

■ vagrant
■ Packer
■ Docker } Hacen cosas similares que docker

■ Metodología agil

- métodos de empaquetamiento
- analogía de la maquila
- agil con devops = agil + it
- docker es multi & cross platform
- instala dependencias
- Multi-stage build

Peculiaridades de Git

2019-08/20

- git fetch = Hacer un vistazo a los cambios
- ssh-keygen rsa
- git pull
- git clone
- git log
- git reset --hard
- No es recomendable borrar el .git

Docker

- No es una máquina virtual
- Es un contenedor.
- Un ambiente aislado

Dockerfile = permite describir el ambiente y genera un "dockerImage"

Docker run = equivalente a correr un ejecutable

Syntax Docker file.

version linux

- FROM python: alpine 3.6
Imagen de python
- RUN pip install colorama
copiando desde el directorio
- COPY app.py / código

• imágenes de librería = comienzan con -.

directorío
git demo

```
docker build -rm -f "Dockerfile" -t git demo
docker build -t
docker images | grep
docker run -it contador:0.0.
-i -t
```

list (0001 - 0100)

$$L = [0, 0, 0, 1, 0, 1, 0, 0]$$

$$\text{len}(L) = 8$$

$$\text{Posiciones} = 8$$

for i in range(8):

$$\text{suma} = \sum_{i=0}^7 (L[\text{Posiciones}]) * 2^{**\text{contador}}$$

$$(0 * 2^0) + (0 * 2^1) + (1 * 2^2) + (0 * 2^3) + (1 * 2^4) + (0 * 2^5) + \\ (0 * 2^6) + (0 * 2^7) \quad \text{contador de } 0-7$$

IC

Memory

ALU

Registers

RAM

PARENT manufacturer
[IC] build - date

PARENT (inherits IC)
[Memory] 4 bit cu

Inherits IC
[ALU] : zero
outflow
negative

Registers
[RAM]

inherits IC
[CU]

① separar y ordenar

PARENT
[IC]

inherits, IC
[CU]

Inherits IC
[ALU]

(inherits, IC)

Inherits IC
[Memory]

Registers
[RAM]

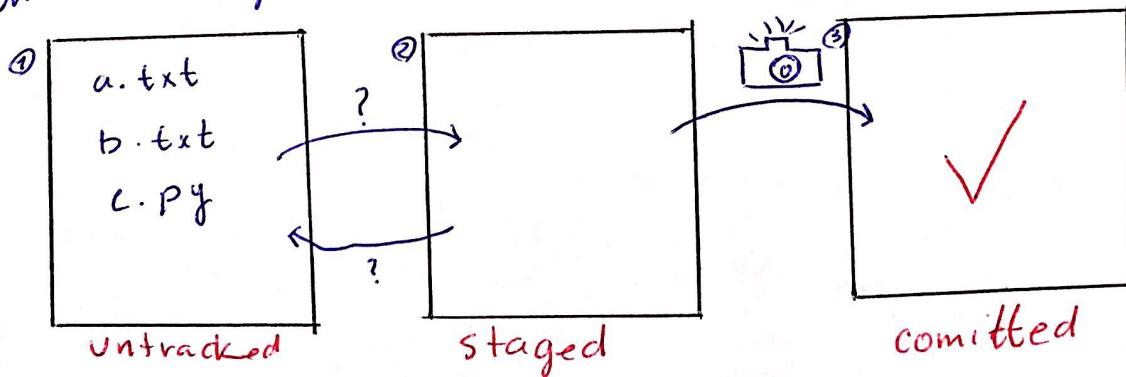
• instructions in "string" format.

Git

2019/08/08

- Bash - Lenguaje interpretado para interactuar con el sistema operativo.
 - Git está hecho en C.
 - 1) git init
 - 2) touch A.txt
 - 3) cat A.txt
 - 4) cat > A.txt { concatenar a un archivo}
 - 5) echo "otra linea" >> a.txt
 - 6) vim a.txt
 - 7) ls -la (a, es muestra todo)
- bash-terminal default en OS tipo nix.
cmd-terminal de ~~os~~ windows
parentesis: Powershell o mezcla de las dos

untracked = git no sabe de ese archivo.



- git status = estamos en el primer estado { solo muestra si fue modificada o no.}
- git add a.txt = estamos en el caso 2
- git commit -m "primer commit en git"
- git log { aparece la bitácora del repositorio}
- git add . { añade todos los archivos}
- git diff { modificación específico}
- git diff ~ cache d
- git diff a.txt { cambios en archivo a.txt, se puede poner un directorio}

Redirecciónamiento

- `git diff`

■ `.gitignore`

`touch .gitignore` `*.log` ; para ignorar directorios o archivos o extensiones

`!otro.log` ; para hacer una excepción

Remote

- `git remote add origin git@gitlab:VFM/Directory`
sustituye esto →

TAREA:

Primitivas Data types:

- Desarrollar tipos de lenguajes de programación.
- Investigar y explicar.
- Ramas: intérpretes, compilados
- Imprimir "Hello world" en los programas

▲ Functions as FCO: en python todo es un objeto.

• Una función es un first class objects.

• functions are objects

• functions can be stored in data functions.

• functions can be passed to other functions

• functions can be defined with nested functions within

▲ Instance method = recibe parámetro

1/08/2019

- Campo estatíco = una variable en el scope de la clase
- Constructor = `_init_(self, params)`

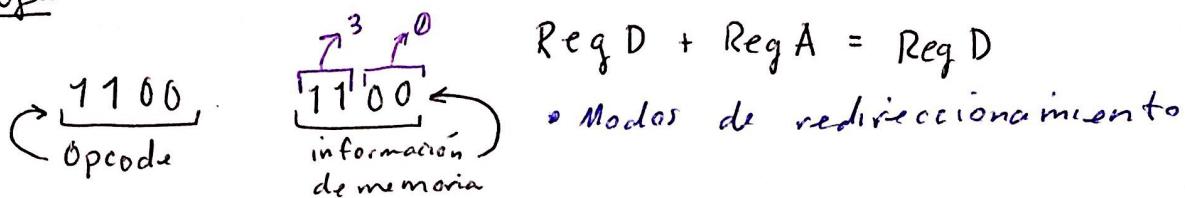
- Clases Padre = son las que heredan las clases no necesita "interfaces"

▲ `isinstance()` - returns a boolean

`isinstance(□, class name)`

El procesador hace un fetch, decode, execute.

Ej:



Python OOP

- Non-procedural Language: el código no se ejecuta necesariamente ordenadamente (SQL)
- Procedural Language: es secuencia y el código se ejecuta ordenadamente.

► Paradigmas de programación:

- uno de los primeros lenguajes de programación es C.
- un lenguaje que está basado en C es Ruby.
- C no hay listas, ni strings, no hay que preocuparse de la memoria.

► Programación Orientada a Objetos

- Sirve para resolver problemas específicos particulares.
- Una lista de listas para los votantes corre a problemas ya que no se puede asegurar la información correcta.
- Example
 - Person - Name, age, Address
 - Behaviour - how are you feeling
 - Email
 - Title
 - Subject
 - Body
 - Un perro
 - Birthdate

• Es diferente hablar de una clase y de un objeto.

- Objeto: la instancia de esa clase.

- Clase: el programa con atributos.

• En python todo es un objeto: strings, lists, dictionary; son objetos.

a = 'string'
se instanció
la clase string

► Classes in python - Camel Case

```
class Dog():  
    pass
```

* usar camelcase en las clases, es un convenio que se utiliza en lenguajes de programación

```
class Dog:  
    # initialize / instance attributes  
    def __init__(self, name, age):  
        self.name = name  
        self.age = age
```

self.name = name,
para el parámetro
objeto que instanciamos

Clases deben de tener

- Atributos / fields
- funciones / acciones / behaviour
- constructor __init__

* La sobre carga de constructores no existe en python.

* Sobre escribir la clase

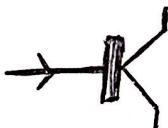
self = yo mismo

Programación III

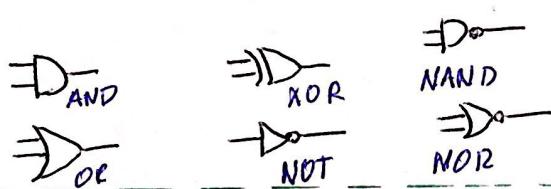
25/07/2019

■ CPU = Central Processing Unit

- el ser humano necesita contar, solían usar un abaco
- Computar = es hacer cosas repetitivas y tediosas
- 1800 se hace el primer intento de computar, Charles.
- El tubo del vacío, se uso una computadora electromecánica.
- El proyecto manhattan, usaban relays (un dispositivo que cierra o abre un circuito) [→ ↘ ↗ →], era un proceso electromecánico.
- Pregunto de examen; un bug se metía entre los relays y producían errores en ejecución.
- Principio de 1800, el tubo del vacío, funciona por medio de energía térmica.
- Un transistor, fue inventado por william shockley



• Compuertas lógicas



• Las compuertas están hechas de transistores

• Basic components of a CPU

■ control unit

■ ALU = Aritmética, comparaciones

■ Registro

• Instruction Registry

• Instruction Address

■ RAM = Random access Memory

■ Clock

• Binario así como en decimal

$$1 \quad 1024_{10}$$

$$1 \quad 1 \times 10^3 + 0 \times 10^2 + 2 \times 10 + 4 \\ * 10^0$$

$$1010_2$$

$$1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$$

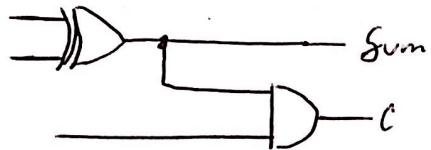
$$100 \text{ es } 10_{10}$$

• División computacional

$$\begin{array}{r} 17 \\ - 5 \\ \hline 12 \\ - 5 \\ \hline 7 \\ - 5 \\ \hline 2 \end{array}$$

así se divide en binario

• Medio sumador



$$2^2 = 4$$

$$2^{32} = 4294967290$$

• Overflow = cuando pasan

• Punto flotante

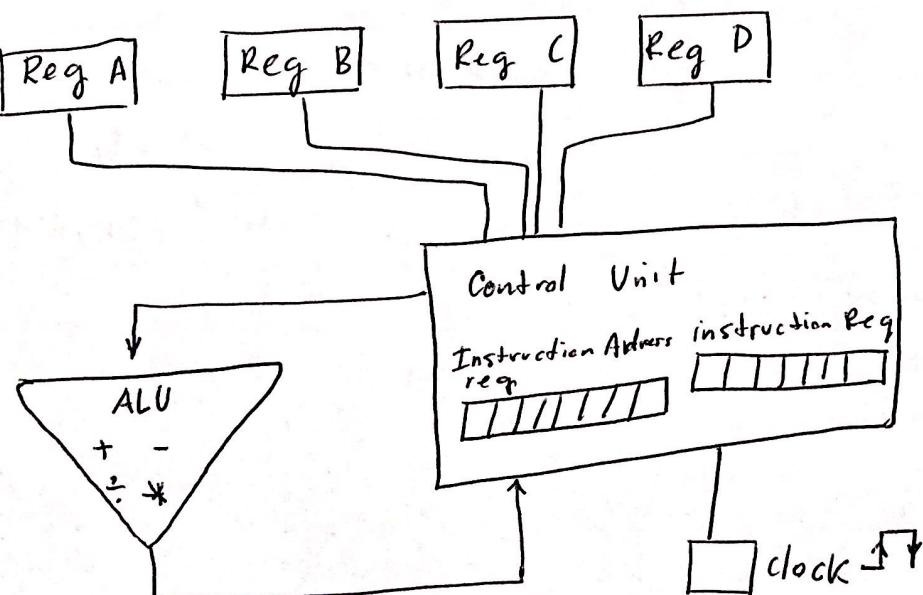
1] 010 1010

① Se le quita la cifra más significativa y se parte a la mitad la capacidad de representación de negativos

• Francis Bacon: inventa una manera

• 1 byte = 8 bits

• investigar ASCII, diferencia entre procesador CISC, RISC.



• overflow flag

• Instrucción Adress Reg es el puntero

• La unidad de control ejecuta

■ ISA: Instruction set architecture

- La data en los operandos puede ser constante, dir de memoria, Reg. de RAM

- Modos de Direccionamiento

- Instruction set:

- Operaciones Aritmética

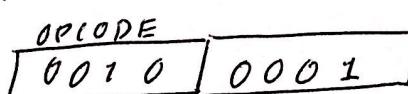
- Instruction Example

Instruction	Description	4-bit opcode	Operand
LOAD A	Read Ram location in to Reg A	0010	4 bit RAM Address
LOAD B	Read Ram location into Reg B		4
STORE A	Write from Reg A into Ram loc.		
ADD			2 bit RegID 2 2-bit reg Id.
HALT			

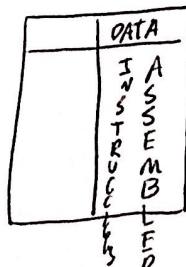
Fetch

- RAM: aquí se guarda el programa

- OPCODE:



- Halt, Infinite, cómo se ejecuta



CISC = Complex instruction Set Computer

RISC = Reduced instruction set computer

- CISC opera con grupos grandes de instrucciones usualmente requieren varios ciclos para completarse, Pentium por ejemplo.
- RISC opera con instrucciones simples requieren pocos ciclos de ejecución, el resultado es más rápido.

CISC	RISC
• más de un conjunto de inst.	• Solo un
• más baratos	• Megahertz & gigahertz
• caros	• más parados
• más transistores	• menos transistores