating heads……..…….
3. a cooling fan…….....
4. string handling routines ……
5. several warning messages …….
6. floating-point numbers…….....
7. debugging purposes to preserve the application program screen contents……...

…….....

8. the resulting multidimensional array……..
9. computer training services ……..……...
10. links to a no-longer existing executable file…….
11. separate heads for recording and playback..……...
12. the cost of designing and developing a complex integrated circuit…….....

…….....

13. the value of evaluating the main function …….
14. facilities for managing memory……..……
15. a program for controlling, blocking and restricting internet & network access ……..

…….....

16. an exit code indicating successful execution…….

…….....

17. expressions including pointers…….....
18. new types using keywords……..
19. automated source code checking and auditing…….....

…….....

20. a thin membrane producing some proportional electrical signal ……..

…….....

21. a user working at a remote location……..
22. processing numerical data……..
23. organizing large programmes…….
24. "Using the mouse" Tutorial …….

25. copying of arrays or strings …….
26. using network resources through a logical segmentation of a single physical network. ……...

…….....

27. The problem-solving skills and tactics involved in writing or debugging software programs and applications ……....
28. Inexpensive C development packages resulting from advances in incremental compiling

...…….....

✓ ***Exercise 7***

1. a section of a program that performs a specific task

…….....

2. cards that are fast and built to last

…….....

3. examples of specific features that might be needed

…….

4. specific cost data that will change over time

…….....

5. an acoustic to electric transducer that converts sound into an electrical signal

…….....

6. a self-contained entity that consists of both data and procedures

…….....

7. software that can run on computers with a new architecture

…….....

8. applications that you have already deleted or uninstalled

…….....

9. requirements that would make the machine competitive in that market

…….....

10. more than just the electronic components that make up the computer itself

…….....

11. users who do not have a lot of experience with the computer

…….....

12. the types of operations that can be applied to the data structure

…….....

13. a highly educated and well trained staff that is dedicated to the computer market

…….....

14. a form of cathodoluminescent display that can operate at low voltages

…….....

15. management tools which are able to dynamically assign and efficiently manage workloads

…….....

16. a modem that accesses a private wireless data network or a wireless telephone system

…….....

17. the ways that software and technology in the cloud are accessed by digital media

…….....

18. any board that plugs into one of the computer's expansion slots.

…….....