UNLP

Facultad de Informática

Taller de Lecto Comprensión y Traducción de Inglés

AÑO 2022

CLASES TEÓRICO/PRÁCTICAS

ADJ. ANALIA NAPOLITANO
ADJ. SANDRA PERALTA

COMISIONES DE TRABAJOS PRACTICOS

Ayudantes Diplomados:

Gabriela FRAGA

Florencia REGUERAL

María Paula GAVAGNIN

REGLAMENTO DE CURSADA – MODALIDAD PRESENCIAL

- La asignatura tiene carácter de LIBRE. No obstante, la cátedra ofrece la posibilidad de cursarla como alumno regular y aprobarla por promocion -sin examen final (cumplidos los requisitos correspondientes).

- La asignatura consta de clases teórico-prácticas dictadas en dos horarios a elección de los alumnos, de tres horas reloj, a cargo de las Prof. Adjuntas, y clases prácticas organizadas en tres comisiones, de tres horas, a cargo de las ayudantes de la cátedra. Al momento de la inscripcion los alumnos deberán elegir UNA comisión de Trabajos Practicos y UNA de las clases teórico prácticas.

1. ***Examen final libre:*** se podrá rendir todos los meses. Los exámenes finales se tomarán los días martes a las 16.00 hs en aula a confirmar. Tanto las fechas de las mesas como las de inscripciones coincidirán con las del resto de las asignaturas según Calendario Académico (CA).

Los alumnos que deseen rendir LIBRE, deberán inscribirse por SIU en la semana de inscripción correspondiente a cada llamado. Podrán visitar el curso **“Inglés: Exámenes y Consultas”** en el entorno IDEAS (UNLP) donde encontrarán toda la información necesaria sobre el examen, incluidas las clases de consulta previas a cada mesa de finales.

2. ***Cursada Regular con examen final:*** los alumnos que opten por esta modalidad deberán:

- **asistir al 80% de las clases prácticas** en la comisión elegida,
- **aprobar una de las dos Evaluaciones Diagnósticas** previas al parcial,
- aprobar el examen parcial con una calificación **mínima de 4 puntos.**
- **preparar y presentar un Trabajo Final grupal obligatorio, seguido de un coloquio oral individual**. EL resultado de este trabajo no se computará para la aprobación de la cursada regular.

Una vez obtenida la cursada regular, los alumnos rendirán un Examen Final Regular en la fecha de su elección**, inscribiéndose por el SIU en carácter de LIBRES por ser la única opción que acepta el sistema** (la cátedra guarda registros de la condición REGULAR de los alumnos para el día del examen final.)

3. ***Promoción sin examen final:*** los alumnos que opten por esta modalidad deberán:

- **asistir al 80% de las clases prácticas** en la comisión elegida**,**
- **asistir al 80% de las clases teórico prácticas** en el horario elegido al momento de la inscripción,
- **aprobar las dos Evaluaciones Diagnósticas** previas al parcial (que incluyen una instancia de recuperación),
- aprobar el examen parcial con una calificación **mínima de 6 puntos, sin hacer uso de instancias recupertorias** y

- **preparar y presentar un Trabajo Final grupal obligatorio, seguido de un coloquio oral individual**. EL resultado de este trabajo no se computará para la aprobación de la cursada por promoción.

NOTA: se accede a esta promoción aún rindiendo el recuperatorio de las evaluaciones diagnósticas.

Una vez aprobada la promoción, **los alumnos deberán inscribirse por SIU al examen final en carácter de LIBRES** para que la nota de promoción sea pasada al Acta de Finales y quede así asentada en los regitros correspondientes.
Aquellos alumnos que obtengan **una nota menor a 6 en el parcial**, pasarán automáticamente al régimen de cursada regular y deberán rendir un Examen Final Regular en la fecha de su elección.

- Habrá **2 recuperatorios** para el examen parcial en fechas que se informarán al inicio de la cursada.

- Las docentes coordinarán con los alumnos (en forma personal o por medio de la cartelera física o la virtual o el entorno de IDEAS) **las muestras de todas las evaluaciones** que se tomen durante la cursada.

- Los alumnos que se presenten a rendir examen final libre o regular deberán hacerlo con libreta de estudiante o DNI con foto, diccionarios bilingües y/o glosarios impresos. **No se permitirá** el uso de dispositivos electrónicos ni computadoras.

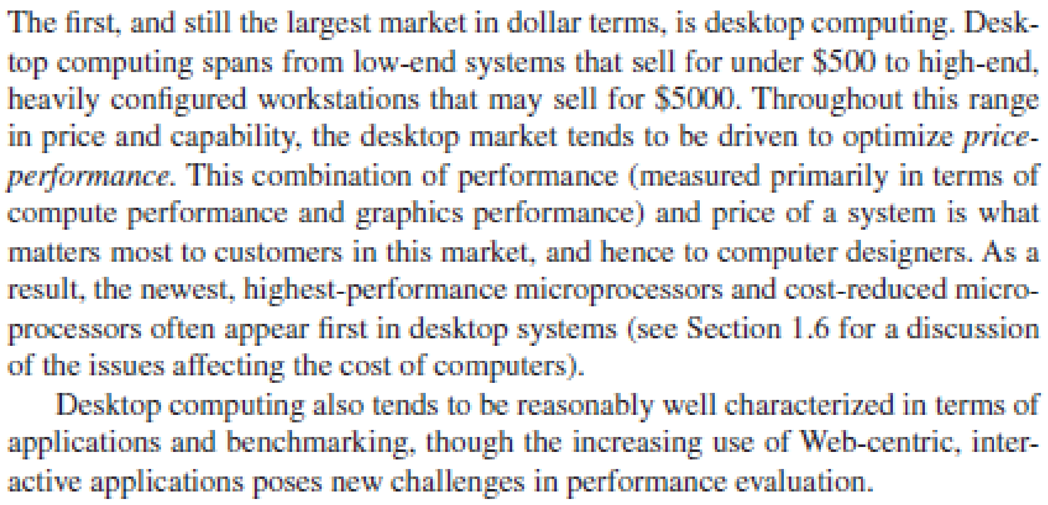
¿De qué tratan estos textos?

- **Lea estos textos en silencio y reflexione con un compañero sobre lo siguiente:**

    1. ¿Qué tienen en común?
    2. ¿En qué se diferencian?
    3. ¿Cuál es más fácil o difícil de entender? ¿Por qué?
    4. ¿Con qué disciplina o campo del saber se relacionan?

A. The principles of social psychology, including the ABCs—affect, behavior, and cognition—apply to the study of stereotyping, prejudice, and discrimination, and social psychologists have expended substantial research efforts studying these concepts (Figure 12.1). The cognitive component in our perceptions of group members is the stereotype—the positive or negative beliefs that we hold about the characteristics of social groups. We may decide that “Italians are romantic,” that “old people are boring,” or that “college professors are nerds.” And we may use those beliefs to guide our actions toward people from those groups. In addition to our stereotypes, we may also develop prejudice—an unjustifiable negative attitude toward an outgroup or toward the members of that outgroup. Prejudice can take the form of disliking, anger, fear, disgust, discomfort, and even hatred—the kind of affective states that can lead to behavior such as the gay bashing you just read about. Our stereotypes and our prejudices are problematic because they may create discrimination—unjustified negative behaviors toward members of outgroups based on their group membership.

B.



The first, and still the largest market in dollar terms, is desktop computing. Desktop computing spans from low-end systems that sell for under $500 to high-end, heavily configured workstations that may sell for $5000. Throughout this range in price and capability, the desktop market tends to be driven to optimize *price-performance*. This combination of performance (measured primarily in terms of compute performance and graphics performance) and price of a system is what matters most to customers in this market, and hence to computer designers. As a result, the newest, highest-performance microprocessors and cost-reduced microprocessors often appear first in desktop systems (see Section 1.6 for a discussion of the issues affecting the cost of computers).

Desktop computing also tends to be reasonably well characterized in terms of applications and benchmarking, though the increasing use of Web-centric, interactive applications poses new challenges in performance evaluation.

C. Bahia Grass (*Paspalumnotatum*) , which is native to Brazil (and named after the eastern state of Bahia), Argentina, Uruguay and Paraguay, has been introduced into warm areas of the US for use both in lawns and in pastures, but sometimes becomes a roadside weed. It is a perennial species that spreads both by rhizomes and seeds, and finds its greatest use in Florida and coastal areas of nearby states. Bahia Grass is useful because while it prefers moist or even wet soils, it can tolerate drought well, and it tends to grow quickly enough to crowd out weeds. It is a “warm season” grass that grows to be some 12 to 20 inches tall and the flowering stalk usually forms a distinct “Y”. Allergenically, it demonstrates moderate cross-reactivity with Johnson Grass but relatively little with most other grasses, and is considered to be one of the three main “Southern Grasses”.

TEXTO 1

What are the five elements of a computer system?

The five elements of a computer system are datapath, control, memory, input and output. All these elements work together to allow the computer to function properly.

The input and the output elements of a computer are the elements with which end users interact. These elements include input devices such as keyboards, mice and external drives. The output elements include devices such as monitors and printers.

The remaining three elements of a computer system function to process data. The datapath works with the data that comes through the processor. The memory component stores the data and instructions for programs that are in use on the computer. The control element directs the operation of the datapath and memory.

What are the external parts of a computer?

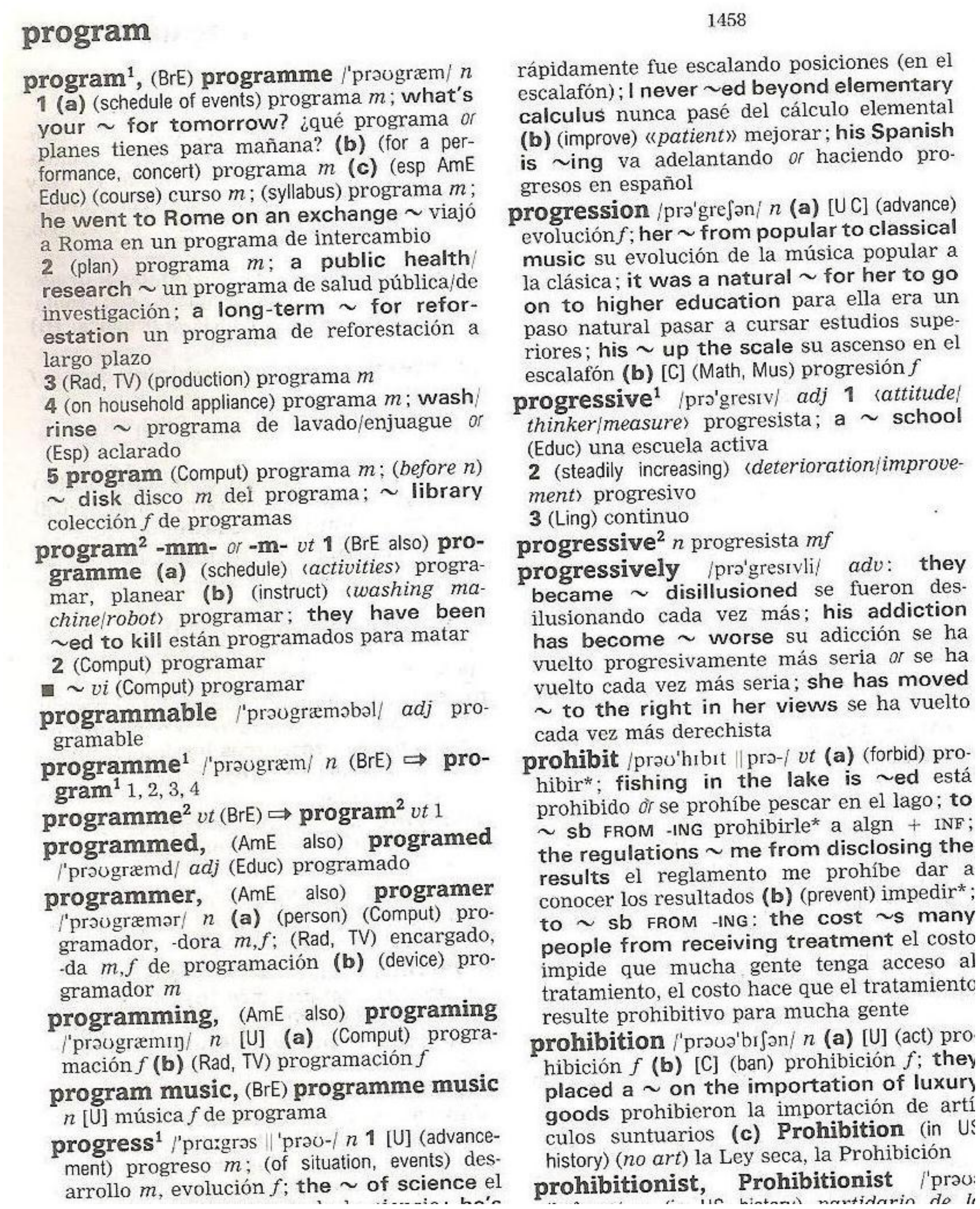
External computer parts are those that connect to the case, often to provide ways to input or output data. Most computers use a keyboard and mouse as external input devices and a monitor as an output device. Speakers, printers, modems, network routers and external storage devices are other external devices.

External computer components connect to the motherboard through a series of ports on the computer case. Many components have their own dedicated port connections, such as those to connect audio equipment or the computer monitor. Others may share a single type of connector, such as the USB, that connects everything from keyboards and mice to game pads and external hard drives.

(from https://www.reference.com)

Búsqueda en el diccionario

- **Observemos qué información nos ofrece el diccionario:**



program

1458

program^1, (BrE) **programme** /'prəʊgræm/ *n* **1 (a)** (schedule of events) programa *m*; **what's your ~ for tomorrow?** ¿qué programa *or* planes tienes para mañana? **(b)** (for a performance, concert) programa *m* **(c)** (esp AmE Educ) (course) curso *m*; (syllabus) programa *m*; **he went to Rome on an exchange ~** viajó a Roma en un programa de intercambio
2 (plan) programa *m*; **a public health/research ~** un programa de salud pública/de investigación; **a long-term ~ for reforestation** un programa de reforestación a largo plazo
3 (Rad, TV) (production) programa *m*
4 (on household appliance) programa *m*; **wash/rinse ~** programa de lavado/enjuague *or* (Esp) aclarado
5 program (Comput) programa *m*; (*before n*) **~ disk** disco *m* del programa; **~ library** colección *f* de programas

program^2 -mm- *or* **-m-** *vt* **1** (BrE also) **programme (a)** (schedule) ‹*activities*› programar, planear **(b)** (instruct) ‹*washing machine/robot*› programar; **they have been ~ed to kill** están programados para matar
2 (Comput) programar
■ ~ *vi* (Comput) programar

programmable /'prəʊgræməbəl/ *adj* programable

programme^1 /'prəʊgræm/ *n* (BrE) ⇨ **program^1** 1, 2, 3, 4

programme^2 *vt* (BrE) ⇨ **program^2** *vt* 1

programmed, (AmE also) **programed** /'prəʊgræmd/ *adj* (Educ) programado

programmer, (AmE also) **programer** /'prəʊgræmər/ *n* **(a)** (person) (Comput) programador, -dora *m,f*; (Rad, TV) encargado, -da *m,f* de programación **(b)** (device) programador *m*

programming, (AmE also) **programing** /'prəʊgræmɪŋ/ *n* [U] **(a)** (Comput) programación *f* **(b)** (Rad, TV) programación *f*

program music, (BrE) **programme music** *n* [U] música *f* de programa

progress^1 /'prɑːgrəs || 'prəʊ-/ *n* **1** [U] (advancement) progreso *m*; (of situation, events) desarrollo *m*, evolución *f*; **the ~ of science** el

rápidamente fue escalando posiciones (en el escalafón); **I never ~ed beyond elementary calculus** nunca pasé del cálculo elemental **(b)** (improve) «*patient*» mejorar; **his Spanish is ~ing** va adelantando *or* haciendo progresos en español

progression /prə'greʃən/ *n* **(a)** [U C] (advance) evolución *f*; **her ~ from popular to classical music** su evolución de la música popular a la clásica; **it was a natural ~ for her to go on to higher education** para ella era un paso natural pasar a cursar estudios superiores; **his ~ up the scale** su ascenso en el escalafón **(b)** [C] (Math, Mus) progresión *f*

progressive^1 /prə'gresɪv/ *adj* **1** ‹*attitude/thinker/measure*› progresista; **a ~ school** (Educ) una escuela activa
2 (steadily increasing) ‹*deterioration/improvement*› progresivo
3 (Ling) continuo

progressive^2 *n* progresista *mf*

progressively /prə'gresɪvli/ *adv*: **they became ~ disillusioned** se fueron desilusionando cada vez más; **his addiction has become ~ worse** su adicción se ha vuelto progresivamente más seria *or* se ha vuelto cada vez más seria; **she has moved ~ to the right in her views** se ha vuelto cada vez más derechista

prohibit /prəʊ'hɪbɪt || prə-/ *vt* **(a)** (forbid) prohibir*; **fishing in the lake is ~ed** está prohibido *or* se prohíbe pescar en el lago; **to ~ sb FROM -ING** prohibirle* a algn + INF; **the regulations ~ me from disclosing the results** el reglamento me prohíbe dar a conocer los resultados **(b)** (prevent) impedir*; **to ~ sb FROM -ING: the cost ~s many people from receiving treatment** el costo impide que mucha gente tenga acceso al tratamiento, el costo hace que el tratamiento resulte prohibitivo para mucha gente

prohibition /'prəʊə'bɪʃən/ *n* **(a)** [U] (act) prohibición *f* **(b)** [C] (ban) prohibición *f*; **they placed a ~ on the importation of luxury goods** prohibieron la importación de artículos suntuarios **(c) Prohibition** (in US history) (*no art*) la Ley seca, la Prohibición

prohibitionist, Prohibitionist /'prəʊ

(From the Oxford Spanish Dictionary)

TEXTO 2

techopedia™

Central Processing Unit (CPU)

Definition - What does Central Processing Unit (CPU) mean?

The central processing unit (CPU) is the unit which performs most of the processing inside a computer. **This term** is also known as a central processor, microprocessor or chip. To control instructions and data flow to and from other parts of the computer, the CPU relies heavily on a chip set, which is a group of microchips located on the motherboard.

The CPU has two typical components:

- Control Unit: extracts instructions from memory and decodes and executes **them.**
- Arithmetic Logic Unit (ALU): handles arithmetic and logical operations.

To function properly, the CPU relies on the system clock, memory, secondary storage, and data and address buses.

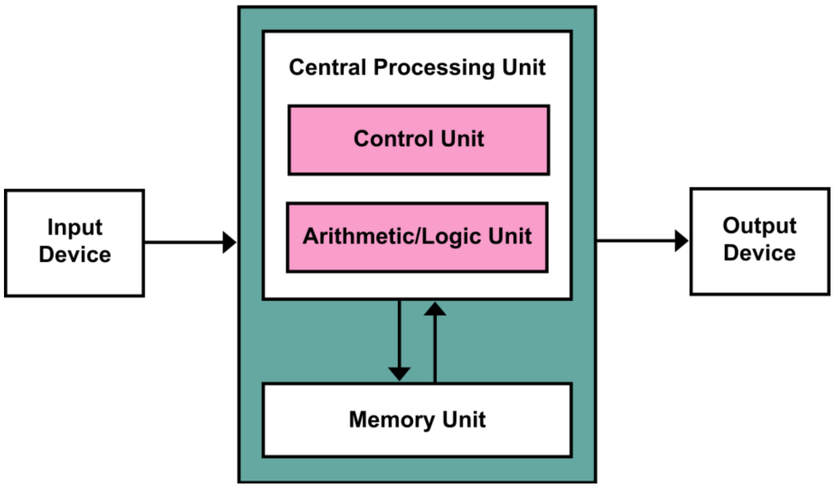
Techopedia explains Central Processing Unit (CPU)

The CPU is the heart and brain of a computer. **It** receives data input, executes instructions, and processes information. **It** communicates with Input/Output (I/O) devices, which send and receive data to and from the CPU. Additionally, the CPU has an internal bus for communication with the internal cache memory, called the backside bus. The main bus for data transfer to and from the CPU, memory, chipset, and AGP socket is called the front side bus.

CPU contains internal memory units, which are called registers. **These registers** contain data, instructions, counters, and addresses used in the ALU information processing.

Some computers have dual or multiple processors. **These** consist of two or more separate physical CPUs located side-by-side on the same board or on separate boards. Each CPU has an independent interface, separate cache, and individual paths to the system front-side bus. Multiple processors are ideal for intensive parallel tasks requiring multitasking.

(from https://www.techopedia.com)



TEXTO 3

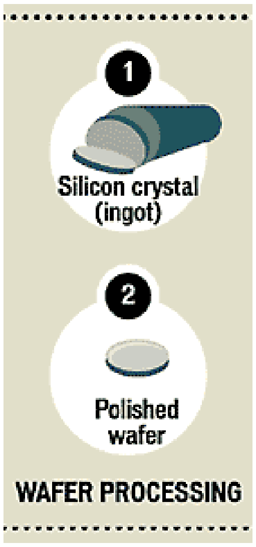
Making Microchips

By Gary Anthes

The simple element silicon in sand is the starting point for making integrated circuits. Silicon is a natural semiconductor. Under some conditions, it conducts electricity; under others, it acts as an insulator. That is why we use the term “semiconductor.” This on/off capability is what underlies the transistor switching action that forms the ones and zeros of digital logic.

Several types of microchips are made today. Microprocessors are logic chips that perform the computations inside most commercial computers. Memory chips store information. Digital signal processors convert between analog and digital signals. Application-specific integrated circuits are special-purpose chips used in things such as cars and appliances.

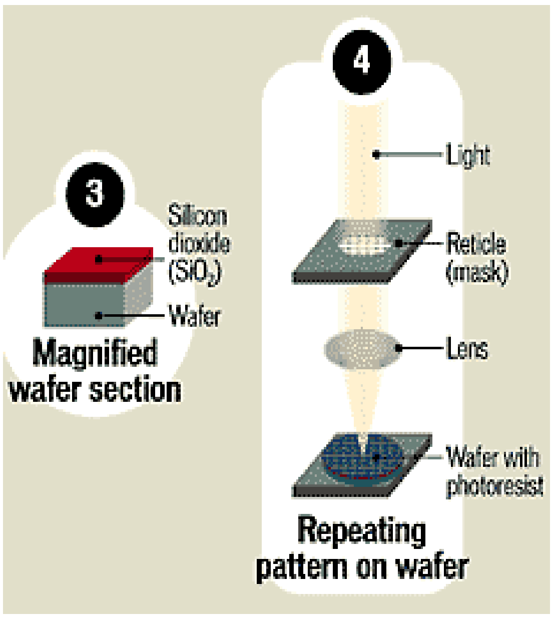
The Process



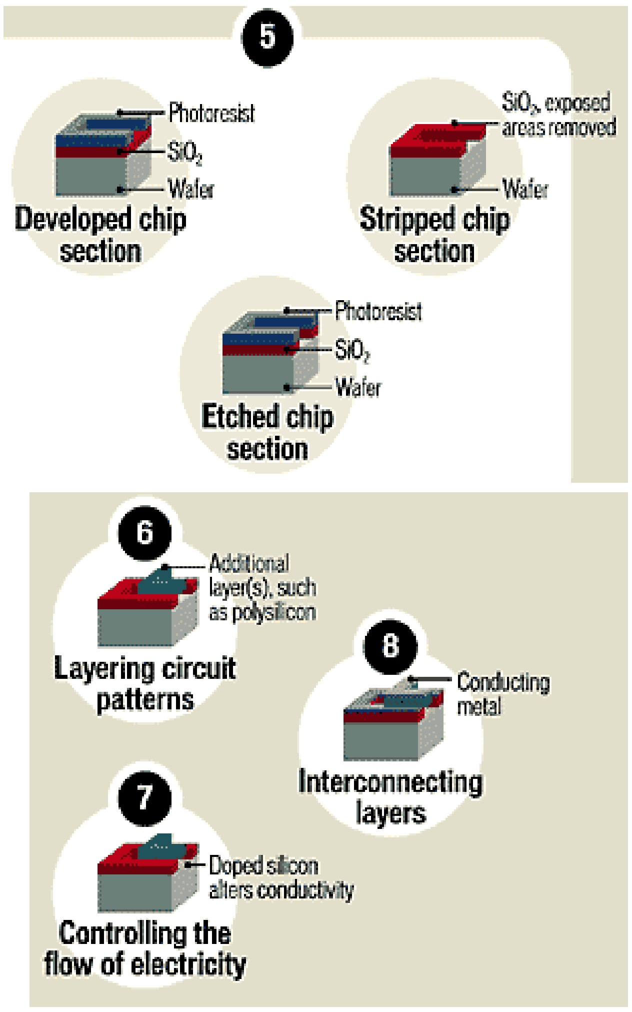
Chips are made in multibillion-dollar fabrication plants called fabs. Fabs melt and refine sand to produce 99.9999% pure single-crystal silicon ingots. The ingot is then carefully sawed into thin wafers the diameter of the ingot, most commonly 200 mm (8-inches) or 300 mm (12-inches) across.

The wafers are cleaned and polished, and each one is used to build multiple chips. These and subsequent steps are done in a "clean room" environment, where extensive precautions are taken to prevent contamination by dust and other foreign substances.

A nonconducting layer of silicon dioxide is deposited on the surface of the silicon wafer, and that layer is covered with a photosensitive chemical called a photoresist.



The photoresist is exposed to ultraviolet light, through a patterned plate (a mask) to harden the areas exposed to the light. Unexposed areas are then etched away by hot gasses to reveal the silicon dioxide base below. The base and the silicon layer below are further etched to varying depths.


5
Photoresist
SiO_2
Wafer
Developed chip section
SiO_2, exposed areas removed
Wafer
Stripped chip section
Photoresist
SiO_2
Wafer
Etched chip section
6
Additional layer(s), such as polysilicon
Layering circuit patterns
8
Conducting metal
Interconnecting layers
7
Doped silicon alters conductivity
Controlling the flow of electricity

The photoresist hardened by this process of photolithography is then stripped away, leaving a 3-D landscape on the chip that replicates the circuit design embodied in the mask. The electrical conductivity of certain parts of the chip can also be altered by doping (adding impurities) them with chemicals under heat and pressure. Photolithography using different masks, followed by more etching and doping, can be repeated hundreds of times for the same chip, producing a more complex integrated circuit at each step.

Each chip on the wafer is tested for correct performance and then separated from other chips on the wafer by a saw. Good chips are placed into the supporting packages that allow them to be plugged into circuit boards, and bad chips are marked and discarded.

THE FINAL PRODUCT
9
Wafer's chips are tested and sorted

<u>Glosario:</u>
Wafer: oblea (de silicio)
Photoresist. capa fotoresistente
Etch away: realizar o someter a ataque químico o fresado químico.

TEXTO 4

3.2 Varieties of Arrays

In some languages, the size of an array must be established once and for all at program design time and cannot change during execution. Such arrays are called static arrays. A chunk of memory big enough to hold all the values in the array is allocated when the array is created, and thereafter elements are accessed using the fixed base location of the array. Static arrays are the fundamental array type in most older procedural languages, such as Fortran, Basic, and C, and in many newer object-oriented languages as well, such as Java.

Some languages provide arrays whose sizes are established at run-time and can change during execution. These dynamic arrays have an initial size used as the basis for allocating a segment of memory for element storage. Thereafter, the array may shrink or grow. If the array shrinks during execution, then only an initial portion of allocated memory is used. But if the array grows beyond the space allocated for it, a more complex reallocation procedure, must occur as follows:

1. A new segment of memory large enough to store the elements of the expanded array is allocated.
2. All elements of the original (unexpanded) array are copied into the new memory segment.
3. The memory used initially to store array values is freed and the newly allocated memory is associated with the array variable or reference.

This reallocation procedure is computationally expensive, so systems are usually designed to minimize its frequency of use. For example, when an array expands beyond its memory allocation, its memory allocation might be doubled even if space for only a single additional element is needed. The hope is that providing a lot of extra space will avoid many expensive reallocation procedures if the array expands over time.

Dynamic arrays are convenient for programmers because they can never be too small –whenever more space is needed in a dynamic array, it can simply be expanded. One drawback of dynamic arrays is that implementing language support for them is more work for the compiler or interpreter writer. A potentially more serious drawback is that the expansion procedure is expensive, so there are circumstances when using a dynamic array can be dangerous. For example, if an application must respond in real time to events in its environment, and a dynamic array must be expanded when the application is in the midst of a response, then the response may be delayed too long, causing problems.

(From: *Concise notes on data structure and algorithms*)

TEXTO 5

2.3 Input/Output

Input arrives at the computer at unpredictable intervals. The system must be able to detect its arrival and respond to it.

2.3.1 Interrupts

Interrupts are hardware triggered signals which cause the CPU to stop what it is doing and jump to a special subroutine. Interrupts normally arrive from hardware devices, such as when the user presses a key on the keyboard. They can also be generated in software by errors like *division by zero* or *illegal memory address*.

When the CPU receives an interrupt, it saves the contents of its registers on the hardware stack and jumps to a special routine which will determine the cause of the interrupt and respond to it appropriately. Interrupts occur at different levels. Low level interrupts can be interrupted by high level interrupts. Interrupt handling routines have to work quickly, or the computer will be drowned in the business of servicing interrupts. For certain critical operations, low level interrupts can be ignored by setting a *mask*.

2.3.2 Buffers

The CPU and the devices attached to it do not work at the same speed. *Buffers* are therefore needed to store incoming or outgoing information temporarily, while it is waiting to be picked up by the other party. A buffer is simply an area of memory which works as a waiting area. It is a *first-in first-out* (FIFO) data structure or *queue*.

2.3.3 Synchronous and asynchronous I/O

To start an I/O operation, the CPU writes appropriate values into the registers of the device controller. The device controller acts on the values it finds in its registers. For example, if the operation is to read from a disk, the device controller fetches data from the disk and places it in its local buffer. It then signals the CPU by generating an interrupt.

While the CPU is waiting for the I/O to complete it may do one of two things. It can do nothing or *idle* until the device returns with the data (synchronous I/O), or it can continue doing something else until the completion interrupt arrives (asynchronous I/O). The second of these possibilities is clearly much more efficient.

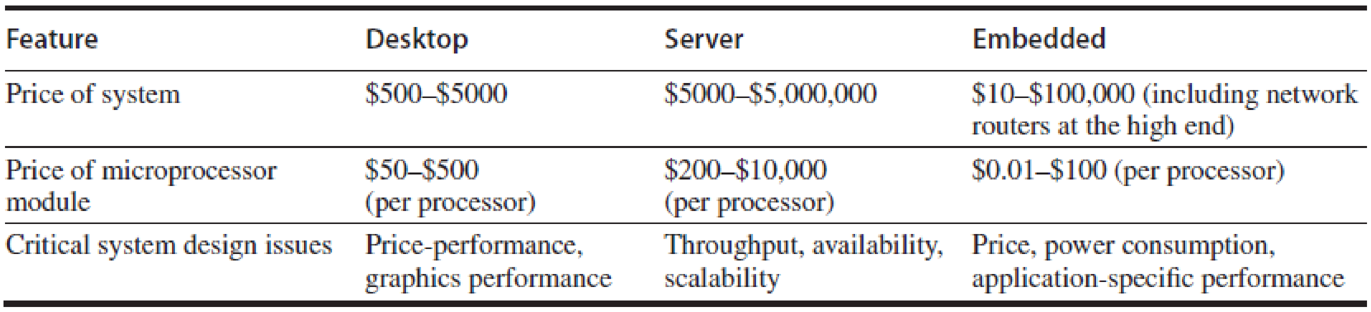
(**From** *A short introduction to operating systems* by Mark Burgess. October 3, 2001)

TEXTO 6

Classes of Computers

In the 1960s, the dominant form of computing *was* on large mainframes –computers costing millions of dollars and stored in computer rooms with multiple operators overseeing their support. Typical applications *included* business data processing and large-scale scientific computing. The 1970s *saw* the birth of the minicomputer, a smaller-sized computer initially focused on applications in scientific laboratories, but rapidly branching out with the popularity of time-sharing –multiple users sharing a computer interactively through independent terminals. **That decade** also *saw* the emergence of supercomputers, which *were* high-performance computers for scientific computing. Although few in number, **they** *were* important historically because **they** *pioneered* innovations that later *trickled down to* less expensive computer classes. The 1980s *saw* the rise of the desktop computer based on microprocessors, in the form of both personal computers and workstations. The individual owned desktop computer *replaced* time-sharing and *led* to the rise of servers –computers that provided larger-scale services such as reliable, long-term file storage and access, larger memory, and more computing power. The 1990s *saw* the emergence of the Internet and the World Wide Web, the first successful handheld computing devices (personal digital assistants or PDAs), and the emergence of high-performance digital consumer electronics, from video games to set-top boxes. The extraordinary popularity of cell phones <u>has been</u> obvious since 2000, with rapid improvements in functions and sales that far exceed **those** of the PC. **These more recent applications** use embedded computers, where computers are lodged in other devices and **their** presence is not immediately obvious.

These changes have set the stage for a dramatic change in how we view computing, computing applications, and the computer markets in this new century. Not since the creation of the personal computer more than 20 years ago have **we** seen such dramatic changes in the way computers appear and in how **they** are used. These changes in computer use have led to three different computing markets, each characterized by different applications, requirements, and computing technologies. Figure 1.2 summarized these mainstream classes of computing environments and **their** important characteristics.



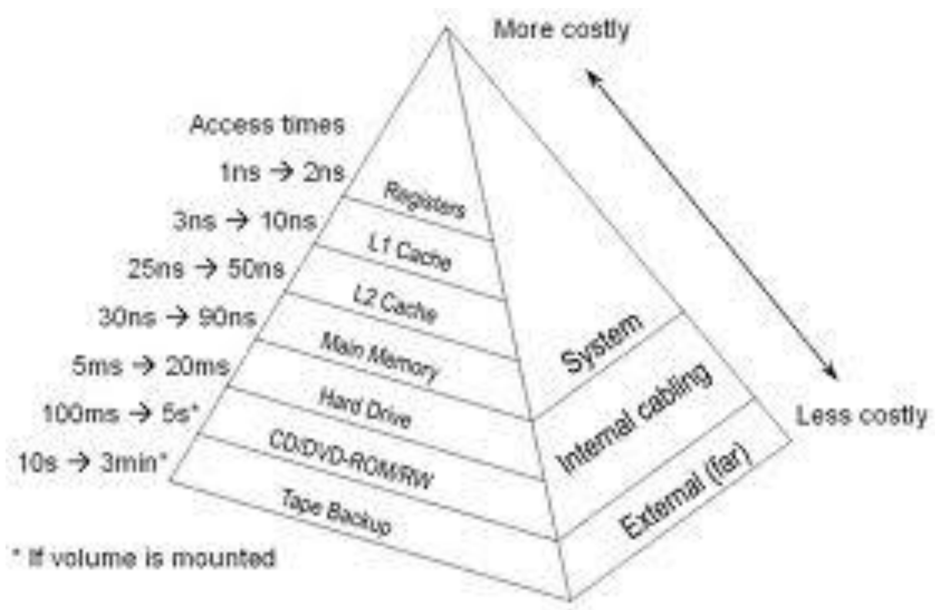
Feature	Desktop	Server	Embedded
Price of system	\$500–\$5000	\$5000–\$5,000,000	\$10–\$100,000 (including network routers at the high end)
Price of microprocessor module	\$50–\$500 (per processor)	\$200–\$10,000 (per processor)	\$0.01–\$100 (per processor)
Critical system design issues	Price-performance, graphics performance	Throughput, availability, scalability	Price, power consumption, application-specific performance

Figure 1.2 A summary of the three mainstream computing classes and their system characteristics. Note the wide range in system price for servers and embedded systems. For servers, this range arises from the need for very large-scale multiprocessor systems for high-end transaction processing and Web server applications. The total number of embedded processors sold in 2005 is estimated to exceed 3 billion if you include 8-bit and 16-bit microprocessors. Perhaps 200 million desktop computers and 10 million servers were sold in 2005.

TEXTO 7

Memory Hierarchy

Memory Hierarchy with its accessing speed



At the top level of the memory hierarchy are the CPU's general purpose registers. The registers provide the fastest access to data possible on the 80x86 CPU. The register file is also the smallest memory object in the memory hierarchy (with just eight general purpose registers available). By virtue of the fact that it is virtually impossible to add more registers to the 80x86, registers are also the most expensive memory locations.

Working our way down, the Level One Cache system is the next highest performance subsystem in the memory hierarchy. On the 80x86 CPUs, the Level One Cache is provided on-chip by Intel and cannot be expanded. The size is usually quite small (typically between 4Kbytes and 32Kbytes), though much larger than the registers available on the CPU chip. Although the Level One Cache size is fixed on the CPU and you cannot expand it, the cost per byte of cache memory is much lower than that of the registers because the cache contains far more storage than is available in all the combined registers.

The Level Two Cache is present on some CPUs, on other CPUs it is the system designer's task to incorporate this cache (if it is present at all). For example, most Pentium II, III, and IV CPUs have a level two cache as part of the CPU package, but many of Intel's Celeron chips do not. The Level Two Cache is generally much larger than the level one cache (e.g., 256 or 512KBytes versus 16 Kilobytes). On CPUs where Intel includes the Level Two Cache as part of the CPU package, the cache is not expandable. On systems where the Level Two Cache is external, many system designers let the end user select the cache size and upgrade the size. For economic reasons, external caches are actually more expensive than caches that are part of the CPU package, but the cost per bit at the transistor level is still equivalent to the in-package caches.

Below the Level Two Cache system in the memory hierarchy falls the main memory subsystem. This is the general-purpose, relatively low-cost memory found in most computer systems. Typically, this is DRAM or some similar inexpensive memory technology.

<u>TEXTO 8</u>

1. Introduction to database development

Databases are at the centre of most information systems in everyday use. Therefore, it is important that they are designed and built using appropriate methods to ensure that they meet users' requirements whilst being robust and maintainable. A database system is usually regarded as the database which contains related tables of data maintained by a database management system (DBMS), along with applications that provide controlled access to the database.
In order to build an effective database system, it is important to understand and apply the database development lifecycle, which includes the following phases:

1. Strategy and planning –typically the cycle starts with the strategy and planning phase to identify the need and scope of a new system.
2. Requirement analysis phase –a more detailed requirement analysis will be carried out which will include identifying what the users require of the system; this will involve conceptual analysis.
3. Design phase –this will involve producing a conceptual, logical and physical design. To undertake these processes it is important to be able to understand and apply the data modeling techniques which are covered in this book. When a suitable logical design has been obtained, the development phase can begin.
4. Development phase –this involves creating the database structure using an appropriate Database Management System (DBMS) and usually includes the development of applications that provide a user interface consisting of forms and reports which will allow controlled access to the data held in the database.
5. Deployment/implementation –when the system has been developed, it will be tested, it will then be deployed ready for use.
6. Operations and maintenance –following the system release for use it will be maintained until it reaches the end of its useful life, at this stage the development lifecycle may restart.

1.1 Conceptual data modeling

Why do you need to model?
In many environments modeling is used to ensure that a product will satisfy the user's requirements before it is produced. For example, an architect may use a scale model of a building so the client can see what it will look like before it is built. This allows for any changes to be made to the design following feedback and before any expensive building work takes place. Similarly, a modeling approach is needed when designing a database system so that interested parties can check that the design will satisfy the requirements.
(from *Database design and implementation: A practical introduction using Oracle SQL*)

TEXTO 9

How to set up dual monitors in Windows 10

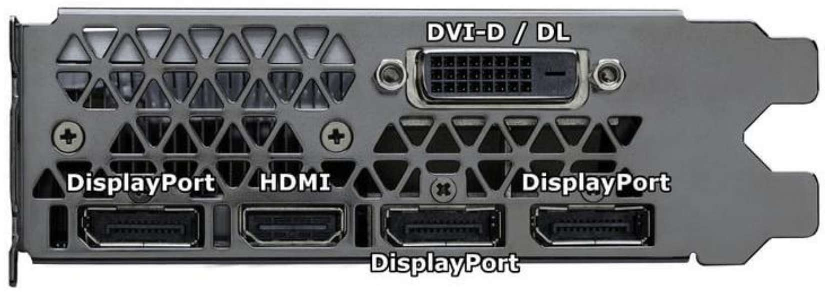
There are plenty of reasons to set up a second monitor for your Windows computer: ergonomics, easier scanning of large work spaces, and sharing presentations on a larger screen, to name a few.

Setting up dual monitors helps your productivity immensely. From having multiple browser windows open to using complex sets of editing tools for photos or video or having guides open on a second screen for research or gaming — it's always an excellent way to maximize your productivity. Here you will see how to set up your second monitor quickly and simply.

Step 1: Check your I/O panel and GPU for connections

Your PC has an area for all important cable connections, typically called the I/O panel. If it's been a while since you've peeked back there, take a look *before* you buy a secondary monitor. If you have a discrete (non-integrated) GPU, then there may also be a GPU section with ports of its own to take stock of. Snap a quick photo of this whole section for quick reference if necessary.

Now check to see what kind of display connections you have to work with. For modern monitors and PCs, the two common options are HDMI and DisplayPort, with even newer models also offering USB-C and Thunderbolt 3 for A/V data. You may also have a DVI-I port for managing older digital/analog connections, and some PCs might still have a VGA port (although we don't advise using this for a second monitor).



Make sure you have at least two of these display ports for two monitors. Note which spare port you will be attaching the new monitor to and what connection type you'll need.

You can also use a monitor as a second screen with a laptop, as long as that laptop has compatible display ports of its own. The laptop screen itself can also be used as a second monitor with the right setup, although that isn't as common.

Step 2: Make sure your monitors are compatible and connect them

With port information in hand, you're ready to pick out the best new monitor for your dual-display setup. Double-check to see that the monitor includes the right type of port for your open PC connection, and buy any necessary cables as well. For a smooth whole-screen experience, it's a good idea to pick a monitor with a "bezel-less" or thin-bezel screen.

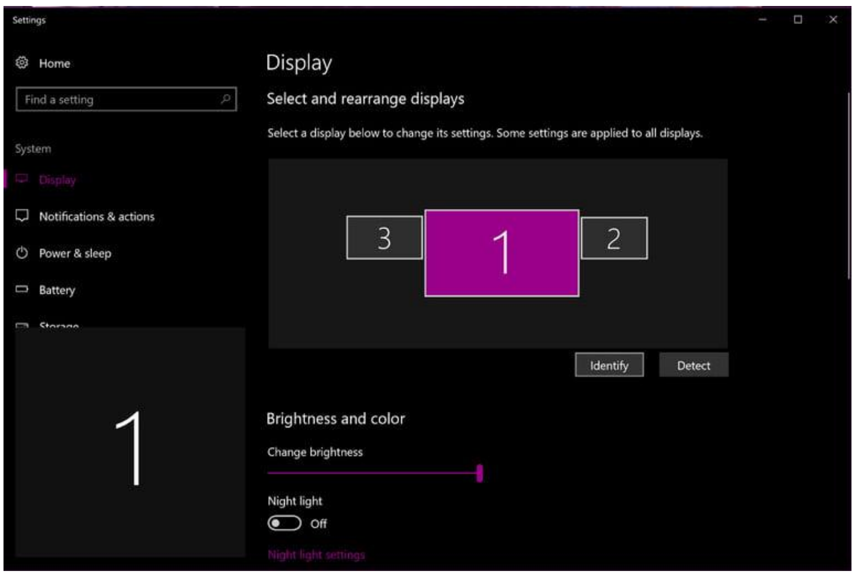
Now you're ready to position both monitors on your desk and connect them to your PC.

Step 3: Go to Display settings on Windows 10

Fire up your PC. When you reach the home screen, right-click on an empty part of the screen, and choose *Display Settings*.

The window will show your connected displays, which display is your primary display, and on which side the secondary is positioned (you can choose a different side by dragging the screen numbers). If you do not see your second monitor show up, try unplugging the cables and plugging them back in again. There is also a wireless option if you scroll down and choose *Connect to a Wireless Display* — less common but potentially useful. If it's still not working, check for Windows 10 updates, reboot, and try again.

Step 4: Choose your display option



Select a display below to change its settings. Some settings are applied to all displays.

Night light: Off

If both of your monitors are showing up, then you need to choose how they will work. Select your secondary monitor at the top, then scroll down in the *Display Settings* window, and select the *Multiple Displays* list to see your options.

You have two primary choices. The first is to *Extend Desktop to This Display*. This will make your two monitors function as a single whole monitor with a stretched-out desktop that you can freely move things between — the ideal choice for most dual-monitor users. The other option is to *Duplicate Desktop on 1 and 2*. This option is usually reserved for teaching and training setups where one screen will be facing the learners.

TEXTO 10

Trends in Technology

If an instruction set architecture is to be successful, it must be designed to survive rapid changes in computer technology. After all, a successful new instruction set architecture may last decades—for example, the core of the IBM mainframe has been in use for more than 40 years. An architect must plan for technology changes that can increase the lifetime of a successful computer. To plan for the evolution of a computer, the designer must be aware of rapid changes in implementation technology. Four implementation technologies, which change at a dramatic pace, are critical to modern implementations:

■ Integrated circuit logic technology —Transistor density increases by about 35% per year, quadrupling in somewhat over four years. Increases in die size are less predictable and slower, ranging from 10% to 20% per year. The combined effect is a growth rate in transistor count on a chip of about 40% to 55% per year. Device speed scales more slowly, as we discuss below.

■ Semiconductor DRAM (dynamic random-access memory) —Capacity increases by about 40% per year, doubling roughly every two years.

■ Magnetic disk technology —Prior to 1990, density increased by about 30% per year, doubling in three years. It rose to 60% per year thereafter, and increased to 100% per year in 1996. Since 2004, it has dropped back to 30% per year. Despite this roller coaster of rates of improvement, disks are still 50–100 times cheaper per bit than DRAM. This technology is central to Chapter 6, and we discuss the trends in detail there.

■ Network technology —Network performance depends both on the performance of switches and on the performance of the transmission system. We discuss the trends in networking in Appendix E. These rapidly changing technologies shape the design of a computer that, with speed and technology enhancements, may have a lifetime of five or more years. Even within the span of a single product cycle for a computing system (two years of design and two to three years of production), key technologies such as DRAM change sufficiently that the designer must plan for these changes. Indeed, designers often design for the next technology, knowing that when a product begins shipping in volume that next technology may be the most costeffective or may have performance advantages. Traditionally, cost has decreased at about the rate at which density increases. Although technology improves continuously, the impact of these improvements can be in discrete leaps, as a threshold that allows a new capability is reached. For example, when MOS technology reached a point in the early 1980s where between 25,000 and 50,000 transistors could fit on a single chip, it became possible to build a single-chip, 32-bit microprocessor. By the late 1980s, first level caches could go on chip. By eliminating chip crossings within the processor and between the processor and the cache, a dramatic improvement in cost-performance and power-performance was possible.