¿Ingeniería Informática?

Daniel González Peña
Miguel Reboiro Jato

Ourense, 2 de septiembre de 2022

Universidade Vigo
ESCOLA SUPERIOR DE ENXEÑERÍA INFORMÁTICA

Índice

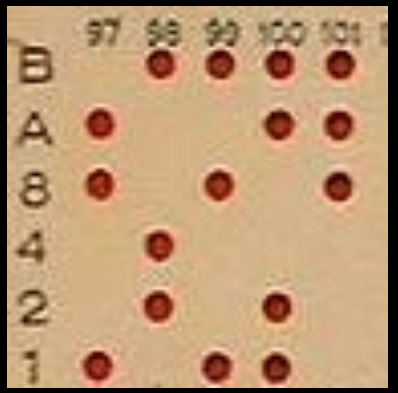
1. El Computador
2. Programación
3. Sistemas de Información
4. Ingenería del Software
5. Licencias de Software
6. ¿Dónde puedes llegar?
7. ¿Preguntas?

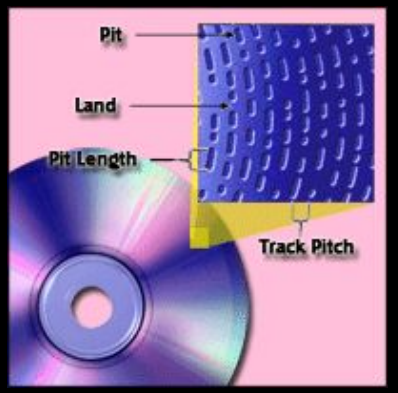
El Computador

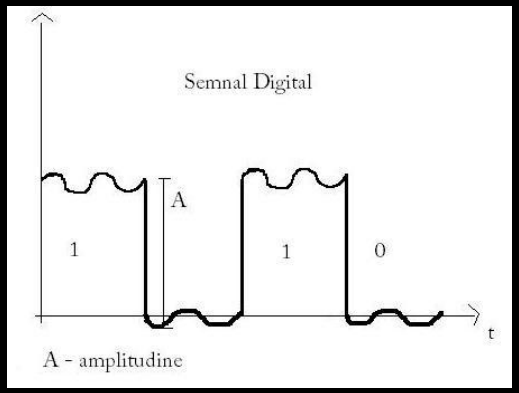
¿Unos y ceros?

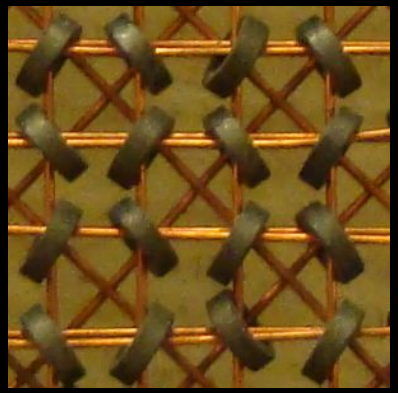
Toda la información que maneja un computador se representa como unos y ceros

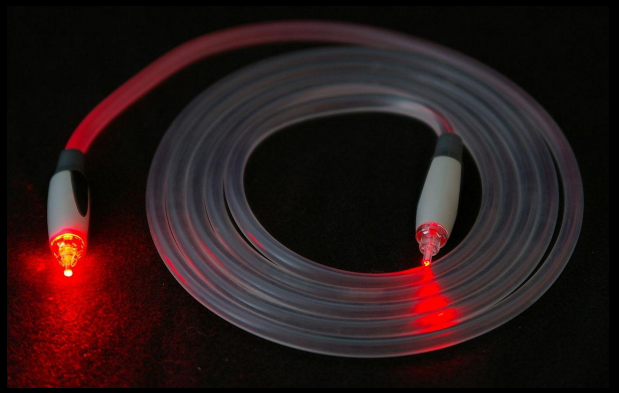
El motivo es la simplicidad de representación en distintos medios (p.ej. magnéticos, ópticos, eléctricos, lumínicos, etc.)











* Imágenes: Wikipedia, Wikimedia commons y http://www.cyberlink.com/stat/bd-support/enu/next-gen-tech.jsp

¿Unos y ceros?

Solo con unos y ceros es posible representar casi cualquier información:

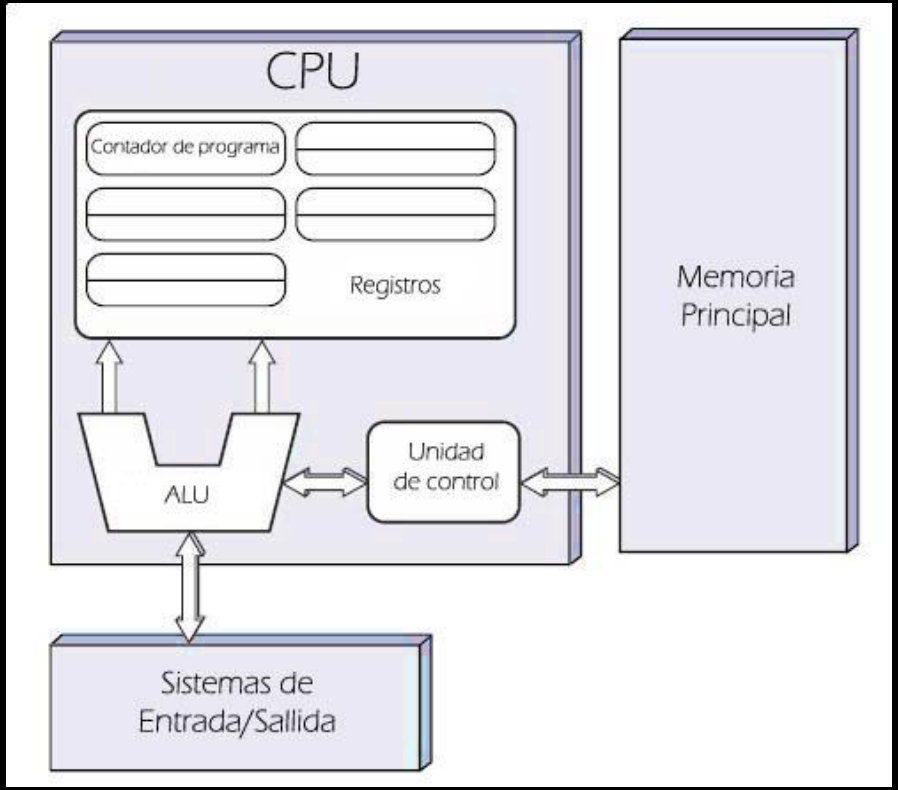
- Números (binario):
  01011001 => 89
- Letras (codificaciones de caracteres):
  01011001 => Y [ASCII]
- Instrucciones (ensamblador):
  0000010000000101 => ADD al, 05 [x86]

Estructura de un computador

La base de la estructura de un computador actual fue definida en 1945 por Von Neumann

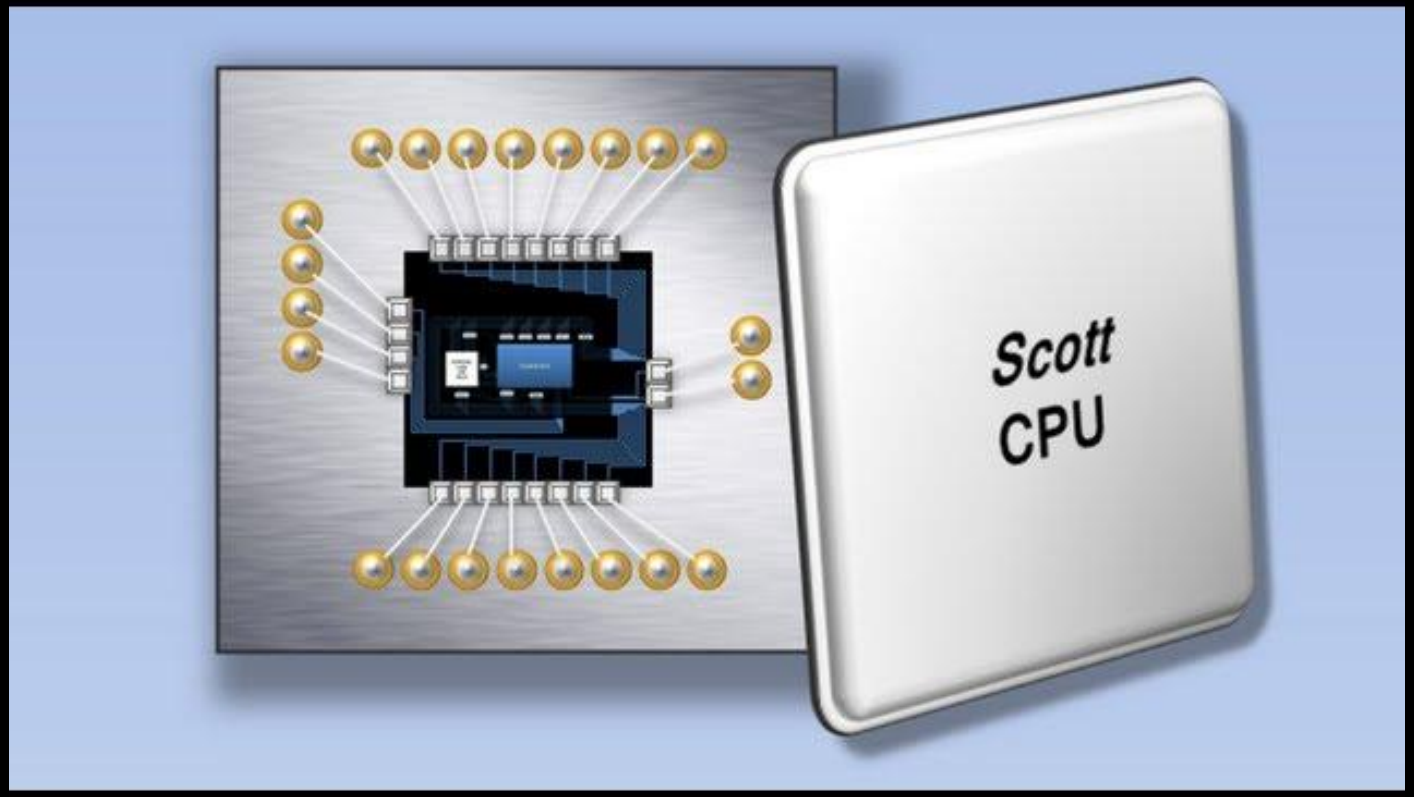
Componentes

- CPU
- Memoria (RAM)
- Entrada/Salida

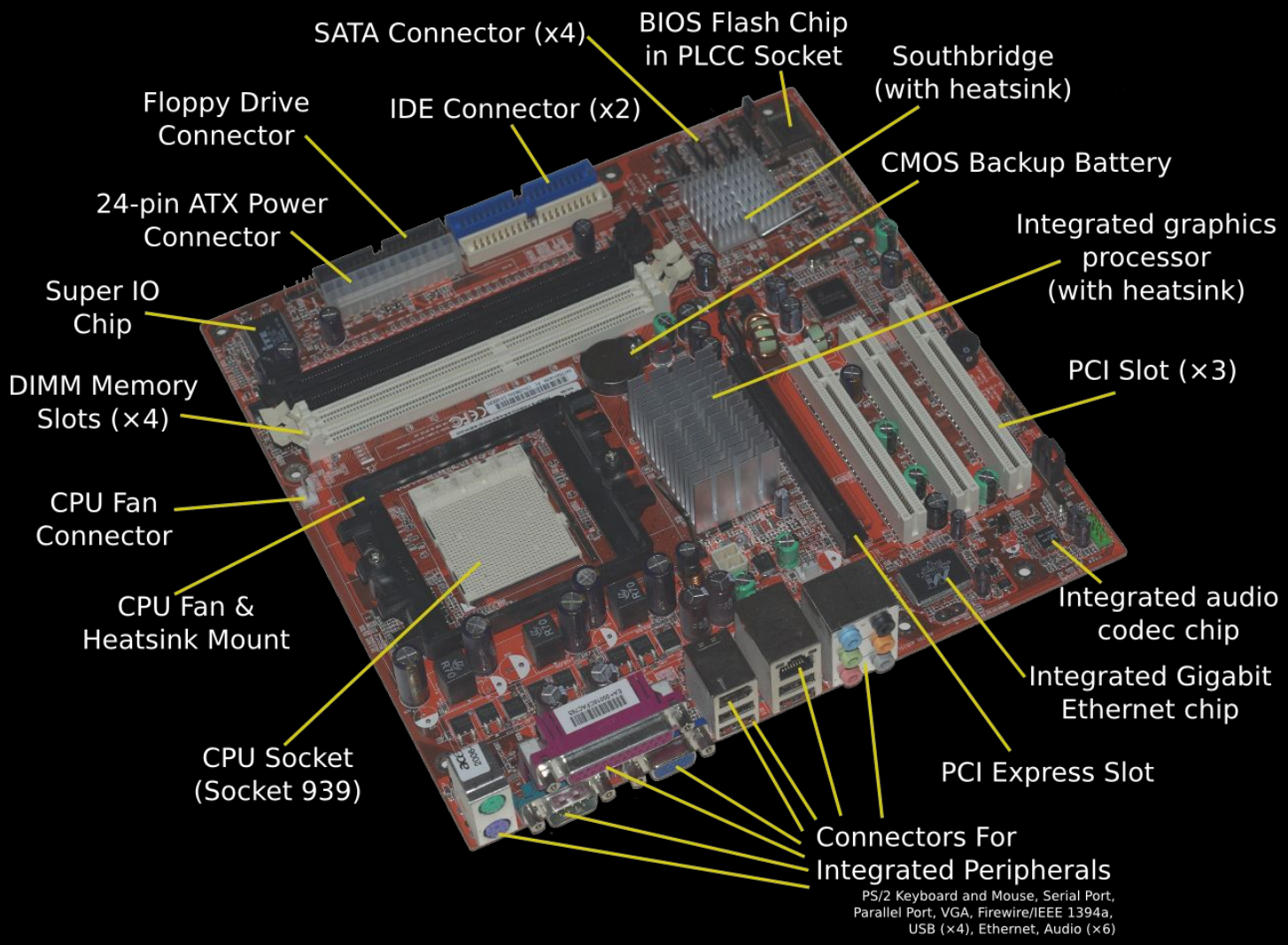


* Imagen: Wikipedia

¿Cómo funciona?



Componentes de un PC actual

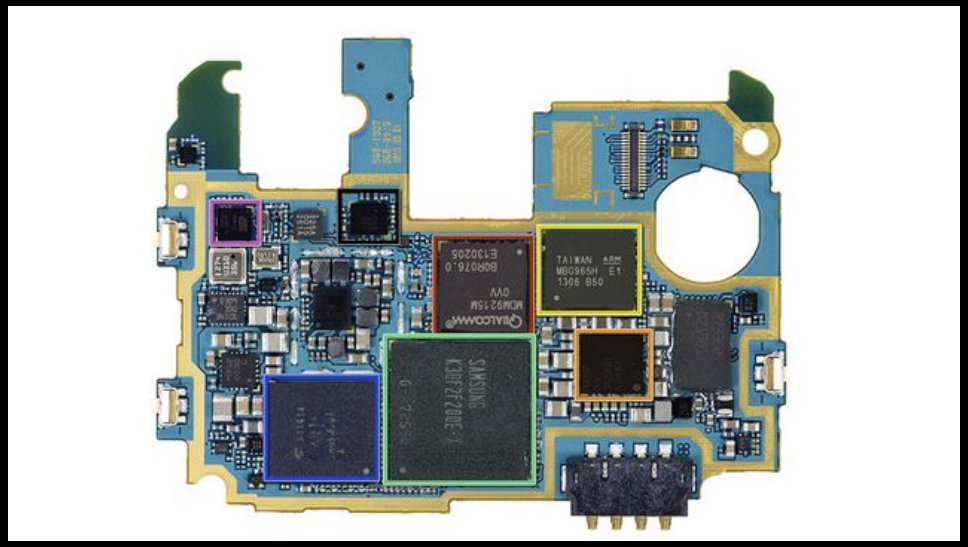


* Imagen: Wikipedia

Componentes de un PC actual

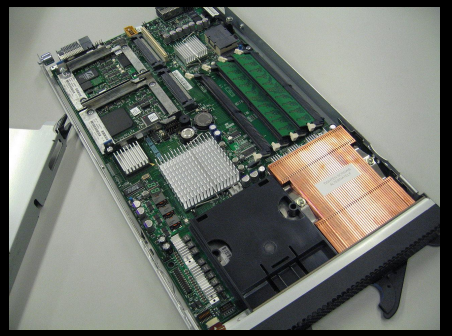
Samsung Galaxy S4 motherboard con partes indicadas

https://es.ifixit.com/Desmontaje/Samsung+Galaxy+S4+Teardown/13947



Tipos de computadores

Hoy en día existen muchos tipos de computadores: televisores, móviles, relojes, PCs, videoconsolas, servidores, supercomputadores.



Supercomputador “Blue Gene/P” de IBM

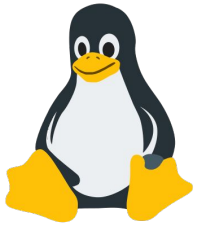
Los supercomputadores son, básicamente, muchos computadores pequeños unidos. Su fuerza estriba en ejecutar muchas tareas al mismo tiempo.

* Imágenes: Wikipedia

El sistema operativo

Es un programa que siempre se está ejecutando y que facilita el trabajo al resto de programas.

- Permite que varios programas se ejecuten a la vez [Gestión de procesos]
- Trabaja con los diferentes modelos de hardware (p.ej: distintos discos duros) de forma que los programas no tengan que preocuparse de los detalles. Cada fabricante programa un “driver” [Abstracción hardware]
- Permite leer y escribir información desde y hasta el exterior [Gestión de la Entrada/Salida]
- Permite que varios programas usen la memoria RAM del computador sin notar que hay otros programas usándola (piensan que tienen toda la memoria para ellos) [Gestión de memoria]



macOS

UNIX®

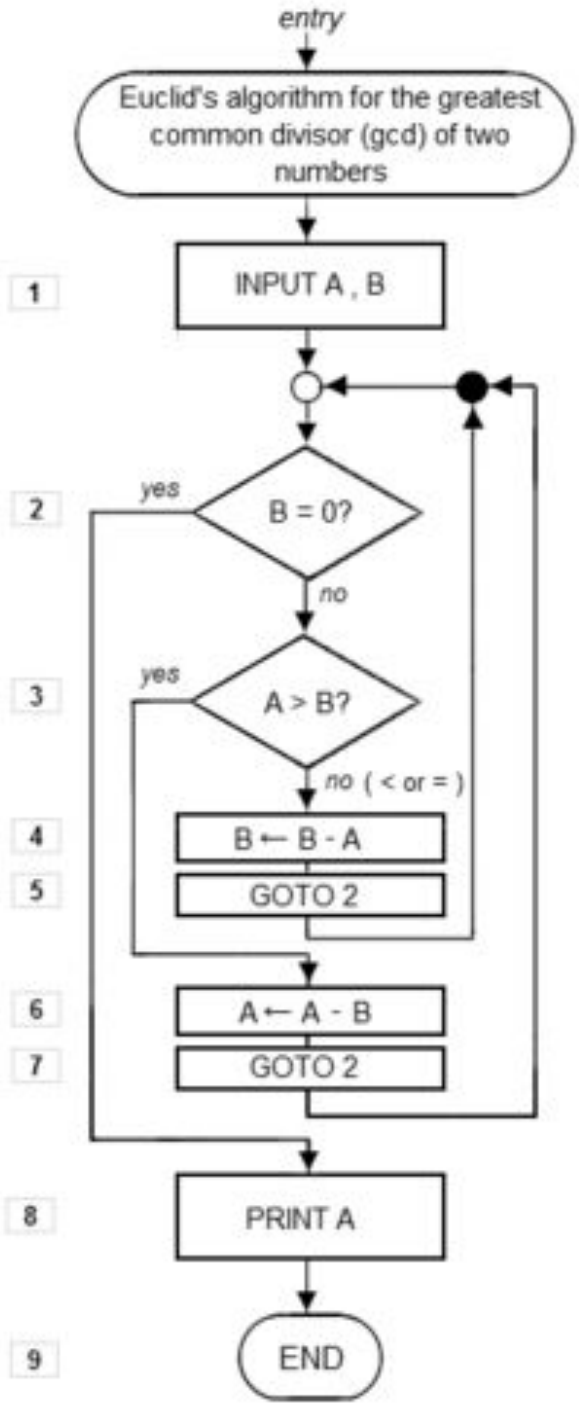
Programación

Algoritmos

Un algoritmo (concepto matemático y de ciencias de la computación) define un procedimiento para llevar a cabo una tarea

Los “programas de ordenador” o software están formados por algoritmos ejecutables

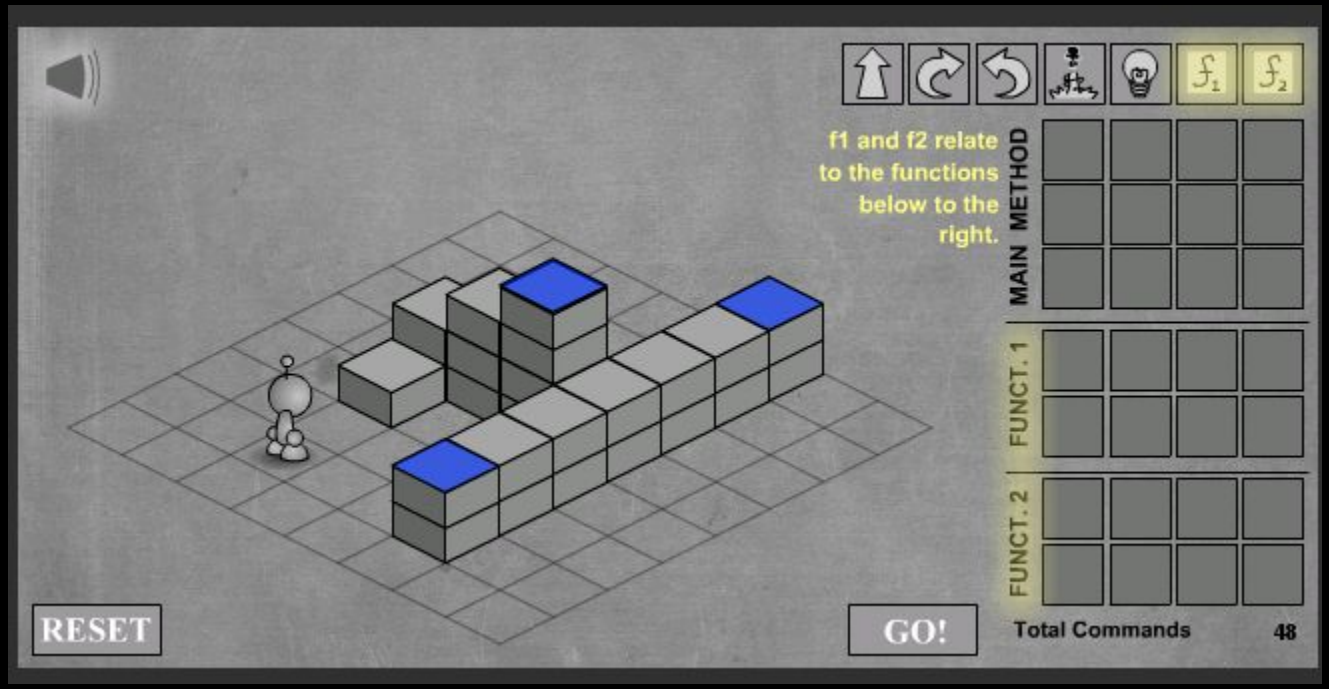
Antes de escribir el programa, debemos pensar el algoritmo



* Imagen: Wikipedia

Algoritmos

¡Dale un algoritmo al robot para que cumpla su objetivo!



La programación

Un “programa” consiste en un conjunto de instrucciones que son ejecutadas por el computador para hacer una tarea concreta. Es un algoritmo preparado para ser ejecutado en un computador

¿Cómo indicarle el programa a un computador?

En **código máquina:** lista de instrucciones con sus parámetros ¡en binario!

¿En binario? -> difícil!!

Ensambladores

Mediante un **ensamblador** podemos ponerle un nombre a cada instrucción y no trabajar en binario

El ensamblador traduce línea a línea a las instrucciones en binario para enviar al procesador

Sigue siendo difícil!

- Difícil de escribir
- Difícil de mantener
  - Saltos (GOTO, JMP) = código “espaguetti”
- Depende de cada máquina
  - No todos las CPU tienen las mismas instrucciones

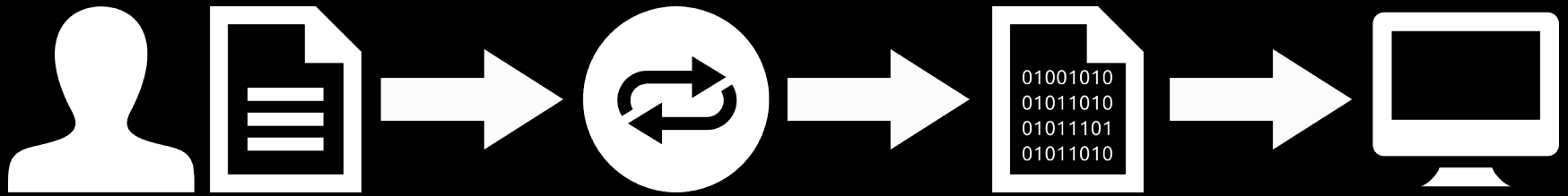
* Imagen: Wikipedia

```
MONITOR FOR 6802 1.4          9-14-80  TSC ASSEMBLER  PAGE    2


C000                    ORG     ROM+$0000 BEGIN MONITOR
C000 8E 00 70  START    LDS     #STACK

                ****************************************
                * FUNCTION: INITA - Initialize ACIA
                * INPUT: none
                * OUTPUT: none
                * CALLS: none
                * DESTROYS: acc A

0013            RESETA  EQU     %00010011
0011            CTLREG  EQU     %00010001

C003 86 13      INITA   LDA A   #RESETA   RESET ACIA
C005 B7 80 04           STA A   ACIA
C008 86 11              LDA A   #CTLREG   SET 8 BITS AND 2 STOP
C00A B7 80 04           STA A   ACIA

C00D 7E C0 F1           JMP     SIGNON    GO TO START OF MONITOR

                ****************************************
                * FUNCTION: INCH - Input character
                * INPUT: none
                * OUTPUT: char in acc A
                * DESTROYS: acc A
                * CALLS: none
                * DESCRIPTION: Gets 1 character from terminal

C010 B6 80 04   INCH    LDA A   ACIA      GET STATUS
C013 47                 ASR A             SHIFT RDRF FLAG INTO CARRY
C014 24 FA              BCC     INCH      RECIEVE NOT READY
C016 B6 80 05           LDA A   ACIA+1    GET CHAR
C019 84 7F              AND A   #$7F      MASK PARITY
C01B 7E C0 79           JMP     OUTCH     ECHO & RTS

                ****************************************
                * FUNCTION: INHEX - INPUT HEX DIGIT
                * INPUT: none
                * OUTPUT: Digit in acc A
                * CALLS: INCH
                * DESTROYS: acc A
                * Returns to monitor if not HEX input

C01E 8D F0      INHEX   BSR     INCH      GET A CHAR
C020 81 30              CMP A   #'0       ZERO
C022 2B 11              BMI     HEXERR    NOT HEX
C024 81 39              CMP A   #'9       NINE
C026 2F 0A              BLE     HEXRTS    GOOD HEX
C028 81 41              CMP A   #'A
C02A 2B 09              BMI     HEXERR    NOT HEX
C02C 81 46              CMP A   #'F
C02E 2E 05              BGT     HEXERR
C030 80 07              SUB A   #7        FIX A-F
C032 84 0F      HEXRTS  AND A   #$0F      CONVERT ASCII TO DIGIT
C034 39                 RTS

C035 7E C0 AF   HEXERR  JMP     CTRL      RETURN TO CONTROL LOOP
```

Lenguajes de programación

Un lenguaje de programación trata de acercarse al lenguaje del humano

Un **compilador** o **intérprete** (otro programa) se encarga de traducir el programa al lenguaje de la máquina



Surgen los LPs Estructurados (ej: C)

- Más restrictiva (no permite saltos a cualquier lugar GOTO) -> más fácil de mantener

Luego los LPs Orientados a Objetos (ej: Java)

- Facilitan más la reutilización de código y un diseño más cercano a la realidad del problema

Otros lenguajes:

- Declarativos: más específicos, pero se dice qué se busca no cómo buscarlo (ej: SQL)
- Funcionales: más matemáticos, orientados a funciones sin efectos laterales
- De marcado: Se emplean para representar documentos con información (HTML, XML, etc.)

* Imágenes: iconmonstr (composición propia)

Lenguajes, lenguajes, lenguajes

```c
/* Hello World program */

#include<stdio.h>

main()
{
    printf("Hello World");

}
```

Programa en C

```sql
SELECT NOMBRE, APELLIDOS
FROM EMPLEADOS
WHERE SALARIO > 1000
```

Sentencia en SQL

```java
class HelloWorld {
    public static void main(String args[])
    {
        System.out.println("Hello world!");
    }
}
```

Programa en Java

```python
#!/usr/bin/python
# My first python program

print "Hello, World!\n"
```

Programa en Python

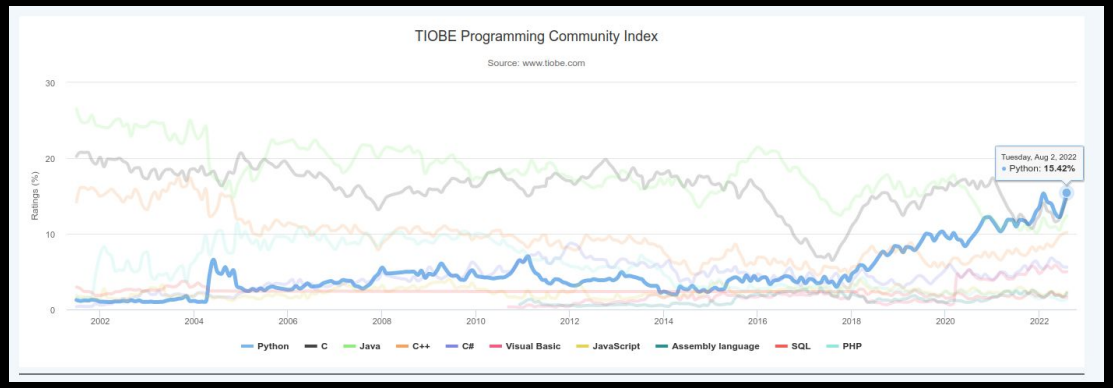
```html
<html>
<body>
Hola mundo
</body>
</html>
```

Página web en HTML

Lenguajes, lenguajes, lenguajes

Popularidad de los lenguajes actuales (agosto 2022)

Aug 2022	Aug 2021	Change	Programming Language	Ratings	Change
1	2	↑	Python	15.42%	+3.56%
2	1	↓	C	14.59%	+2.03%
3	3		Java	12.40%	+1.96%
4	4		C++	10.17%	+2.81%
5	5		C#	5.59%	+0.45%
6	6		Visual Basic	4.99%	+0.33%
7	7		JavaScript	2.33%	-0.61%
8	9	↑	Assembly language	2.17%	+0.14%
9	10	↑	SQL	1.70%	+0.23%
10	8	↓	PHP	1.39%	-0.80%

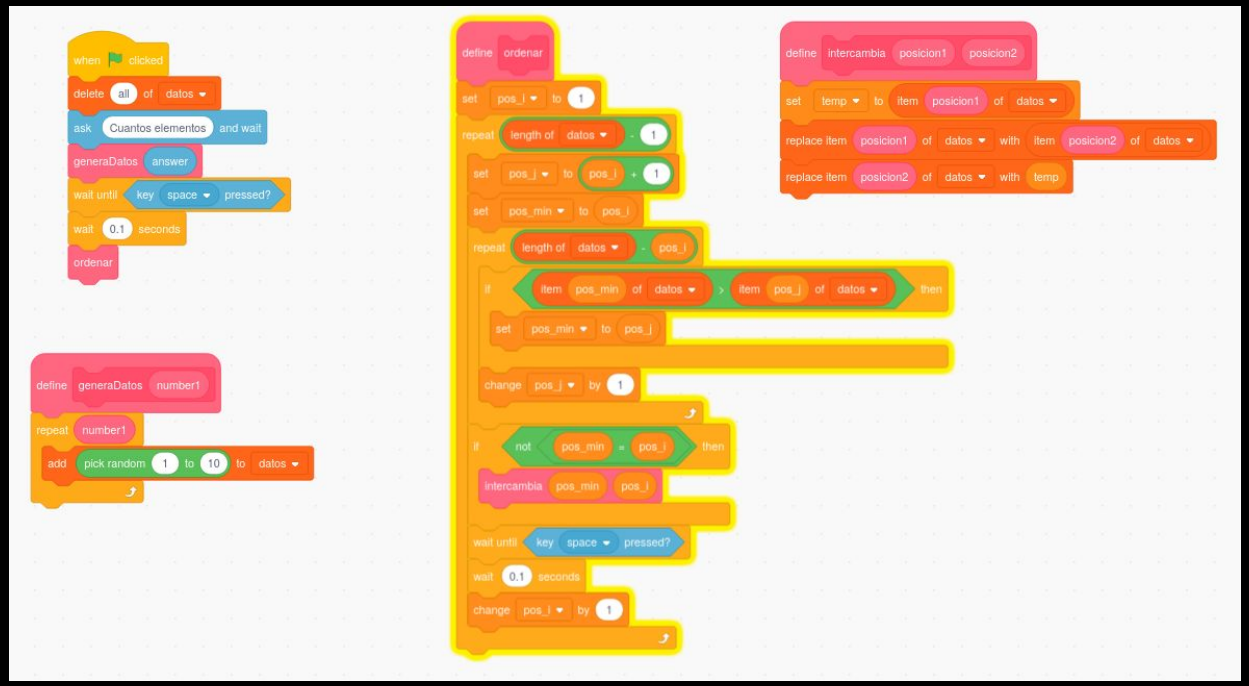


* fuentes: http://www.tiobe.com

Programación estructurada

¡Prueba a programar de forma estructurada con SCRATCH!

Problema a resolver: “Ordenar una lista de números”



Sistemas de información

¿Qué partes típicas tienen los sistemas de información? (I)

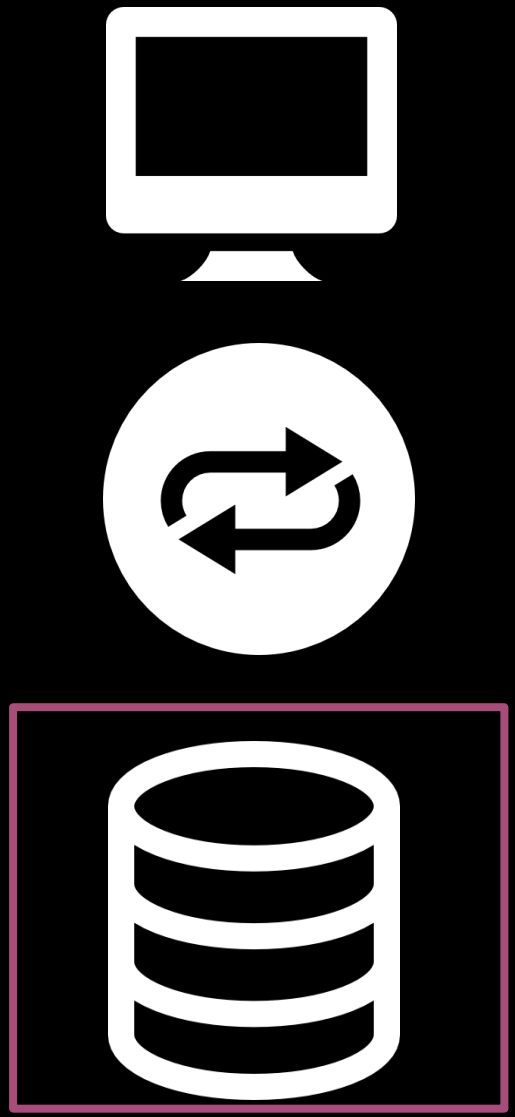
Base de datos

Almacena la información de forma persistente y de una forma organizada para ser fácil y rápido acceder desde el programa a desarrollar

Por ejemplo: Facebook. En su base de datos están todas las publicaciones, todos los comentarios, las relaciones de amistad entre todos los usuarios

Suele estar gestionada por un programa especial (Oracle, MySQL, Informix, etc.)

Suelen accederse mediante un lenguaje estándar y específico: SQL



* Imágenes: iconmonstr

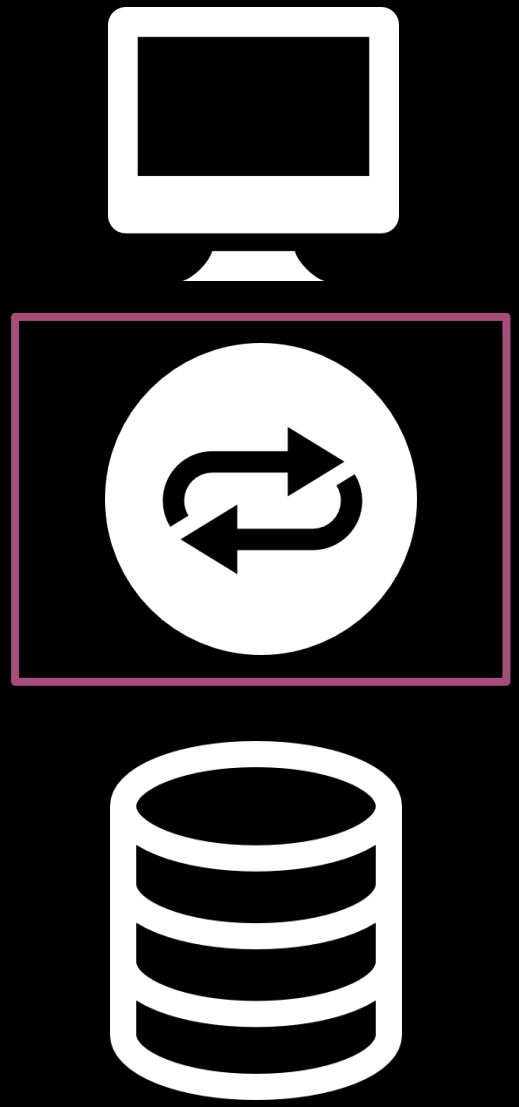
¿Qué partes típicas tienen los sistemas de información? (II)

Lógica de negocio

Todos los algoritmos y cálculos que implementan las tareas del programa

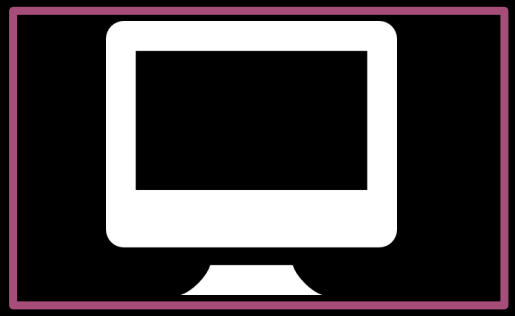
Ejemplo: Facebook. Calcular posibles amigos, enviar notificaciones a tu mail, buscar la imagen representativa de un enlace que quieres compartir

Aquí se trabaja sobre todo con los lenguajes de programación de propósito general: Java, C++, PHP, etc.



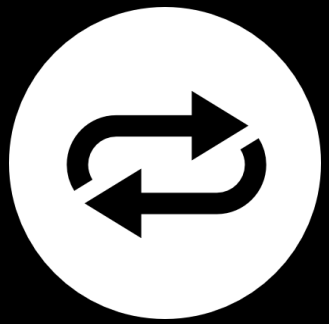
* Imágenes: iconmonstr

¿Qué partes típicas tienen los sistemas de información? (III)



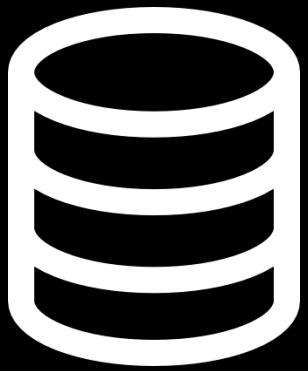
Interfaz de usuario

Básicamente recoge datos, gestos, clics, etc. del usuario y presenta resultados



Se suelen distinguir dos tipos:

- Web (en el navegador)
- Gráficas (ventanas, en escritorio o móvil)
- En modo texto (“letra blanca, fondo negro”)

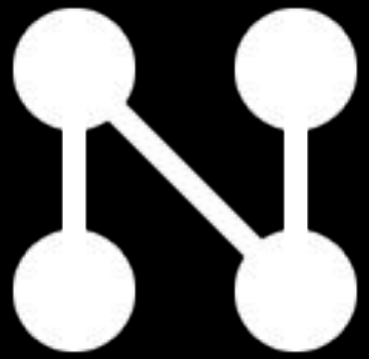


Ejemplo: Facebook. La interfaz gráfica de usuario es la página web que nos muestra en nuestro navegador la red social

* Imágenes: iconmonstr

La red

La mayoría de los sistemas de información no funcionan en una misma máquina



Muchos de ellos funcionan bajo el paradigma cliente-servidor

Ejemplo. Facebook. La web se muestra en la máquina del usuario (PC o móvil), el procesamiento se hace en el servidor, donde también está la base de datos de Facebook

Una de las redes más importantes es Internet, pero también las redes locales que hay en cada casa u organización

Las tecnologías de red permiten a los programadores enviar y recibir datos a máquinas remotas sin apenas preocuparse de cómo llegar a ellas

* Imágenes: iconmonstr

Ingeniería de Software

Ingeniería del Software

La Ingeniería del Software trata sobre el proceso de desarrollo de software

El objetivo principal es desarrollar aplicaciones en tiempo y presupuesto, minimizando riesgos y asegurando la calidad

Aunque existen muchas metodologías, todas ellas suelen contener las siguientes actividades

- Recogida de requisitos
- Análisis
- Diseño
- Implementación
- Pruebas
- Despliegue
- Mantenimiento

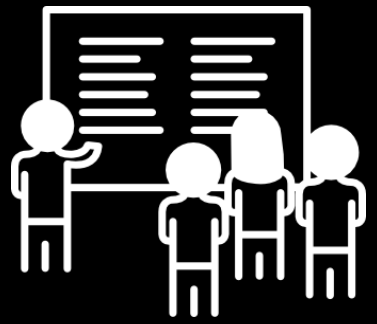
Actividades Principales de la IS

Recogida de Requisitos

Identificar las necesidades del cliente y las funciones de la aplicación

Análisis

Estudiar en detalle los requisitos definiendo bien los conceptos del dominio del problema. También se planifica el desarrollo



* Imágenes: The noun project (Hedie Assadi Joulaee & Ainsley Wagoner)

Actividades Principales de la IS

Diseño

Definir la arquitectura de la aplicación y la estructura de las distintas partes de la misma, hasta un llegar a un alto nivel de detalle

Implementación

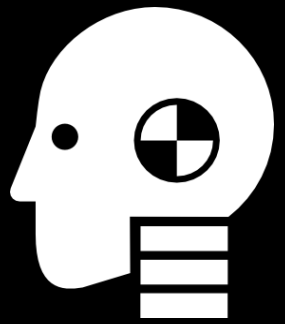
Programar la aplicación siguiendo el diseño definido

* Imágenes: The noun project (Samuel Q. Green & Mayene de Leon)

Actividades Principales de la IS

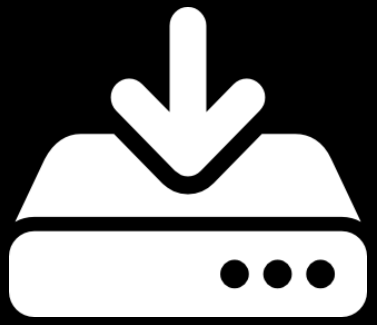
Pruebas

Comprobar que la aplicación cumple con los requisitos



Despliegue

Instalar la aplicación en los sistemas del cliente. Suele incluir la formación de los usuarios



* Imágenes: The noun project (Luis Prado & Alain LOUBET)

Actividades Principales de la IS

Mantenimiento

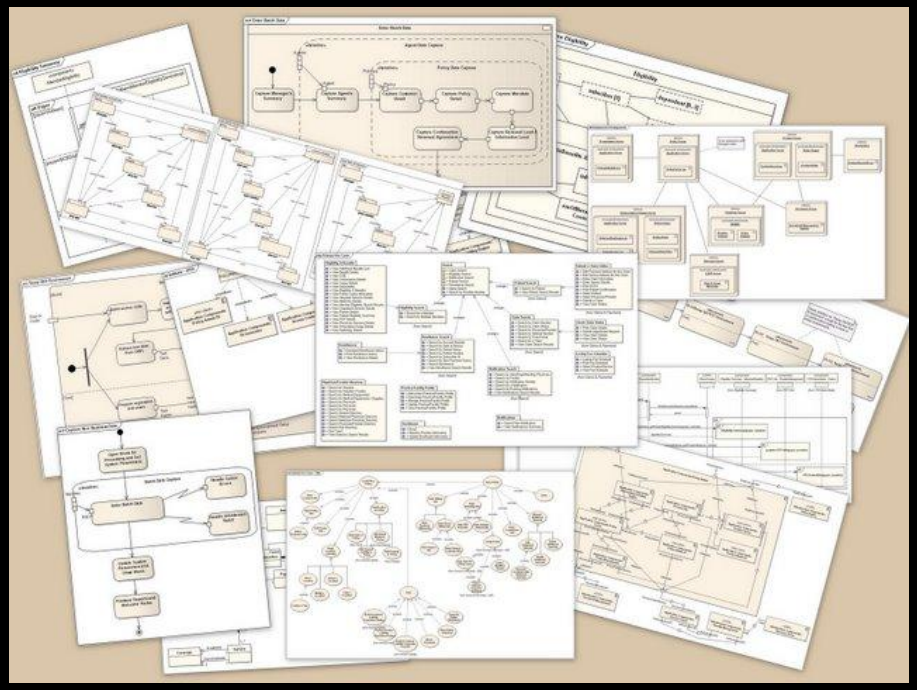
Corregir los errores detectados durante el funcionamiento de la aplicación y añadir algunas funcionalidades menores requeridas por el cliente

* Imágenes: The noun project (Dan Hetteix)

Tareas Principales de la IS

Documentación

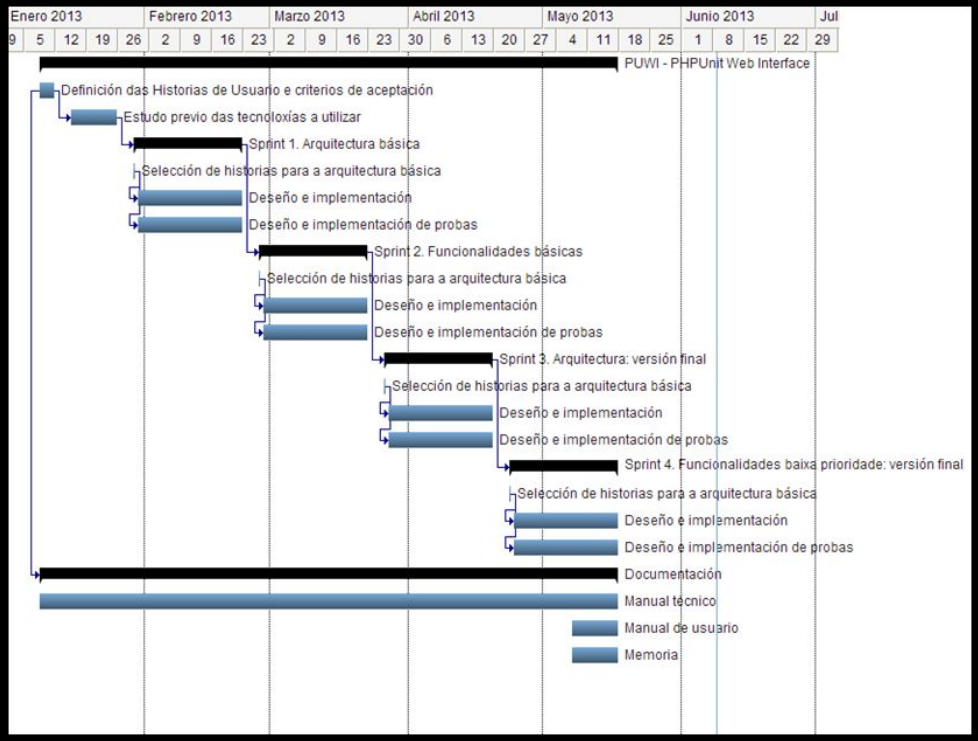
Existen lenguajes y diagramas técnicos (p.ej. UML, diagramas de entidad-relación para la base de datos, etc.) específicos para la documentación del diseño de una aplicación



Tareas Principales de la IS

Planificación

Permite estimar la duración del proyecto y reducir los riesgos asociados al incumplimiento de plazos



Tareas Principales de la IS

Gestión de la Configuración del Software

Existen una serie de tareas que son necesarias para asegurar la integridad del software desarrollado

- Control de versiones
- Backup
- Integración (continua)
- Seguridad
- Construcción
- Gestión de la configuración
- *Bug-tracking*

Licencias de Software

Licencias de Software

La selección de una licencia adecuada es muy importante a la hora de publicar una aplicación

Se puede distinguir entre

- Software privativo (de código cerrado, aunque puede ser gratuito o *freeware*)
- Software libre (de código abierto, suele ser gratuito)

Actualmente existen muchos tipos de licencias de software libre

- GPL, LGPL, AGPL, BSD, MIT, Mozilla, etc.

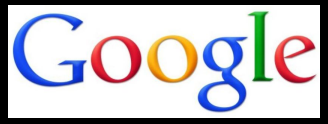
¿Dónde puedes llegar?

Antiguos alumnos

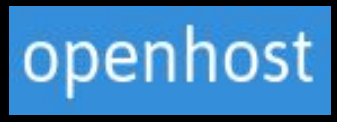
Centros de Investigación

Organismos Públicos

Empresas Privadas



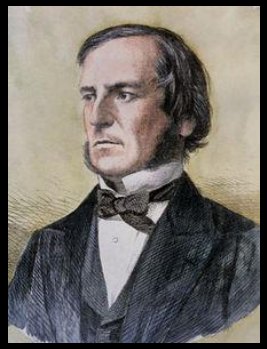
Emprendedores



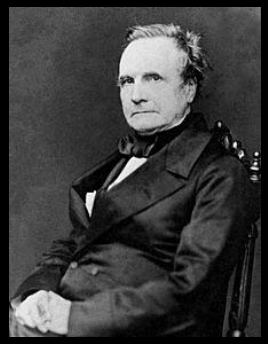
¿Preguntas?

Un poco de historia

Personajes clave

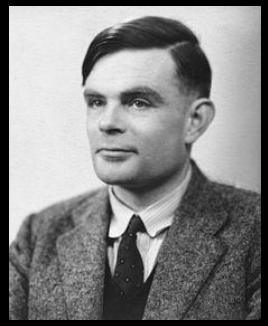


George Boole
Lógica booleana (1847)



Charles Babbage y Ada Lovelace
- Matemáticos y colaboradores
- “Máquina Analítica” (1830)
- “Primera programadora”

C. E. Shannon
Aplicó la lógica booleana al diseño de circuitos electrónicos (1937)

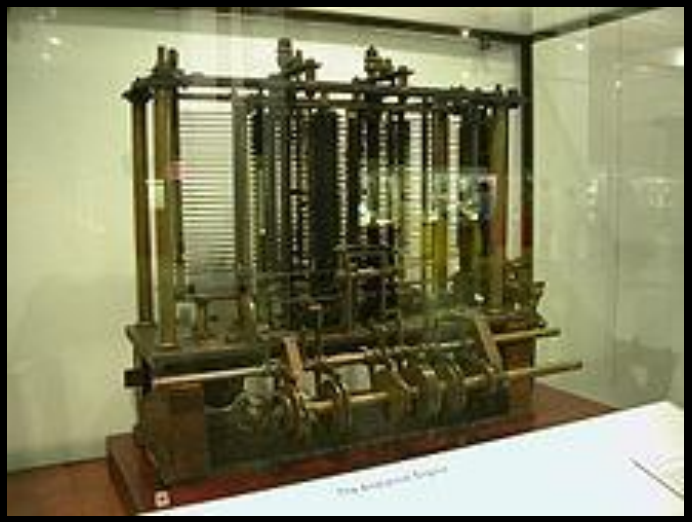


Alan Turing & Alonzo Church
Formalizaron la idea de “algoritmo” y
Estudiaron qué es computable por un ordenador (1936)
Máquina de Turing (1937)

John Von Neumann
Arquitectura actual de la mayoría de ordenadores (1945)

* Imágenes: Wikipedia

Los primeros computadores



Máquina Analítica

(Charles Babbage, 1837)

Mecánica.

Sistema de numeración: Decimal

Programación: Tarjeta perforada

Turing-completa: Sí

Nota: Nunca construída.

Z3

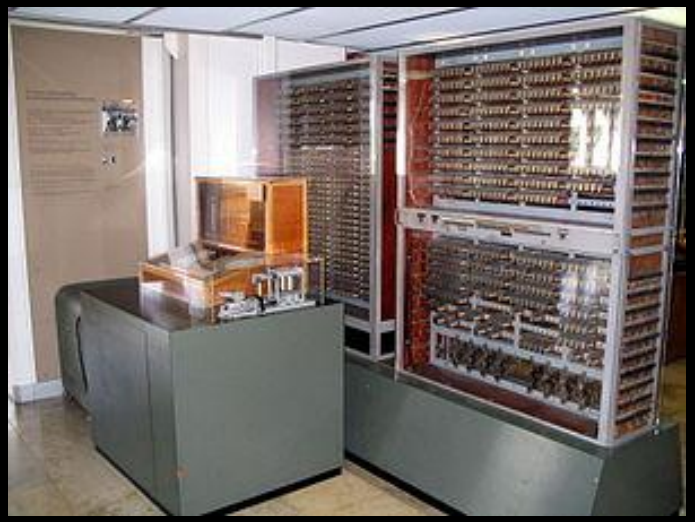
(Konrad Zuse, 1941)

Electro-Mecánico

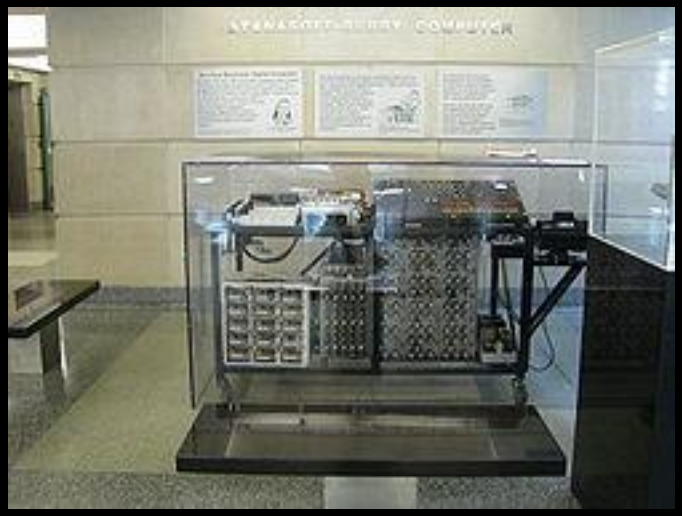
Sistema de numeración: Binaria

Turing-completa: Sí* (demostrado en 1998)

Programación: Tarjeta perforada



Los primeros computadores



Atanasoff-Berry

(J. Mauchly y J.P Eckert, 1942)

Electrónico.

Sistema de numeración: Binaria

Programación: No, propósito específico

Turing-completa: No

Colossus - Mark 1

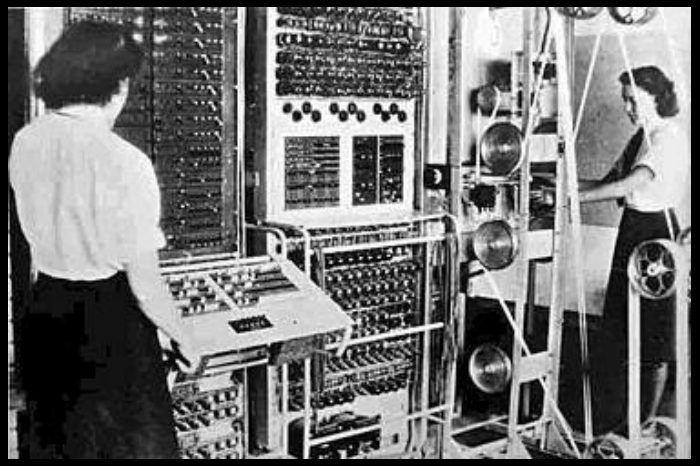
(Tommy Flowers, 1943)

Electrónico.

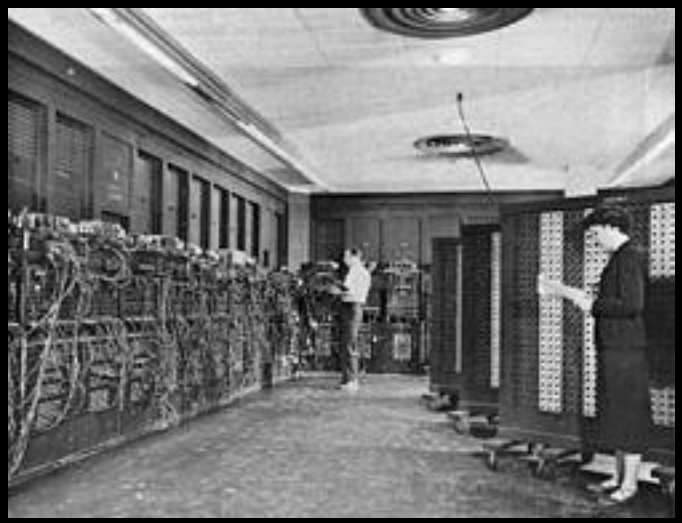
Sistema de numeración: Binaria

Programación: Cables e interruptores

Turing-completa: No



Los primeros computadores



ENIAC

(Howard Aiken, 1946)

Electrónico

Sistema de numeración: Decimal

Programación: Cables e interruptores

Turing-completa: Sí

Manchester Baby

(F.C. Williams, T. Kilburn y G. Tootill, 1948)

Electrónico.

Sistema de numeración: Binaria

Programación: Almacenado en memoria

Turing-completa: Sí