Del mundo real a las soluciones con computadoras.

La Informática es la ciencia que estudia el análisis y la resolución de problemas mediante el uso de una computadora.
La palabra ciencia se relaciona con una metodología fundamentada y racional de analizar y estudiar problemas.
La resolución de un problema es el proceso completo que se extiende desde el momento en que se enuncia el problema hasta que se escribe el programa que lo resuelve.
La resolución de un problema es un **proceso creativo**, no existe un método universal que permite resolver cualquier problema. El conocimiento, la habilidad y la experiencia tienen un papel importante en la resolución. El proceder de manera sistemática, es decir siguiendo un conjunto ordenado de normas, puede ayudar en la resolución, porque se adquiere una metodología de resolución.

Pero…..¿qué es un problema?

- Un problema es un asunto o un conjunto de cuestiones que se plantean para ser resueltas y cuya solución no se conoce de antemano.

La naturaleza de los problemas varía con el ámbito o contexto donde están planteados. De esa forma podemos encontrar problemas de índole matemática, física, de simulación, de control, contables, estadísticos, etc.
En particular nos interesan los problemas cuya solución puede expresarse en términos de una computadora, es decir mediante un algoritmo.
Primero hay que saber qué problema hay que resolver. Si no se tiene una descripción simple y precisa de él, resulta muy complejo modelar, simular o programar su solución. En general, conviene expresar el problema mediante un modelo formal. Una vez modelado, puede buscarse una solución algorítmica: un conjunto finito y no ambiguo de acciones expresadas en cierto orden, entendibles por una computadora.

- Una computadora es una máquina digital y sincrónica, que tiene una cierta capacidad de cálculo matemático y lógico, controlada por un programa almacenado internamente, y que permite comunicarse con el mundo exterior.

Decimos que es una máquina digital porque toda la información que procesa es transformada a dígitos binarios (0 o 1)
Es sincrónica porque su funcionamiento está controlado por un reloj principal interno que envía señales a cada componente de la computadora.
Dispone de una unidad aritmético lógica (UAL) que le permite realizar operaciones básicas de suma, resta, multiplicación, etc., y de lógica proposicional (disyunciones, conjunciones, negaciones, etc.).
La frase controlada por un programa, significa que internamente se tienen órdenes almacenadas (que llamaremos instrucciones) que la computadora podrá leer, interpretar y ejecutar ordenadamente. En otras palabras, utiliza un programa llamado Sistema Operativo

(SO) , que controla el funcionamiento general de la misma: ejecuta los programas, controla los componentes, etc. .y se apoya en la Unidad de Control (UC).
Se comunica con el usuario mediante un conjunto de dispositivos periféricos que le permiten ingresar datos a la computadora y mostrar resultados.

Para comprender mejor el funcionamiento de una pc vamos a mencionar y describir cuáles son sus componentes fundamentales: **Hardware y Software**
Ninguno de estos componentes en forma aislada sirve: el hardware necesita de los programas que los controlen, y el software necesita del hardware para ejecutarse.

- El **software** comprende el conjunto de programas que controlan el manejo de los distintos componentes de una computadora.

- Un **programa** es un conjunto de órdenes escritas en un cierto orden lógico y en un lenguaje entendible por la computadora, que sirve para llevar a cabo una tarea específica.

Los programas se clasifican en:

- Sistemas operativos: aquellos que sirven para arrancar la computadora y controlar el funcionamiento global de la misma, ya que son los encargados de cargar, interpretar y ejecutar cada programa de la misma (MS-DOS, Windows, UNIX, Linux, Novell)

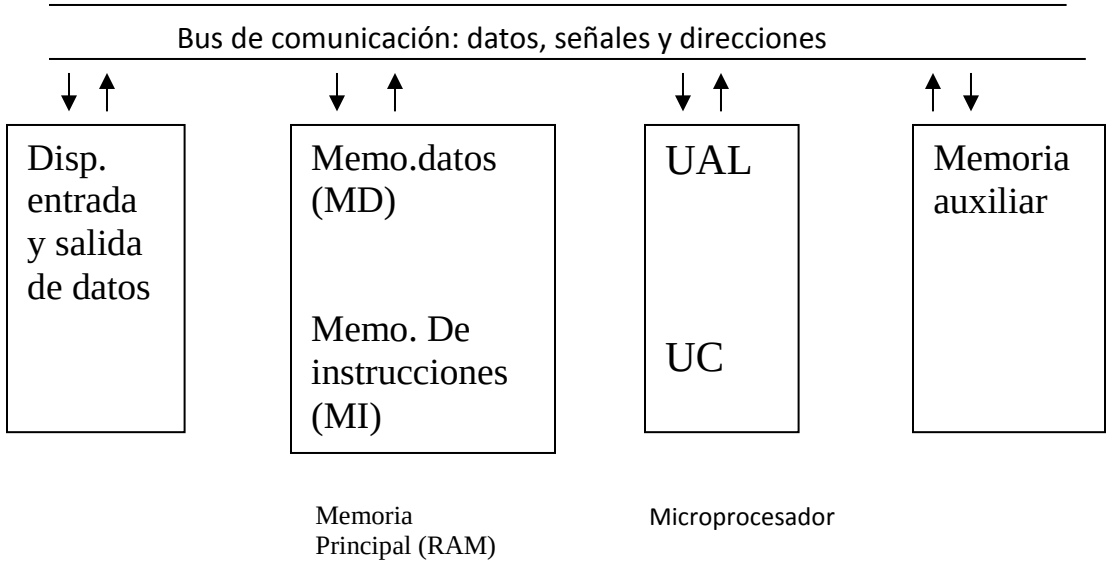
- Utilitarios: aquellos que nos suministran la posibilidad de obtener datos, procesarlos y generarlos (Word, Excel, Access, PowerPoint, PaintBrush)

- De aplicación: son los que desarrollamos para resolver un problema y ejecutar una tarea específica (facturación, altas clínica, biblioteca, etc.)

- Lenguajes de Programación: son los que nos permiten escribir programas de aplicación (Pascal, Basic, Visual Basic, C, Delphi, etc.)

- El **Hardware** comprende el conjunto de componentes físicos de la computadora

Una computadora está compuesta por una serie de elementos físicos, algunos de los cuales son imprescindibles y otros son accesorios. Todos ellos se encuentran conectados de alguna manera a un gabinete principal llamado CPU (unidad central de proceso).
Entre lo más comunes encontramos:



El Bus de comunicaciones es el encargado de transportar los datos, las señales de control y las direcciones entre los distintos componentes de la PC.

El microprocesador es un chip de alta velocidad, encargado de ejecutar las instrucciones de los programas. Para ello se basa en el uso de la unidad de control (UC) que es la encargada de tomar una a una las instrucciones de un programa desde la memoria de instrucciones (MI) , interpretarlas y ejecutarlas, guardando los resultados obtenidos en la memoria de datos (MD). Para llevar a cabo las operaciones matemáticas y lógicas coordina el funcionamiento de la UAL.
Los microprocesadores más modernos son: Intel Pentium G 4560- 4600 // Intel Core i3, i5, i7 // AMD Ryzen 7. etc..

La memoria principal es la que brinda almacenamiento temporal para programas y datos. Todas las computadoras vienen de fábrica con una memoria llamada ROM (memoria de solo lectura) la cual no se puede modificar ni perder, aunque se apague la computadora. Esta memoria es la que contiene una serie de instrucciones básicas para que se efectúe el arranque de la PC.
Pero la máquina necesita otro tipo de memoria, que nos permita trabajar con información temporal, donde se almacenen los datos y los programas durante el período de tiempo en el que la máquina está encendida. Esa memoria recibe el nombre de RAM (memoria de acceso rápido).

La memoria RAM está organizada como centenares o millares de unidades de almacenamiento individual o celdas. Cada celda de la memoria tiene asociada al menos 2 conceptos:

- su dirección o posición relativa, mediante la cual se puede acceder a la información en ella almacenada,
- que es lo que se llama, su contenido.

La mínima unidad de almacenamiento es un byte (octeto de bits), o sea una combinación de 8 dígitos binarios (0 y 1). Un byte permite almacenar un carácter. Generalmente el tamaño de los almacenamientos se mide en bytes y como múltiplos de 8. (8- 16 – 32- 64- 128- 256- …….)
1 byte = 8 bits
1024 bytes = 1 Kb 1024 Kb= 1 Mb 1024 Mb= 1Gb 1024 Gb= Tb

La característica más destacable de la memoria RAM es su rápido acceso, pero la desventaja más importante es que es una memoria volátil y de capacidad limitada.
Para almacenar datos en forma permanente y masiva se usa la memoria auxiliar o permanente: discos rígidos, cd, zip, Dvd, pendrive, usb, etc.. La información así almacenada se organiza en archivos que pueden ser transferidos desde y hacia la RAM fácilmente.
En contraposición a la memoria principal, la memoria auxiliar o secundaria es permanente y de gran capacidad.
1 pendrive 128 Gb / 256 Gb …
Disco rígido = 1 a 4 Terabyte

Los dispositivos de entrada y salida de datos reciben el nombre de periféricos, y son los que nos permiten comunicarnos con la computadora.
Entrada: lápiz óptico, teclado, mousse, scanner, joystick, webcam, micrófono, etc.
Salida: impresora, pantalla, parlantes, plotters, etc.

El Proceso de Resolución de problemas.

Lo primero que debemos hacer a la hora de resolver un problema, es elegir un “Paradigma de programación”.

A lo largo de la historia, el término «paradigma» fue objeto de muchas interpretaciones. En su origen griego, significaba «modelo», «ejemplo» o «patrón». Sobre este punto de partida, podemos hablar de un paradigma como un **conjunto de creencias, prácticas y conocimientos que guían el desarrollo de una disciplina durante un período de tiempo**. En diversas ramas de la ciencia, un conjunto de ideas en vigencia puede ser reemplazado drásticamente por otro que entre en conflicto con él y se demuestre más acertado.

El campo de la programación/informática tiene sus propios paradigmas, pero el término «paradigma de programación» **no necesariamente representa un modelo único que deba ser respetado hasta que aparezca otro mejor**. De hecho, actualmente muchos paradigmas coexisten en armonía.

Un paradigma de programación es un estilo de desarrollo de programas. Es decir, un modelo para resolver problemas computacionales.

Cada **paradigma de programación** es una colección de patrones conceptuales, que moldean la forma de razonar problemas, de formular algoritmos y de estructurar programas. Es decir, cada paradigma significa un modo diferente de analizar problemas y especificar programas.

Los lenguajes de programación, necesariamente, se encuadran en uno o varios paradigmas a la vez a partir del tipo de órdenes que permiten implementar, algo que tiene una relación directa con su sintaxis.

¿Cuáles son los principales paradigmas de programación?

- **Imperativo**. Los programas se componen de un conjunto de sentencias que cambian el estado de la memoria. Son secuencias de comandos que ordenan acciones a la computadora.

- **Declarativo**. Opuesto al imperativo. Los programas describen los resultados esperados sin listar explícitamente los pasos a llevar a cabo para alcanzarlos.

- **Lógico**. El problema se modela con enunciados de lógica de primer orden.

- **Funcional**. Los programas se componen de funciones matemáticas, es decir, implementaciones de comportamiento que reciben un conjunto de datos de entrada y devuelven un valor de salida.

- **Orientado a objetos.** El comportamiento del programa es llevado a cabo por objetos, entidades que representan elementos del problema a resolver y tienen atributos y comportamiento.

Otros paradigmas son de aparición relativamente reciente y no forman parte del grupo principal:

- **Dirigido por eventos**. El flujo del programa está determinado por sucesos externos (por ejemplo, una acción del usuario).

- **Orientado a aspectos**. Apunta a dividir el programa en módulos independientes, cada uno con un comportamiento bien definido.

Cada paradigma es ideal para la resolución de un conjunto de problemas particular, por lo que no puede decirse que uno sea necesariamente mejor que otro.

Dentro de la asignatura nosotros vamos a trabajar sobre el **Paradigma imperativo.**

La gran mayoría de los lenguajes desarrollados hasta hace pocos años eran orientados a la arquitectura impuesta por la máquina de Von Neuman: formada por un procesador, una memoria y elementos auxiliares y donde las instrucciones se ejecutan en forma cíclica y secuencialmente, siguiendo una línea de control única: se selecciona la instrucción siguiente, se interpreta y se ejecuta.
La memoria permite tener valores almacenados en celdas binarias consecutivas, tanto operaciones como datos, siendo la asignación la operación principal de almacenamiento de datos.
Para realizar algún cálculo se parte de ciertos datos almacenados y se realizan diversas operaciones, y, al final, el resultado es almacenado en alguna celda de memoria, denominada variable.
La forma básica de expresar un algoritmo en este paradigma es declarando las variables necesarias, y diseñando una secuencia de asignaciones que transforman los valores almacenados en tales variables y cuyos resultados también son almacenados en variables. Las instrucciones se ejecutan secuencialmente.

Los programas están orientados a sentencias u órdenes. Los efectos de éstas órdenes individuales se combinan en un programa para obtener los resultados deseados. Son lenguajes que se basan en el concepto de variable (como celda de memoria), asignación de valor (cada valor calculado debe ser almacenado), y en la repetición (un programa realiza una tarea ejecutando repetidamente una serie de pasos elementales).

¿Cómo se desarrollan programas de aplicación?

El proceso de resolución de un problema mediante el uso de una computadora, conduce a la escritura de un programa y a su ejecución en la misma.
Las 2 tareas más importantes que debe llevar a cabo un especialista informático al escribir un programa son:

- definir un conjunto de instrucciones o acciones cuya ejecución ordenada nos conduzca a la solución del problema planteado
- elegir la representación adecuada de los datos presentes en el problema

Siempre partimos de un problema del mundo real, el cual debemos modelizar y sintetizar para poder expresar la solución en términos de un programa que sea entendible por una computadora, manejando los datos del mismo mediante una representación válida.
El mundo real es naturalmente complejo y a veces los problemas a resolver son difíciles de sintetizar.

El proceso de análisis del mundo real, que nos permite interpretar aspectos esenciales del problema, se llama ABSTRACCION.
Cuando uno logra abstraer el problema del mundo real y simplificar su expresión para llevarlo a un programa entendible por una máquina, lo que se obtiene es un MODELO.

Etapas en la resolución de un problema.

1- Definición del ambiente. Etapa de abstracción
En esta etapa deben obtenerse los requerimientos del usuario, deben especificarse los datos de entrada (completarlos si fuese necesario), sacar ambigüedades y establecer en forma explícita las relaciones que puedan existir entre los datos de entrada y la información de salida. Se establece e**l QUÉ**. Resultado de esta etapa: modelo y estrategia

2- Diseño del algoritmo
En esta etapa se determina **CÓMO** se resuelve el problema, es decir se detallan las tareas que se llevan a cabo y nos conducen a la solución.
Existen diversas metodologías o **técnicas de diseño** para plantear la solución a un problema (ver apunte anexo Técnicas de diseño).

1- Diagramas de flujo (método gráfico)
2- Método de la fuerza bruta: es un método sencillo pero ineficiente. La idea es elegir una solución poco reflexionada y luego mejorarla hasta obtener una solución adecuada.
3- Método voraz: trata de producir un resultado a partir de un conjunto de opciones candidatas. Paso a paso va seleccionando entre las soluciones posibles. En cada caso se escoge la solución que se considere como óptimo local, pretendiendo que al final del proceso uno obtiene el óptimo global.
4- Retroalimentación o backtracking que consiste en probar alternativas posibles, y si no se alcanza la solución, se retrocede al paso anterior y se prueba con otro valor.
5- Divide y vencerás o diseño descendente
6- Bottom-up
7- Etc.

Las más eficientes se basan en la técnica de **refinamientos sucesivos**, cuyo lema fundamental es 'divide y vencerás'. Esta técnica consiste en descomponer el problema complejo en subproblemas más sencillos y continuar aplicando este proceso de subdivisión hasta que cada subproblema pueda ser implementado en términos de una computadora. Esta metodología de diseño recibe el nombre de **diseño descendente o top-down o modular,** y la técnica de descomposición se llama, refinamientos sucesivos.
De esta forma el problema se descompone en un conjunto de módulos, donde cada módulo cumple una función bien definida y acotada, y se comunican entre ellos a través del envío de datos. Cada módulo es una sucesión ordenada y finita de acciones, es decir se expresa mediante un **ALGORITMO**.

- Definimos entonces que **un algoritmo es *un conjunto finito y no ambiguo, de acciones primitivas expresadas en un cierto orden lógico, que nos permiten resolver un problema en un tiempo finito***.

Un algoritmo debe ser:

- **finito**: tiene un nº determinado de pasos y termina en un lapso limitado de tiempo
- **preciso**: indica un orden de realización de las acciones
- **definido**: si se ejecuta más de una vez, siempre conduce al mismo resultado

Generalmente usamos pseudocódigo para expresar los algoritmos.

Características de un buen algoritmo:

CARACTERISTICA	CONCEPTO
Correcto	El algoritmo debe funcionar satisfaciendo lo solicitado
Eficiente	No debe desaprovechar recursos, en particular el espacio en memoria y tiempo de ejecución.
Claro	El algoritmo debe estar documentado para su comprensión
Confiable	Debe dar seguridad de resolver el problema.
Mantenible	Debe ser flexible ante cambios menores en la lógica de la solución.

3- Etapa de codificación

Llamamos codificación a la expresión del algoritmo de forma tal que pueda ser entendido por una computadora, es decir, a la escritura del mismo en un lenguaje de programación.
Para realizar la conversión se reemplaza cada instrucción del pseudocódigo por la correspondiente del lenguaje de programación elegido.
Todo lenguaje de programación está provisto de reglas sintácticas y semánticas. La sintaxis especifica cómo escribir un programa usando símbolos, palabras claves y reglas de construcción (llamadas gramáticas, como la BNF); la semántica se refiere a qué significa cada símbolo y cada construcción del lenguaje.
Podemos definir ahora qué es un programa:

- **Un programa es la especificación de un algoritmo en un lenguaje de programación determinado, conforme sus reglas sintácticas y semánticas**

El resultado de esta etapa es el programa y representa **CON QUÉ** resolvemos el problema.

4- Etapa de compilación y ejecución.

Una vez que el algoritmo se ha codificado, se ha convertido en lo que llamamos, un programa fuente. Se lo introduce en la memoria de la computadora mediante un editor de

texto y con el teclado. Luego se lo graba en disco duro. Para poder almacenarlo en la máquina se debe traducir a lenguaje de máquina. Este proceso lo hace el compilador y el S.O.
En el momento de la compilación, se analiza la sintaxis y la semántica del programa fuente. Si hay errores, aparecen señalados y acompañados de comentarios que nos permiten corregirlos. Cuando no haya más errores de compilación, recién se obtiene el programa objeto.
El programa objeto es el que vamos a poder ejecutar. Para ello debemos indicarle al S.O. que lo ejecute, momento en el cual realiza una etapa de enlace o link-edición, que significa que se carguen en memoria Ram, el programa objeto junto con programas del compilador, y finalmente se obtiene el programa ejecutable.

5- Etapa de verificación

Una vez compilado el programa, debe verificarse la ejecución del mismo, probando si conduce al resultado esperado. Esta comprobación se hace con datos representativos del mundo real. Generalmente se aconseja probar varias veces el programa con:

- valores normales
- valores especiales
- valores extremos

En esta etapa suelen hacerse pruebas de escritorio: es decir se sigue la lógica del programa a mano, para un conjunto de datos. Esto permite probar el algoritmo obtenido, siempre que el flujo de datos sea medianamente pequeño. No asegura que los resultados sean correctos, pero muestra de manera genérica el recorrido de los datos en el programa de acuerdo a la lógica de control. No asegura que los resultados del mismo estén libres de error.
Esta etapa recibe el nombre de depuración de errores. Los errores que se pueden detectar son de lógica, de cálculo, de ejecución, etc.

6- Documentación y Mantenimiento

La etapa de documentación es la más pesada, pero de gran importancia para la realización de la etapa de mantenimiento.
Documentar significa describir los pasos que se llevan a cabo en la resolución de un problema. La documentación puede ser interna: la que está incluida en las líneas del programa y describen datos y acciones del mismo; externa: la que incluye el análisis previo del problema, el pseudocódigo, el manual del usuario, etc. .
Los programas documentados son más fáciles de leer, corregir, de modificar y mantener.

Resumiendo:

Planteo	**Concepto**
ESTRATEGICO	Consiste en la comprensión del problema a resolver, es decir saber **qué** se debe obtener para satisfacer a lo pedido.

METODOLOGICO	Es analizar **cómo** se puede alcanzar la solución al problema planteado, utilizando los recursos de resolución de problemas asociados al tipo de problema.
HERRAMENTAL	Elegir **con qué** recursos se expresan los pasos a seguir para obtener la solución: en forma gráfica, seudocódigo ó con un leguaje de programación.