PED

Tes criterios que marcan los lenguajes de programacion

El paradigma del lenguaje de programación debe ser adecuado al problema y a los programadores

› Cuanto más rico sea el lenguaje en estructuras de datos, más expresividad permite

› Un buen sistema de tipado bien entendido y usado

Paradigmas:

- Imperativo, porcedimental

Un lenguaje de ordneador consiste en una SeRIE DE ACCIONES QUE CAMBIAN LOS ESTADOS DEL ORDENADOR , es decir, le da una secuencia ordenes (b =3, alamcan¡cenas en una celda el 3 en la memoria)

Ventajas: es intuitivo, (le das órdenes)

Desventajas: si no les dan bien la orden no la acata, el ordeandor hace mal las cosas porque le programador no las transmite bien

Problema: solo funciona de manera eficiente para problemas pequeños, si el problema es muy complejo(grande) no es eficiente

- POO

Es el primer paradigma mixto(POO + imprrativo)

- Funcional

Te comunicas con el ordenador mediante funciones(una relación que para los mimos valores de entrada se obtiene los mismos valores de salida), no tienen asignaciones, es muy difícil de hacer “a=1”

SQL seria un ejemplo de lenguaje de porgramacion funcional

No hay compiladores, hay motores

- Concatenativos

Los programas son funciones compuestas una detrás de otra(g o f)

- Staack based

Lenguaje que va como una pila, los números los metes en la pila, con el . sacas la cima, con + sacas con los primeros y los suma

- Concurrente

Mixto(conc + imperativo)

- Scripting(guion)

So lenguajes para que utilizan otros componentes que son autónomos(programas completos) que puedan hacer comportamientos complejos

Python,

Tipos: darle semantica a 0 y 1 del ordenador

Expresión: combinar diferentes tipos (String, int)

Funciones: procedimiento que devuelve un valor

Comando: una palabra reservada,

Una entidad de primera clase se puede crear en tiempo de ejecucion,

· Assigned to a variable or element in a data structure

· Passed as an argument to a function ·Returned as the result of a function

En java, los objetos

C y C++ tiene punteros y java no, principal diferencia

Los lenguajes en timpo real da su mejor resultado en poco tiempo mienttas que el lenguaje normal tarda mas en hacer la operación corresponinete

DATO: Unidades de tratamiento dentro de un sistema informático. Los datos de entrada junto con los datos de salida constituyen lo que se denomina INFORMACIÓN.

El tipo es lo mas importante, ya que con un tipo puedes hacer unas ocsas que con otro,

Id: numeros, letras y barra bajas, y no puede empezar por números ni puedes usar palabras reservadas

Valores: String con " y int con números(14)

Tipos:

SISTEMAS DE CONTROL DE VERSIONES

Gestión del desarrollo de cada elemento de un proyecto(explorador de directorio) a lo largo del tiempo

Mecanismo de almacenaje de cada elemento que deba gestionarse (archivos de código, imágenes, documentación…) · Posibilidad de añadir, modificar, mover, borrar …· Historial de las acciones realizadas con cada elemento pudiendo volver a un estado anterior · Otros: generación de informes de cambios, informes de estado, marcado con nombre identificativo, etc

Tiempo: es la palabra mas importante,

Se trabaja sobre un repositorio(base de datos del control de versiones), donde se almacena la información de todo el desarrollo

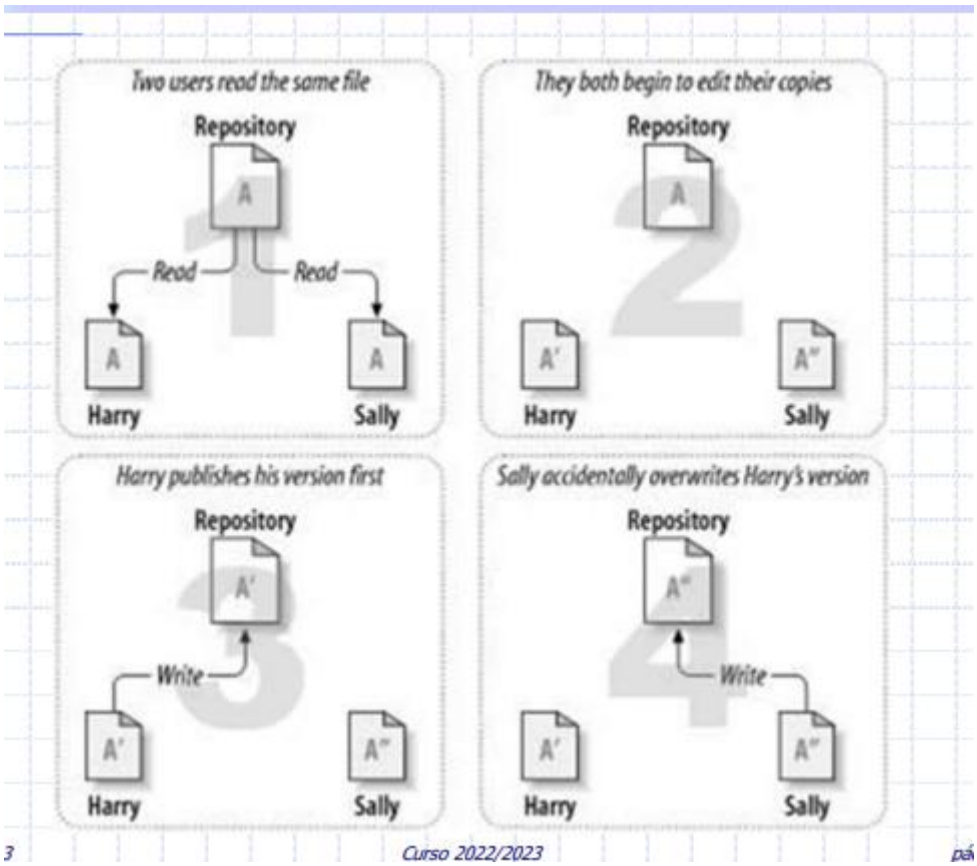
Importantes:

Centralizados: tiene el repo en un solo sitio,

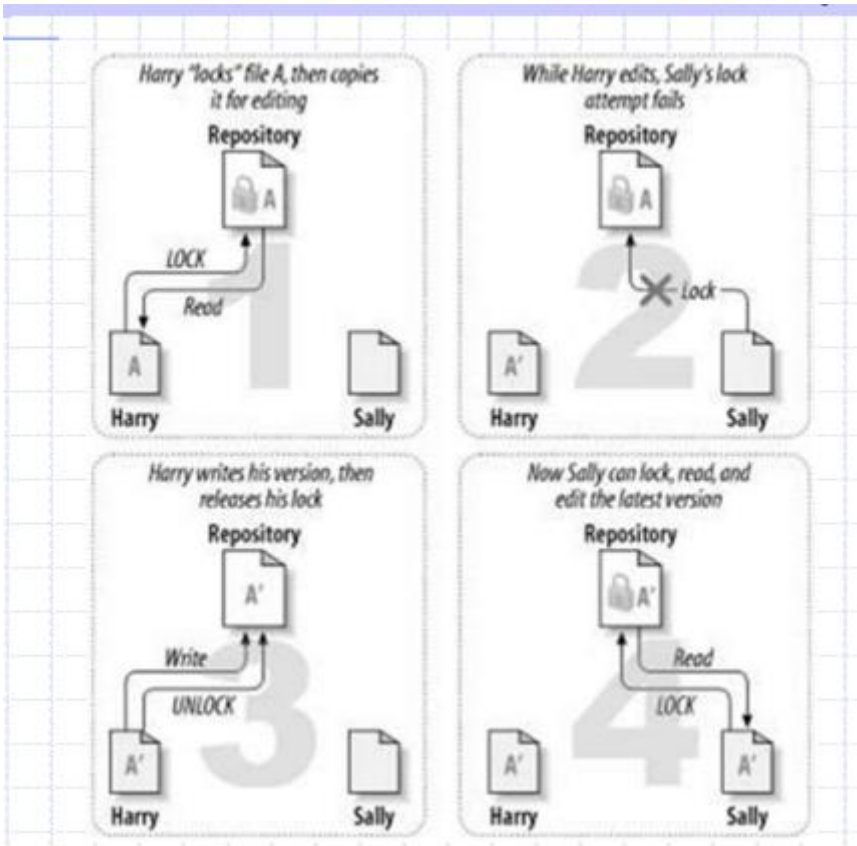
Distribuidos: hay múltiples copias,

Fusionar cambios: cuando vas a subir tu trabajo y alguien de tu equipo ha subido algo justo antes, lo fusionas

Diferencia entre repo y copia local: una copia local solo contiene un punto en el tiempo, el repo los contiene todos

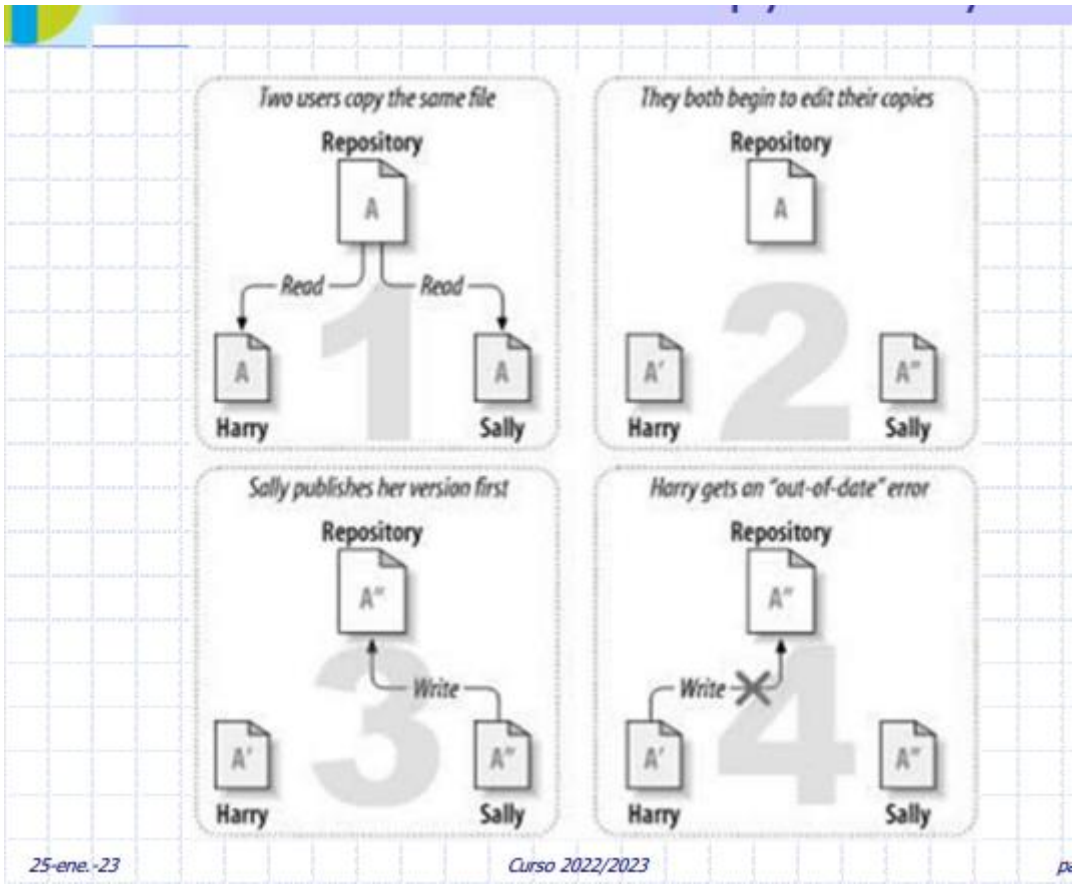


La copia de Harry se pierde, esto se resuelve con semáforos(no es el caso), si es remoto(es el caso), un patron concurrente

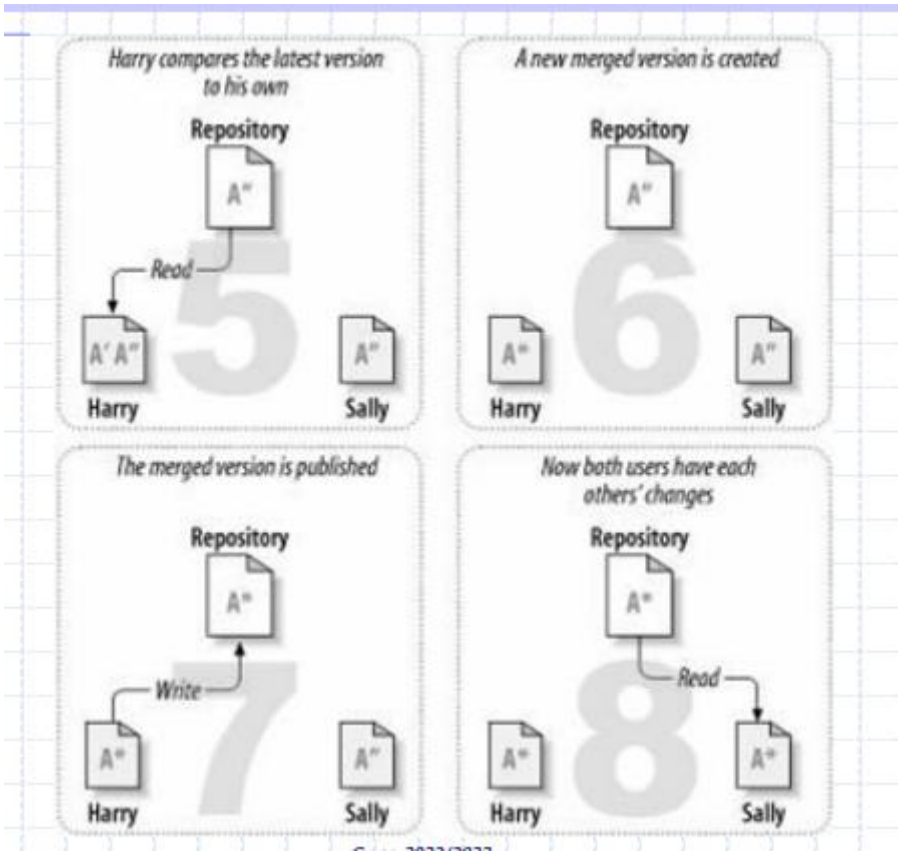


Utilizas un candado(lock),

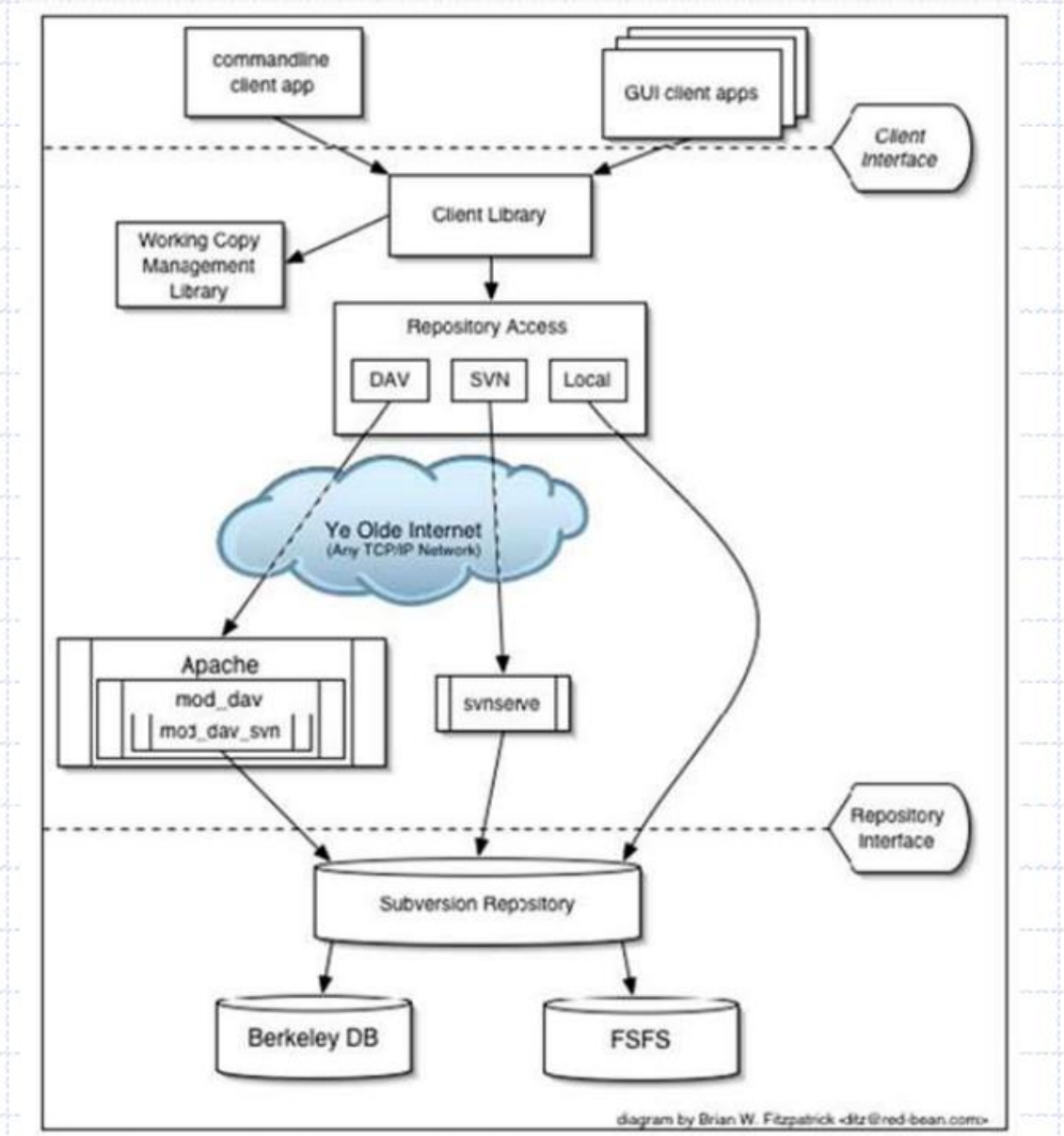
Desv: Solo puede trabajar uno a la vez, si se jode el ordenador a Harry se queda lock siempre,



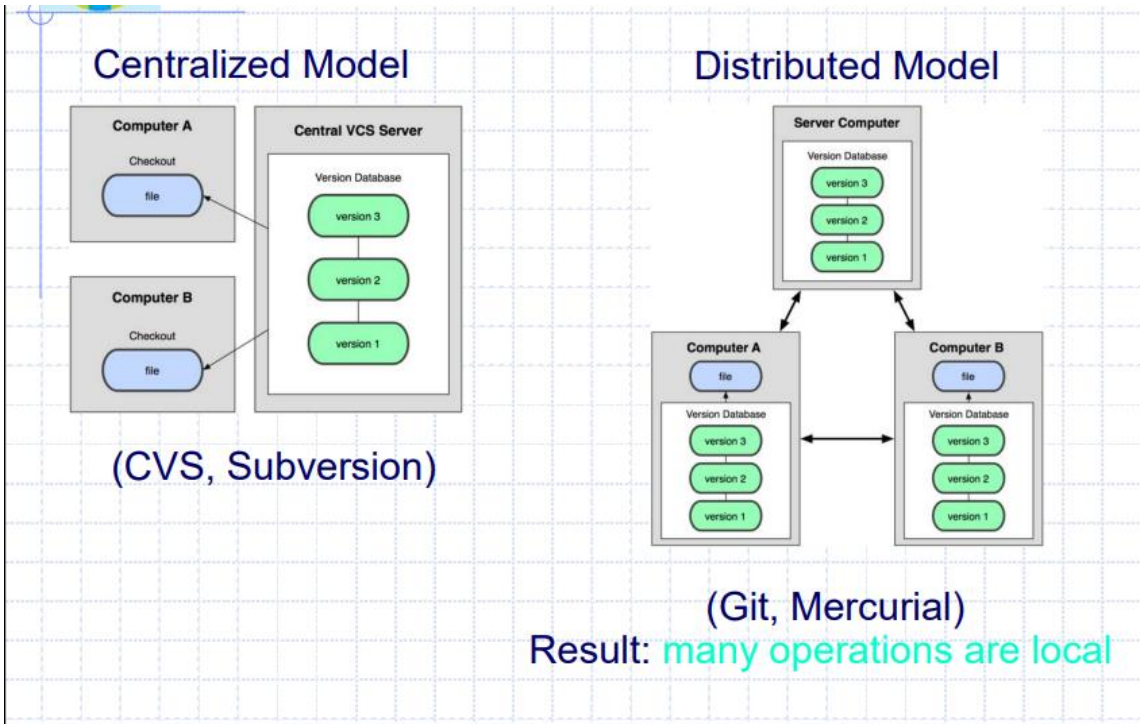
No candado: no puede entregar porque hay una actualización , entonces harry no puede entregar,



Es el modelo que se implanta en todos los sistemas de control de versiones, tienes una copia, y el ultimo(sally) que entrega tiene que leer el ultimo cambio subido

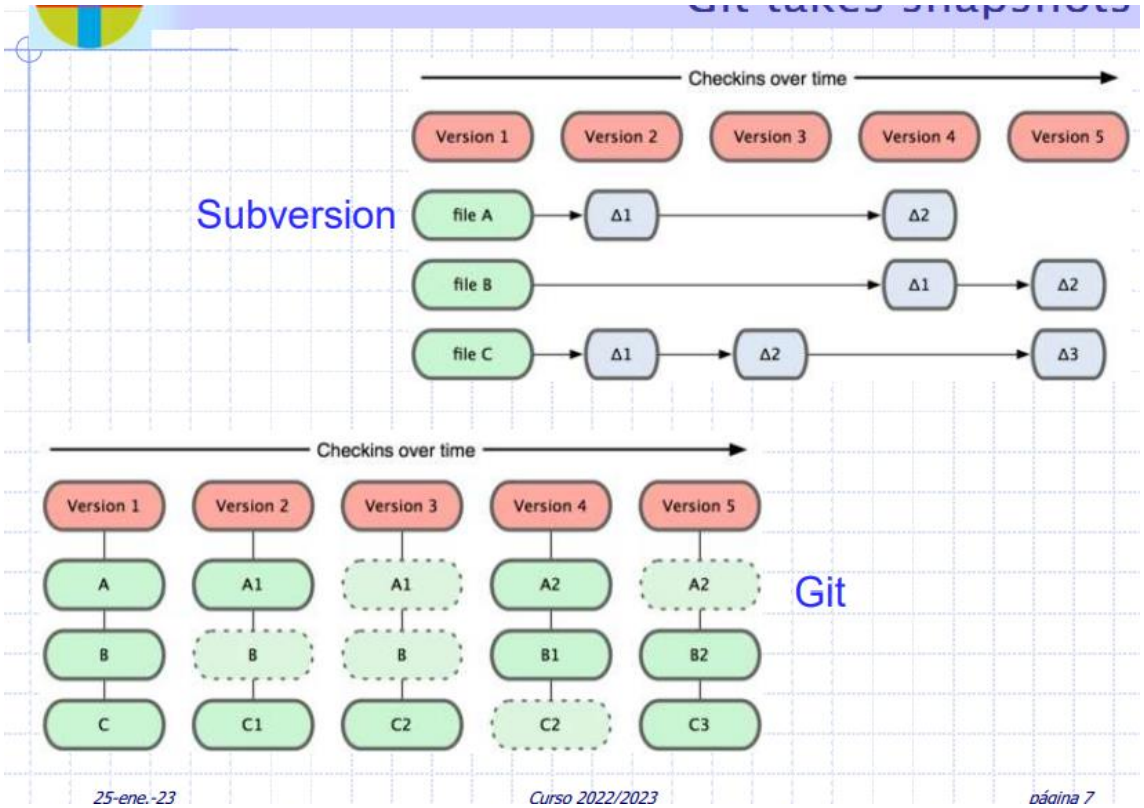


Apache: servidor web,



Diferencia centralizado y distribuido

BD esta en un sitio, en el distribuido neecesitas una copia del repo y una copia del trabajo, es decir, los dstribuidos no tocan la red y el centralizado si



Subversio tarda menos tiempo

Git: ocupa mucho espacio

Dato de carácter:

EBDIC: utilizado en ibm,

ACII: Representa carateres americano(interrogaion , letras, menos la ñ)

00	NUL	10	DLE
01	SOH	11	DC1
02	STX	12	DC2
03	ETX	13	DC3
04	EOT	14	DC4
05	ENQ	15	NAK
06	ACK	16	SYN
07	BEL	17	ETB
08	BS	18	CAN
09	HT	19	EM
0A	LF	1A	SUB
0B	VT	1B	ESC
0C	FF	1C	FS
0D	CR	1D	GS
0E	SO	1E	RS
0F	SI	1F	US

Representar caracteres de la máquina de escribir

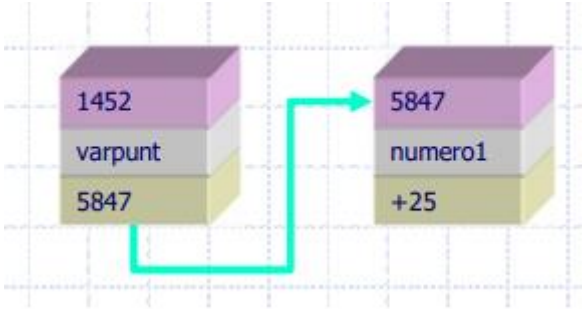
ACII : necesitas una extensión para representar caracteres que no tienes, la ñ, los caracteres con acentos..

Problema: si quieres una aplicación en muchos idiomas, es una movida porque tendrías que hacer una tabla que contenga todos los caracteres para que se pueda comunicar entre ellos.

Solución: UNICODE, puedes representar caracteres números que se dibujan mediante ideogramas o glifos (glyphs), es decir, están todos representados en UNICOD, necesitas 21 bits

UNIODE TRANFROMATION FORMAT: UTF-8, es un sistema de codificación de todos los caracteres de la tierra, los 127 primeros caracteres de Unicode es la tabla ASCII,

Puntero: es un dato que sirve para llegar a otro dato, los punteros tiene tipo, pero las dirección de memoria no, no es idéntico el dato a apuntar,



Como se guarda un objeto, un punro es una herramienta de bajo nivel, por lo que se crean punteros modernos,

A = Alumno(“Pepe”)

B = Alumno(“Paco)

C = a+b

A → Pepe

B -> Paco,

C ->

A y b son referencias que apuntan a alumno, asocias un nombre(a o b) a un objeto(alumno)

***Estudiar diferencia entre puntero y referencia**

Java vs C: java tiene manipulación automática de memoria, es decir, java no utiliza punteros, sino referencias

Tipos de datos estructurados

TABLAS » Estructura estática de datos constituida por un número fijo de elementos del mismo tipo en direcciones de memoria contiguas

Las tablas tienen random Access, e ¡s decir, que pudes acceder arbitrariamenete, están diseñados los arrays para hacer el random acces. VENTAJAS de los arrays, quieres q el random acces vaya a toda ostia, ya que las direcciones d memoria están contiguas

Además, puedes modificar y buscar elementos dentro de ellas, son estructuras estáticas, por lo que no puedes añadir, borrar, no puedes añadir ni partir un array a un array, tienen orden, el inidce 0, el índice 1.. , se puede considerar que son reorndeables,

LISTA: estructura dinámica de datos constituida por elementos (habitualmente, pero no necesariamente, del mismo tipo) denominados nodos. El orden de los elementos es significativo

Las listas también tiene random acces, pero es ma slento que en las tablas, el acceso secuencial es barato,

For i = 1 to len(l)

Print(l[i]) estamos utilizando random Access, de la otra manera, utilziamos manera secuencial

Java no tiene listas, pero si arrays

Pilas, ordenada, heterogénea y dinámicas,

Que operacio no puedes hacer una pila, no tiene random acces, ya que no puedes ir al tercer elemento, solo puedes ir a lacima

Conjunts, no tene random ces ya que no hay orden, tienen acceso secuencial, permite añadir elementos repetidos o no, depende del set qu utilices,

Java no tiene conjuntos, ni arraylists(listas), Python tiene conjuntos(no tienes que importar nada)

Mapa o diccionario: pares de clave valor, claves únicas, valores pueden repetirse, tienes que meter k y v, no solo una, operación. Dame el valor asociado a la clave, no puedes hacr una lista con un diccionario, como no hay orden, da igual si tienes claves de diferentes tipos, el nombre tiene que ser constante(inmutable), además, antes de introducir un elemento en el diccionario, pregunta si hay alguna clave igual a alguna clave que se haya insertado previamente, ,

Árbol: una estructura formada por nodos relacionados cuya propiedad es. Cada nodo tiene un nodo padre, menos el nodo raíz , un nodo puede tener ninguno, uno o muhchos hijos, no puedes acceder a nodo 7 porque no es una estructura de datos lineal(no hay random acces),pero si puedes hacer acceso secuencial, (recorrido en anchura o en profundidad), tiene 0(log 2) cuando el árbol es balanceado

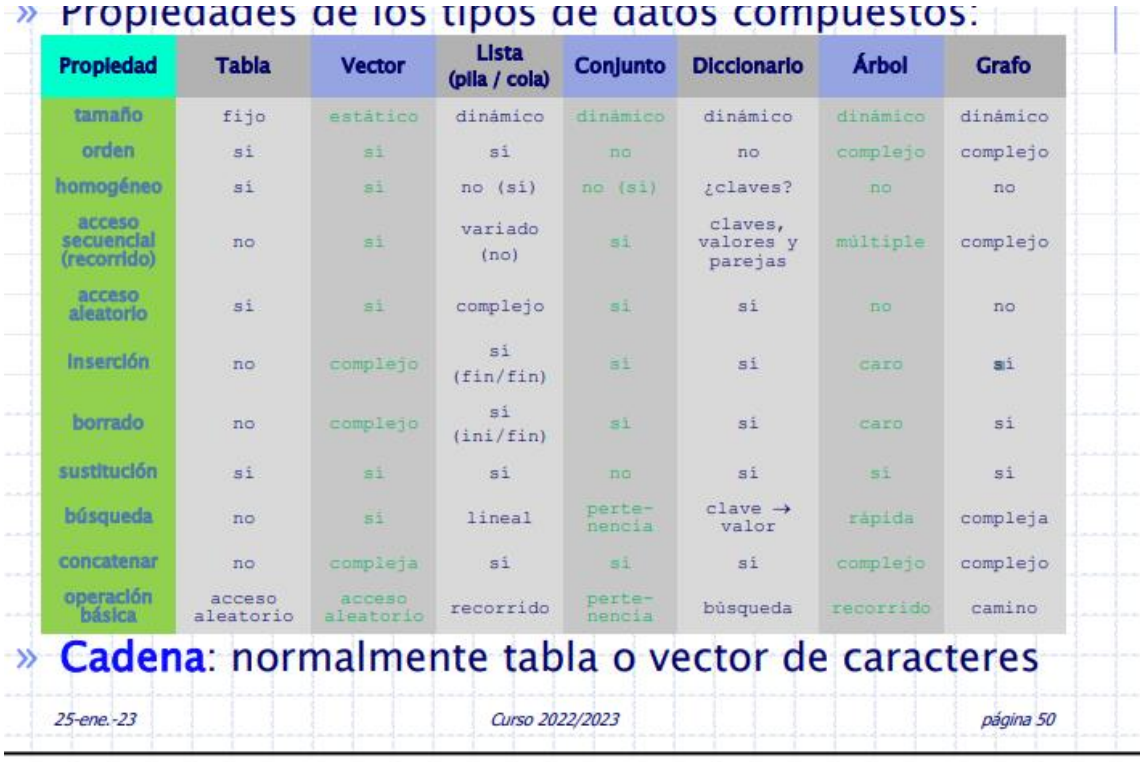
Ej: árbol con 2 hijos, árbol binario ,

Además, los arboles tienen que estar balanceados, es decir, esta balanceado cuando todas las capas menos la ultima están completas

Las listas son un caso particular de los aboles, es decir, las listas son un árbol unario

Grafos, un árbol es un grafo acíclico, pueden tener sentido(flechas=dirigido), y ser ciclicos(que puedes llegar al mismo sitio)

***IMPORTANTE



» Propiedades de los tipos de datos compuestos:

Propiedad	Tabla	Vector	Lista (pila / cola)	Conjunto	Diccionario	Árbol	Grafo
tamaño	fijo	estático	dinámico	dinámico	dinámico	dinámico	dinámico
orden	sí	sí	sí	no	no	complejo	complejo
homogéneo	sí	sí	no (sí)	no (sí)	¿claves?	no	no
acceso secuencial (recorrido)	no	sí	variado (no)	sí	claves, valores y parejas	múltiple	complejo
acceso aleatorio	sí	sí	complejo	sí	sí	no	no
inserción	no	complejo	sí (fin/fin)	sí	sí	caro	sí
borrado	no	complejo	sí (ini/fin)	sí	sí	caro	sí
sustitución	sí	sí	sí	no	sí	sí	sí
búsqueda	no	sí	lineal	perte-nencia	clave → valor	rápida	compleja
concatenar	no	compleja	sí	sí	sí	complejo	complejo
operación básica	acceso aleatorio	acceso aleatorio	recorrido	perte-nencia	búsqueda	recorrido	camino

» **Cadena**: normalmente tabla o vector de caracteres

Sistemas de tipado

Le da significado de colecciones de bits, es decir, un numero, un booleano..

Como son las relaciones entre los diferentes tipos ,hasta que punto pueden interoperar entre los diferentes tipos,

Cada lenguaje de programación tiene un sistema d tipado diferente,

Propiedades básicas de un sistema de tipado de cualquier lenguaje:

Chequeo, precisión , seguridad y equivalencia

Chequeo de tipos: puede ser estático o dinámico, no son exactamente así, tienden a ser estáticos o dinámicos, y pueden tener

Seguridad: seguro o inseguro

Preciiosn: débil o fuerte

Equivalencia: nominativo al estructural, o pato(duck)

Propiedad. **Chequeo estático:** es aquel que se realiza en tiempo de compilación, se asocia un tipo a un nombre y se hace una declaración de datos,

Ej. int n = 3; 3 es un objeto, y n es una referencia(nombre), int el tipo, el int se asocia al objeto 3, es decir, el nombre n lo asocias al tipo int

El chequeo usa la propiedad d equivalencia para ver si los tipos son idénticos, es decir, que sea nominativo(int n = 3), que los dos sean del mimos tipo

Una vez que hemos declarado es de tipo tal, solo se puede asociar(con el operador =) se comprueba que sean del mimo tipo, si es de tipo distinto, da un error de tipado, todos los chequeos se hacen en tiempo de compilación

Ventajas de hacer chequeos en tiempo en compilación: no te da el error en tiempo de ejecución, es decir, es mas eficiente ya que el ejecutable es mas rápido, porque no pones el código de comprobación de tipos,

Inconvenientes: fias toda la seguridad en tiempo de compilación, tienen a no coloca salvaguarda de tipos en el ejecutable,

Chequeo dinámico: el nombre solo esta asociado al objeto, los nombres se enganachan a los objetos, es decir, i = 3; , ¿i tiene tipo? No, solo el 3,

SE COMPRUEBa los tipos en tiempo de ejecución,

Si pones:

A = 3

B = “hola”

Print(a +b)

Salta una excepción de tipado cuando ejecutas el código(en Python)(en otro saldría 3hola, en jaavScript), si no lo eejcutas no hay error, esto quiere decir que la excepción salta en tiempo d ejecución, el operador "+" comprueba los tipos, y no sabe si concatenar o sumar

Evaluación de expresiones:

e = input()

(3+9) + 3*5

eval(e)

Te sale 27

Pero si haces:

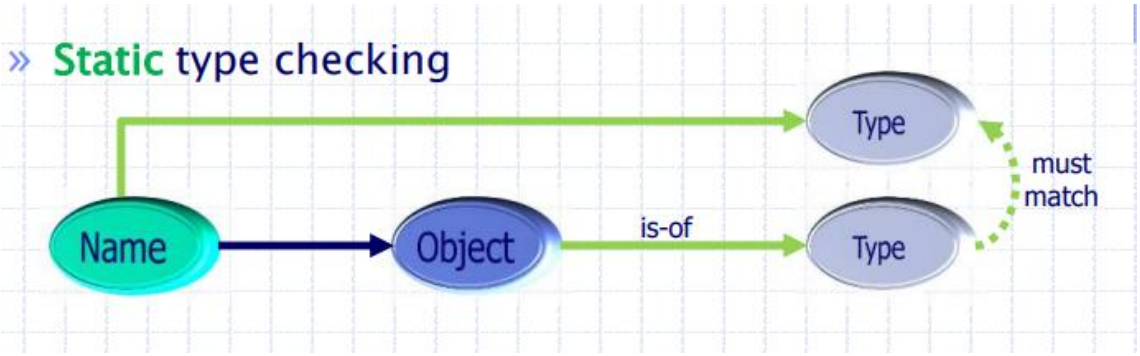
Eval(3 + "hola")

Te da error

REPL: es lo que he escrito arriba, todo esto en Python, en java tienees que compilar todo, no puedes linea por linea

Metaprogramación: que los programas los escriba otros programas

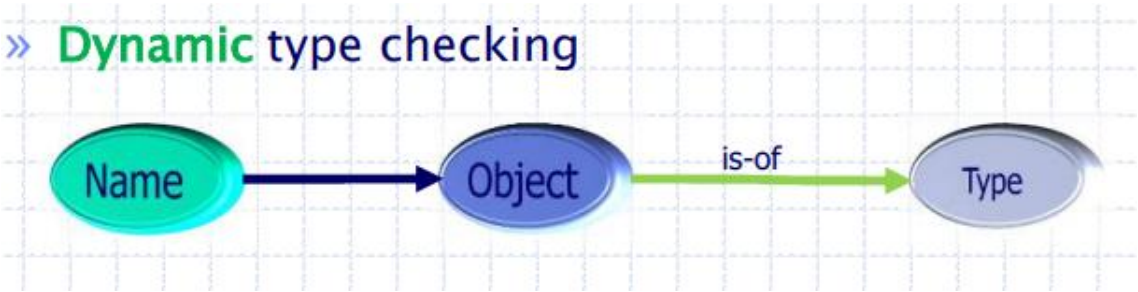
Resumen:



Name -> Type : es String nombre

Name -> object: i = 3

Object -> Type: 3(entero)

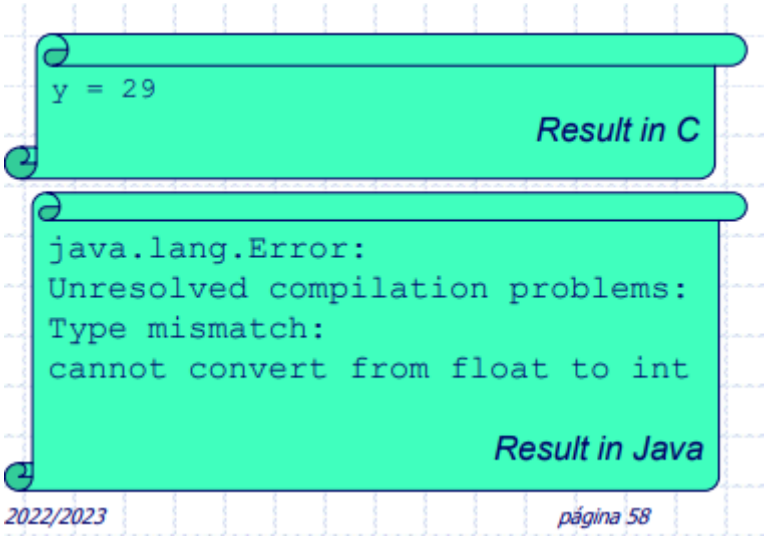


Un nombre se asocia un objeto y un objeto es el que tiene tipos

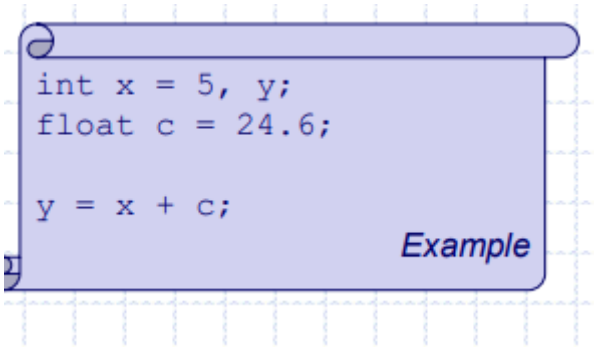
Ejemplos. ESTAICO: C, C++, JAVA,

DINAMICO: Python: JavaSCRIPT, PHP

Precison(lenguaje fuertemento tipado): no permite operaciones con objetos de diferentes tipos



Lenguaje débilmente tipado: el contrario,



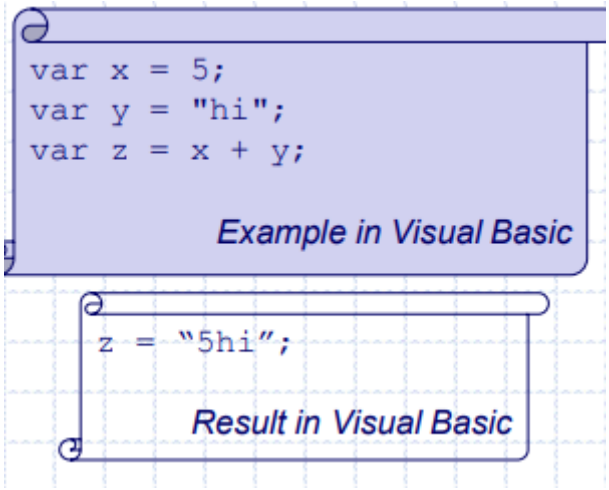
JavaScript tiene un lenguaje debilmente tipado y java no, ya que JavaScript fuerza a poner los mismos tipos

Además, en Python puede sumar el 3 + 3.5, lo hace sin preguntar(coerción= coaccionar, forzar)

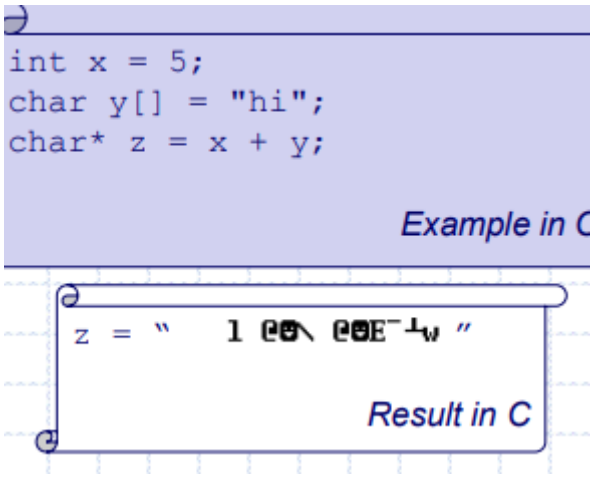
No hay ningún lenguaje mejor o peor, es todo cuestión de estilo del programador

Seguridad: k

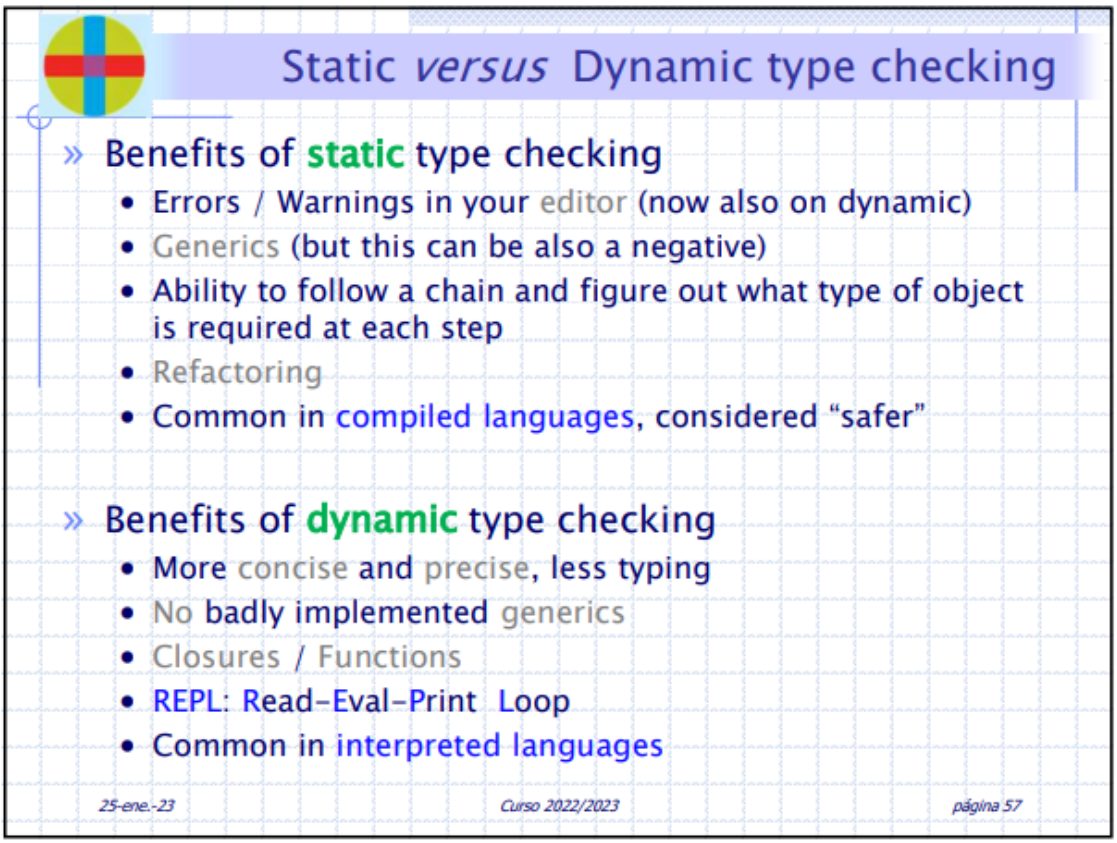
Seguro: el ouptut no es mierda



Inseguro: C es un lenguaje inseguro , por el ouput



Beneficios de tipado estico y dinamkico



Beneficios del chequeo estaico: errores en tu editor,

Generico: son tipos (plantillas), genérico<Integer>, es un tipo paramétrico, hay que parmrtrizarlo con el tipo de los elementos que hay dentro de la lista, (puede ser negativo) si esperas un array de enteros no puedes enviar un array de strings,

Refactorizar(puedes cambiar los tipos), que no todos los lenguajes compilados lo tienen(common)

Tiene cast

Beneficios de chequeo Dinámico: less typing(tiene que escribir menos)

No badly implemnted generics, las listas son heterogéneas,

REPL: abrir el interpete, no metrte dentro de una clase, programar desde la línea de comandos

Clausuras: cerrar la variables libres,

```
Def crear_multitpiclador(n)
    Def multiplicador_interiro(x)
    Return x*n
Return crear_multiplicador
```

P3 = crear_multplilador(3)//esto es al clausura

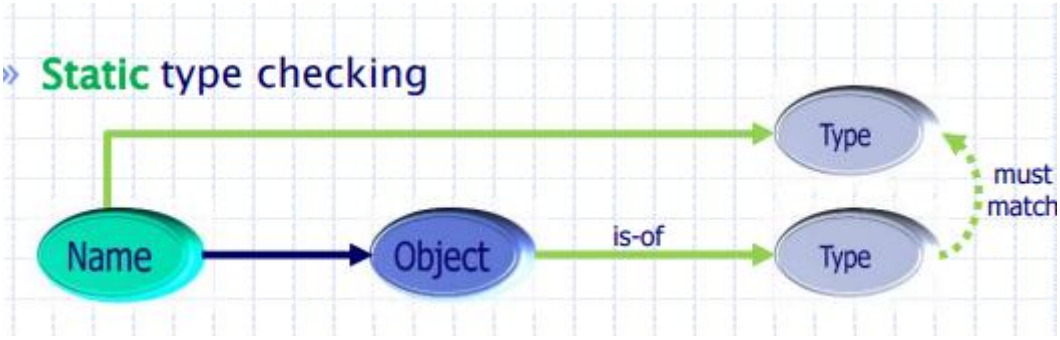
Por ejmplo, javascript

Difrencia type cast(coge el dato lo reinterpretas con otro tipo(es peligroso, pero nunca falla)) y un tipe conversión(dame le valor del tipo destino que corrsepone al valor del tipo origen, cuya representacion interna no coincide),

Una conversión viola o no viola el sistema de tipos, depende del lenguaje, 57.4 a 57

Un conversor es una función que recibe un tipo y devuelve ese tipo

Tipo de Equivalencia:



Cuando son equivalentes pasan el chequeo,

Nominativa: dos tipos son equiv si se llaman igual, también si las familias de los tipos es la misma(cuando haces herencia)

Estructural(patrones de bits), lo mismo pero con estructura

1100011= 111333 , no tiene la misma, tamaño de bits y bits iguales en la misma posición

Duck: no se comprueba el tipo, se compureba las capacidades de los tipos, es decir, si puede ser autónomo,

Coercion: ¾ =0.75, dicides 2 enteros y te un decimal, la división es real, aplica una conversión,

POO:

=(operación de asignación) , x =3, se copia, el valor,

Clase: es una descripción de ejemplares, donde creas ejemplares, comportamiento(métodos o servicios(que hace algo por ti) o mensajes)

Objeto: un objeto es un ejemplar de una clase y todos ellos comparten una serie de propiedades y comportamientos comunes(clases), es la representación de una idea

Ocultación, que un objeto no conozca nada de otro objeto, esto se consigue (en Python)

Encapsulacion:

Sobrecarga: permite definir comportamiento que responden a operadores predefinidos por el lenguaje

Templates:

Test de identidad; que de nombres difeentes hagan referencias al mimsoobjeto

Test igualdad: tu eres el que tienes que dar el criterio de igualdad,

Ej: def __eq__(self,otro):

Return true;

B1 = b()

B2=b8)

B3=b1

B1 is b2

>>false

B1 is b3

>>true

B1 == b2

>>true

B1 == b2

True

Aquí se ha hecho sobreacgr8overloading), consiste ena ñaidrle funonalidad a un operador predefinido

También esta diciendo que es igual y que no

La POO esta basada en recursividad(es algo igual a si mismo cada vez mas pequeño), un objeto sabe como se llama otro objeto, invocando un método para ejecutar una acción(otro método)

Cada obj tiene su propia memoria, es decir, aunq comparta caractrsitcas similares, son distintos xq pertenecen distintos objetos,

***Diferencia fundanetal entre clase y objeto: ***

Clase: donde se definen los objetos,

Los objetos están vivos, tiene un ciclo de vida, nacen, crecen y se reproducción, e destruyen, las clases no, las clases son una plantilla para definir o crea objetos

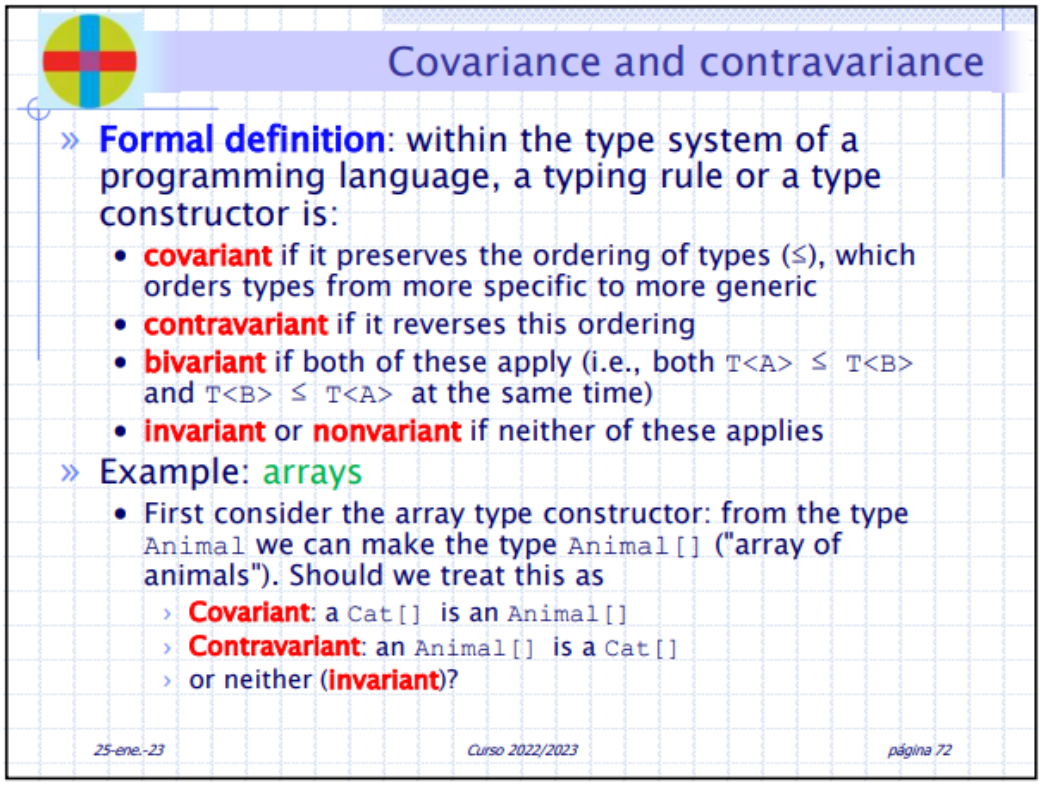
Propiedad de un lenguaje Orientado a objeto: que tiene enlace dinámico(invacacion), Dynamic binding(lo que hace orientado a objeto), polimorfiscomo ,herencia sobrecarga, plantilla,..8(odo esto basado en objeto)

Dynamic binding(enlace dinámico): diferencia ente metodo y funcion y como funcionan, crear un método y añadirselo a una función

SRP: los objetos tienen que ser sencillos, tu motivo de cmabi

OCP:

LSP:



72

Construit tipos(ej, dd tipo alumno): Alumno a = new Alumno()

Tipos compuesto class Gato extdens animal

Tipo covariante: si el orden de estos es el mismo ,

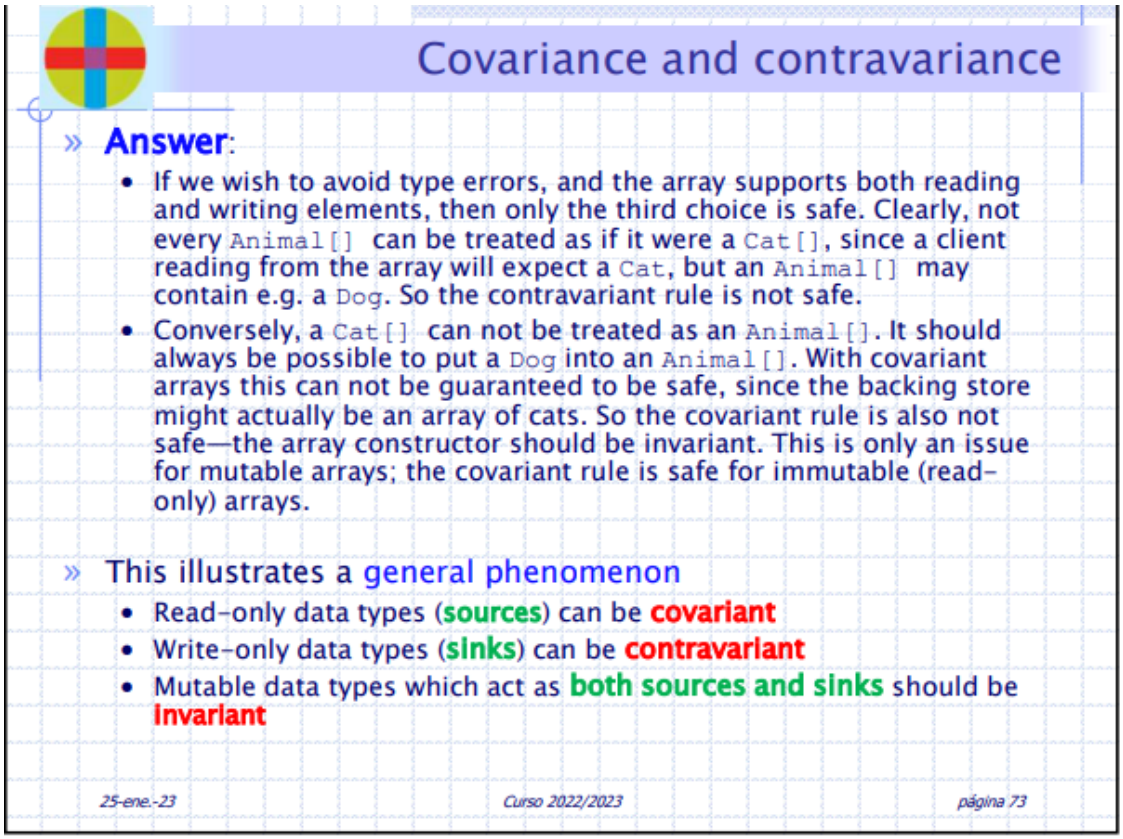
Tipo contravariante: si orden de estos es inverso En un array de animales puedes meter gatos, es decir, Animales[] <- Gato[]

Si es bivariante se cumplen los dos

Si es invriante, no se cuple ninguno de los dos

El array es un constructor d tipos

mirar preguntas



Hrencia multiple: mejor evitarla a no ser que seapas loq haves

PROGRAMACIÓN Y REDES