UNIVERSIDAD DE LOS LLANOS

04 Clases y Objetos

Néstor Suat-Rojas. Ing. Msc (c)
nestor.suat@unillanos.edu.co

Escuela de Ingeniería
Facultad de Ciencias Básicas e Ingeniería

Introducción

¿Por qué aprender Programación Orientada a Objetos?

- Programar más rápido.
- Ser un programador Senior.
- Dejar de copiar y pegar código.

UNIVERSIDAD DE LOS LLANOS

Introducción

Pasos en la Ingeniería de Software

- Análisis (Requerimientos).
    a. Observación
    b. Entendimiento
    c. Lectura
- Diseño
    a. Diagramas
- Desarrollo
    a. Lenguajes de programación

UNIVERSIDAD DE LOS LLANOS

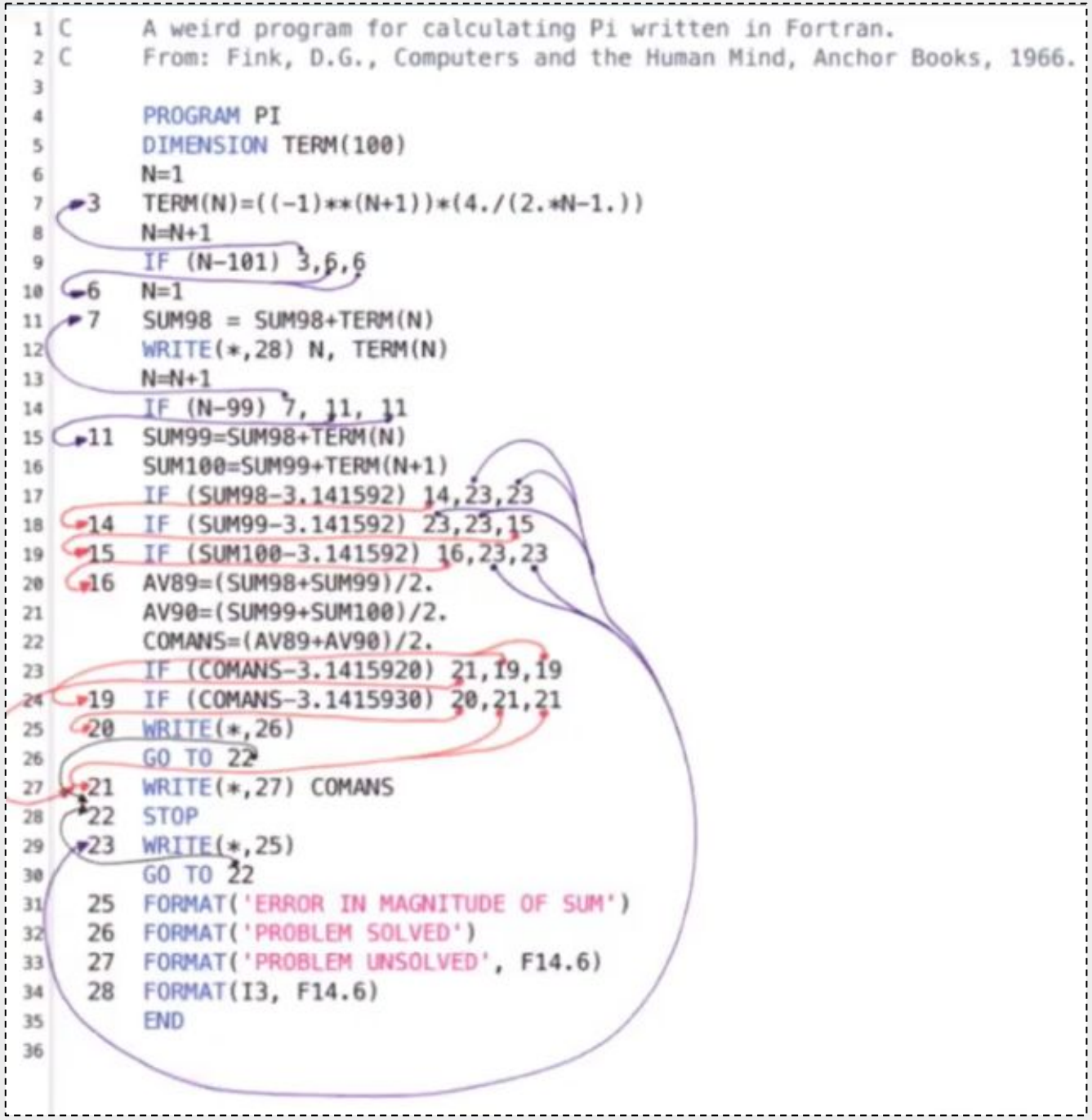
Introducción

Programación Estructurada vs. Objetos

- Código muy largo
- Si algo falla, todo se rompe
- Difícil de mantener
- Código Espagueti

  → if () {} else {}

UNILLANOS UNIVERSIDAD DE LOS LLANOS®



```
C     A weird program for calculating Pi written in Fortran.
C     From: Fink, D.G., Computers and the Human Mind, Anchor Books, 1966.

      PROGRAM PI
      DIMENSION TERM(100)
      N=1
    3 TERM(N)=((-1)**(N+1))*(4./(2.*N-1.))
      N=N+1
      IF (N-101) 3,6,6
    6 N=1
    7 SUM98 = SUM98+TERM(N)
      WRITE(*,28) N, TERM(N)
      N=N+1
      IF (N-99) 7, 11, 11
   11 SUM99=SUM98+TERM(N)
      SUM100=SUM99+TERM(N+1)
      IF (SUM98-3.141592) 14,23,23
   14 IF (SUM99-3.141592) 23,23,15
   15 IF (SUM100-3.141592) 16,23,23
   16 AV89=(SUM98+SUM99)/2.
      AV90=(SUM99+SUM100)/2.
      COMANS=(AV89+AV90)/2.
      IF (COMANS-3.1415920) 21,19,19
   19 IF (COMANS-3.1415930) 20,21,21
   20 WRITE(*,26)
      GO TO 22
   21 WRITE(*,27) COMANS
   22 STOP
   23 WRITE(*,25)
      GO TO 22
   25 FORMAT('ERROR IN MAGNITUDE OF SUM')
   26 FORMAT('PROBLEM SOLVED')
   27 FORMAT('PROBLEM UNSOLVED', F14.6)
   28 FORMAT(I3, F14.6)
      END
```

UNIVERSIDAD DE LOS LLANOS®

```
function iIds(startAt, showSessionRoot, iNewNnVal, endActionsVal, iStringVal, seqProp, htmlEncodeRegEx) {
    if (SbUtil.dateDisplayType === 'relative') {
        iRange();
    } else {
        iSelActionType();
    }
    iStringVal = notifyWindowTab;
    startAt = addSessionConfigs.sbRange();
    showSessionRoot = addSessionConfigs.eUHiddenVal();
    var headerDataPrevious = function(tabArray, iNn) {
        iPredicateVal.SBDB.deferCurrentSessionNotifyVal(function(evalOutMatchedTabUrlsVal) {
            if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                iPredicateVal.SBDB.normalizeTabList(function(appMsg) {
                    if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                        iPredicateVal.SBDB.detailTxt(function(evalOrientationVal) {
                            if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                iPredicateVal.SBDB.neutralizeWindowFocus(function(iTokenAddedCallback) {
                                    if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                        iPredicateVal.SBDB.evalSessionConfig2(function(sessionNn) {
                                            if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                                iPredicateVal.SBDB.iWindow2TabIdx(function(iURLsStringVal) {
                                                    if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                                        iPredicateVal.SBDB.idx7Val(undefined, iStringVal, function(getWindowIndex) {
                                                            if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                                                addTabList(getWindowIndex.rows, iStringVal, showSessionRoot && showSessionRoot.length > 0 ? show
                                                                    if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                                                        evalSAllowLogging(tabArray, iStringVal, showSessionRoot && showSessionRoot.length > 0 ?
                                                                            if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                                                                BrowserAPI.getAllWindowsAndTabs(function(iSession1Val) {
                                                                                    if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                                                                        SbUtil.currentSessionSrc(iSession1Val, undefined, function(initCurrentSe
                                                                                            if (!htmlEncodeRegEx || htmlEncodeRegEx == iContextTo) {
                                                                                                addSessionConfigs.render(matchText(iSession1Val, iStringVal, eva
                                                                                                    id: -13,
                                                                                                    unfilteredWindowCount: initCurrentSessionCache,
                                                                                                    filteredWindowCount: iCtrl,
                                                                                                    unfilteredTabCount: parseTabConfig,
                                                                                                    filteredTabCount: evalRegisterValue5Val
                                                                                                }) : [], cacheSessionWindow, evalRateActionQualifier, undefined,
                                                                                                    if (seqProp) {
                                                                                                        seqProp();
                                                                                                    }
                                                                                                });
                                                                                            }
                                                                                        });
                                                                                    }
                                                                                });
                                                                            }
                                                                        });
                                                                    }
                                                                });
                                                            }
                                                        });
                                                    }
                                                });
                                            }
                                        });
                                    }
                                });
                            }
                        });
                    }
                });
            }
        }), showSessionRoot && showSessionRoot.length > 0 ? showSessionRoot : startAt ? [startAt] : []);
    }
});
```

Programación Orientada a Objetos

Es un paradigma de programación (una técnica o forma de programar), una forma que nosotros podemos resolver un problema usando Clases y Objetos.

Cuatro elementos:	Cuatro pilares:
- Clases - Propiedades - Métodos - Objetos	- Encapsulamiento - Abstracción - Herencia - Polimorfismo

UNIVERSIDAD DE LOS LLANOS

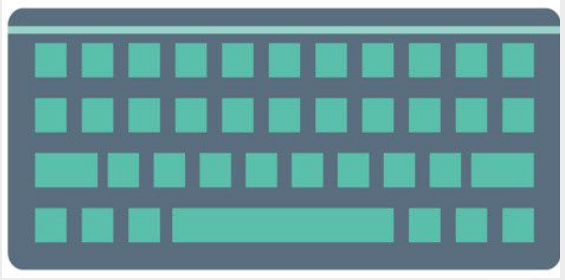
Orientación a Objetos

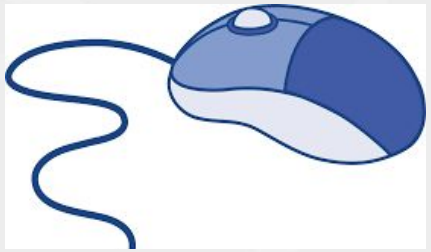
Surge a partir de los problemas que tenemos y necesitamos plasmar en código. Observar los problemas en forma de objetos.

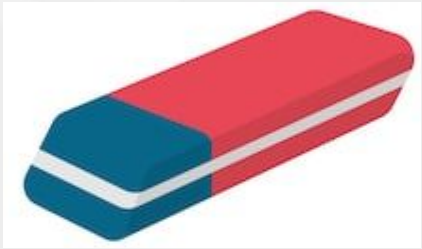
UNIVERSIDAD DE LOS LLANOS

Entidades

En el mundo tenemos diferentes entidades con las que interactuamos, por ejemplo el teclado, mouse, lapiz y borrador.





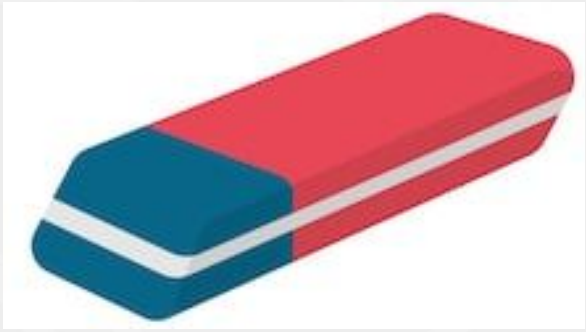


UNIVERSIDAD DE LOS LLANOS

Entidades

Las entidades tiene rasgos que los diferencia a otros.

```
tamaño: 30
color: azul
material: madera
escribir
```

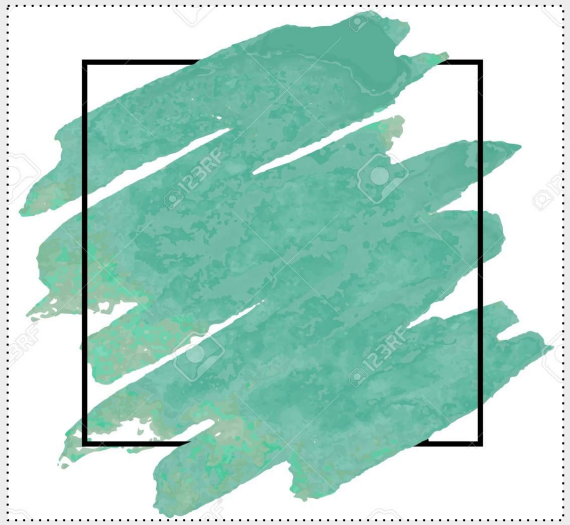


```
tamaño: 10
color: azul-rojo
material: goma
borrar
```

UNIVERSIDAD DE LOS LLANOS

Objeto

Es una entidad que tiene características que lo hacen diferente a otros, también conocidos como propiedades y puede tener acciones diferentes (métodos)

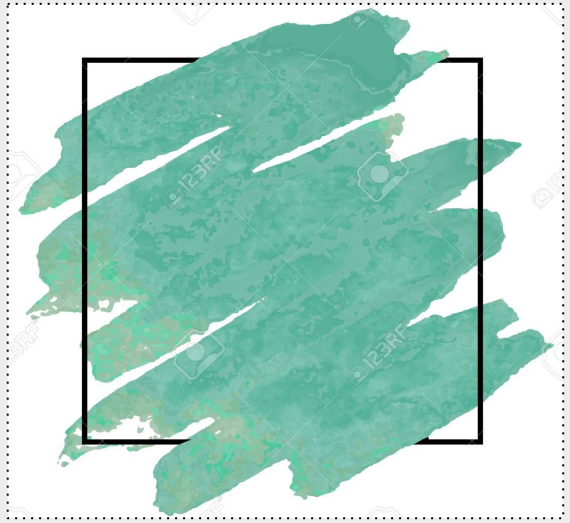


Característica 1: Única
Característica 2: Valor
Acción
Otra acción

UNIVERSIDAD DE LOS LLANOS

Objeto

Es una entidad que tiene características que lo hacen diferente a otros, también conocidos como propiedades y puede tener acciones diferentes (métodos)



~~Característica 1: Única~~ **Variables**

~~Característica 2: Valor~~ **Atributos**

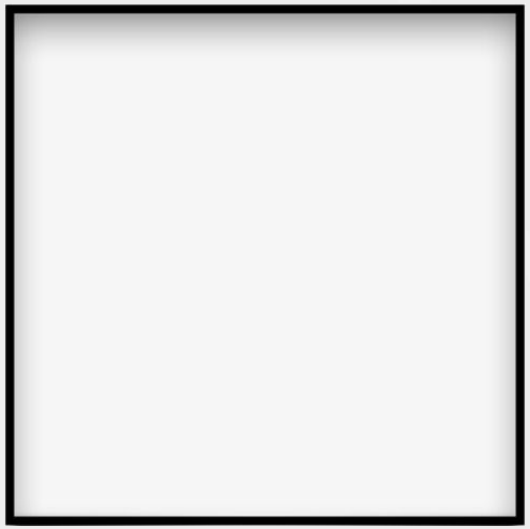
~~Acción~~

~~Otra acción~~

UNIVERSIDAD DE LOS LLANOS

Clase

Una clase es el modelo por el cual nuestros objetos se van a construir y nos van a permitir generar más objetos.



UNIVERSIDAD DE LOS LLANOS

Clase

Si tomamos dos lápices: uno convencional y uno de dibujo:

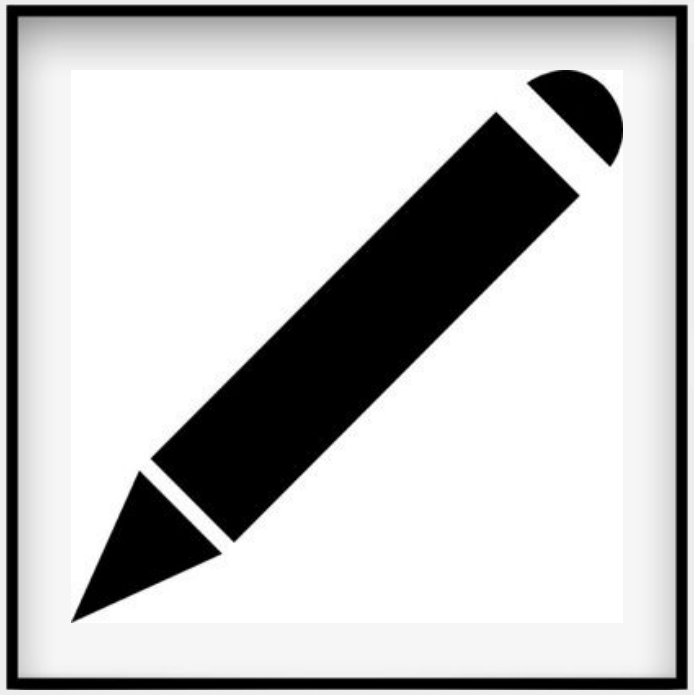
Ambos hacen parte del mismo tipo de objeto: Lápiz, pero, tienen características diferentes (Color, peso, tipo madera)

UNIVERSIDAD DE LOS LLANOS

Clase

Si quitamos todas esas diferencias obtendremos un lápiz genérico.

MOLDE, que se puede utilizar para crear otros tipos de lápices.

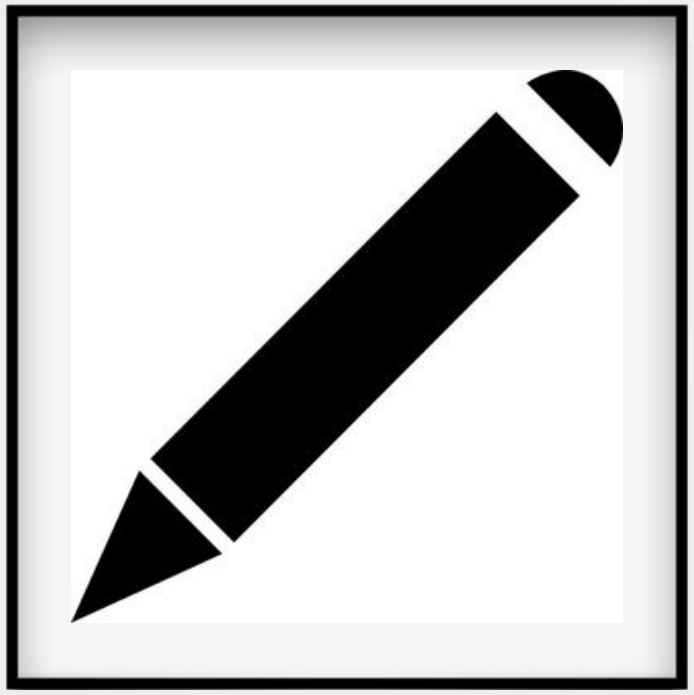


```
Clase Lápiz:
    Tamaño:
    Color:
    Punta:
    Material:
```

UNIVERSIDAD DE LOS LLANOS

Clase

Crear **instancias** es crear un objeto usando un **Molde** (una clase). Una **clase** es un molde para crear objetos.



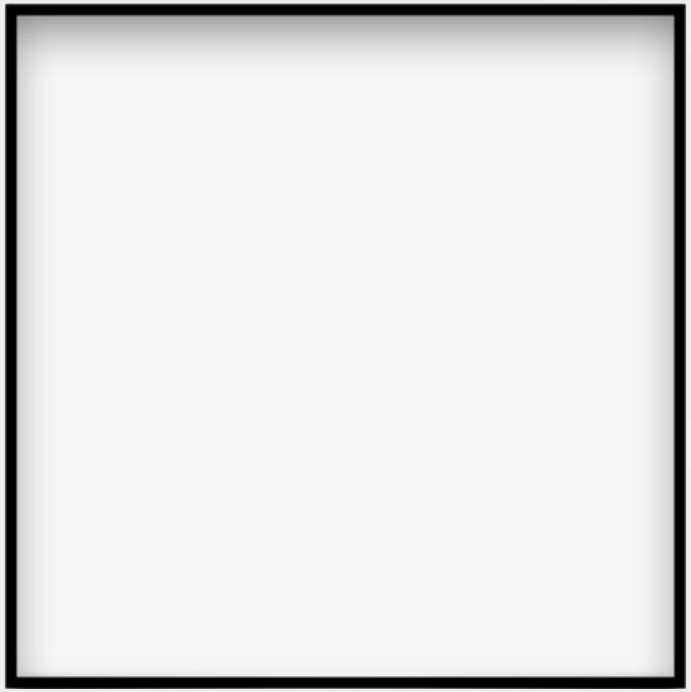
```
Lapiz1 l1 = Lapiz()
```

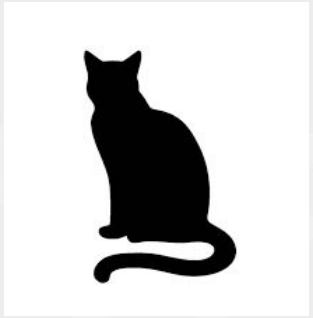
```
Lapiz2 l2 = Lapiz()
```

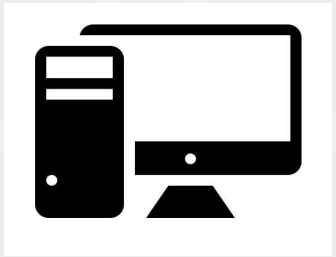
UNIVERSIDAD DE LOS LLANOS®

Abstracción

Identificar objetos y plasmar en programación







UNIVERSIDAD DE LOS LLANOS

Ejercicio Foro

Ejercicio

- Identificar un objeto de la vida real y representar cómo sería su Clase respectiva en C++, es decir como se va llamar y qué características van a tener, de manera que esa clase me permita crear futuros objetos a partir de ella, finalmente escribir 3 ejemplos de objetos creados con su clase.

UNIVERSIDAD DE LOS LLANOS

Identificar Objetos

Cuando tengamos un problema lo primero que debemos hacer es **Identificar Objetos**.

Los objetos son aquellos que tienen **propiedades** y **comportamientos**, también serán sustantivos.

- Pueden ser Físicos o Conceptuales

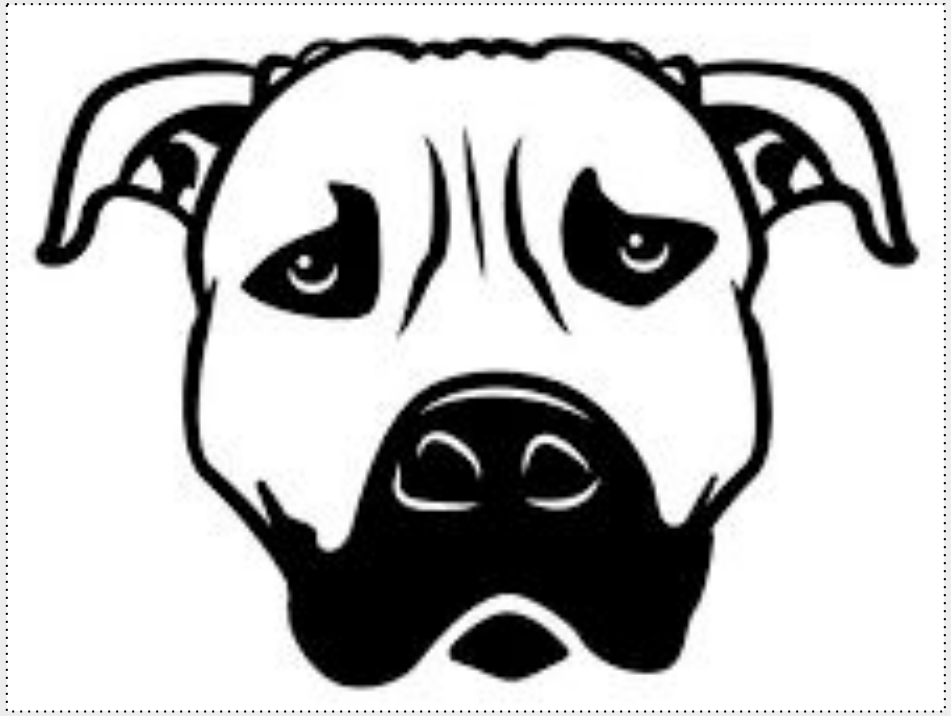
UNIVERSIDAD DE LOS LLANOS

Identificar Objetos

- Las **Propiedades** también pueden llamarse atributos y estos también serán sustantivos.
  - Nombre, tamaño, forma, estado, etc. Son todas las características del objeto.
- Los **Comportamientos** serán todas las operaciones que el objeto puede hacer, suelen ser verbos o sustantivos y verbo.
  - Login, logout

UNIVERSIDAD DE LOS LLANOS

Identificar Objetos



Atributos

+ nombre
+ color
+ raza
+ altura

Comportamientos

+ ladrar
+ comer
+ dormir
+ correr

UNIVERSIDAD DE LOS LLANOS

Modularidad

- Diseño modular
  - Subdividir un sistema en partes más pequeñas (Módulos), estos pueden funcionar de manera independiente.

UNIVERSIDAD DE LOS LLANOS®

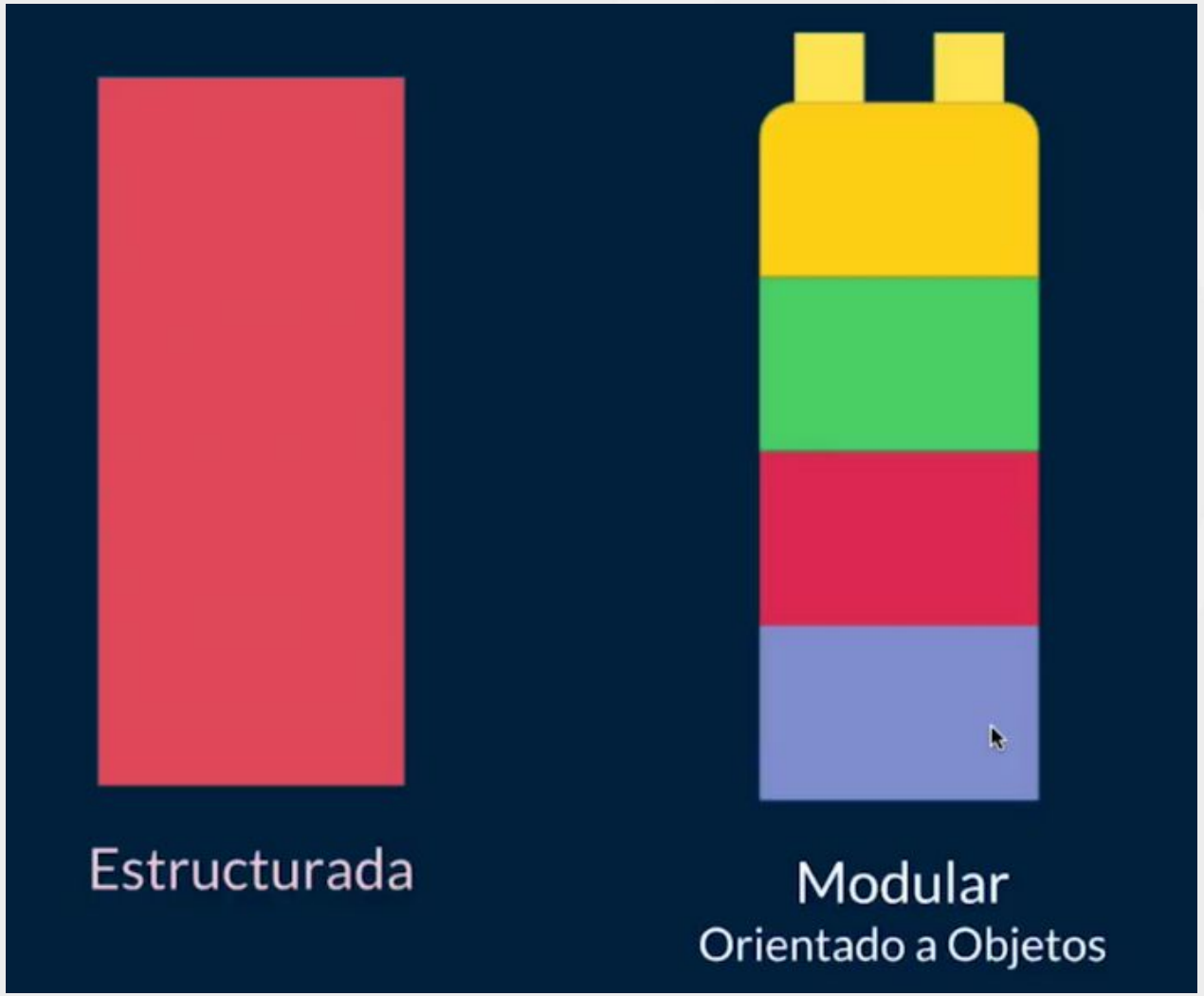
Modularidad

UNIVERSIDAD DE LOS LLANOS®

Modularidad

UNIVERSIDAD DE LOS LLANOS

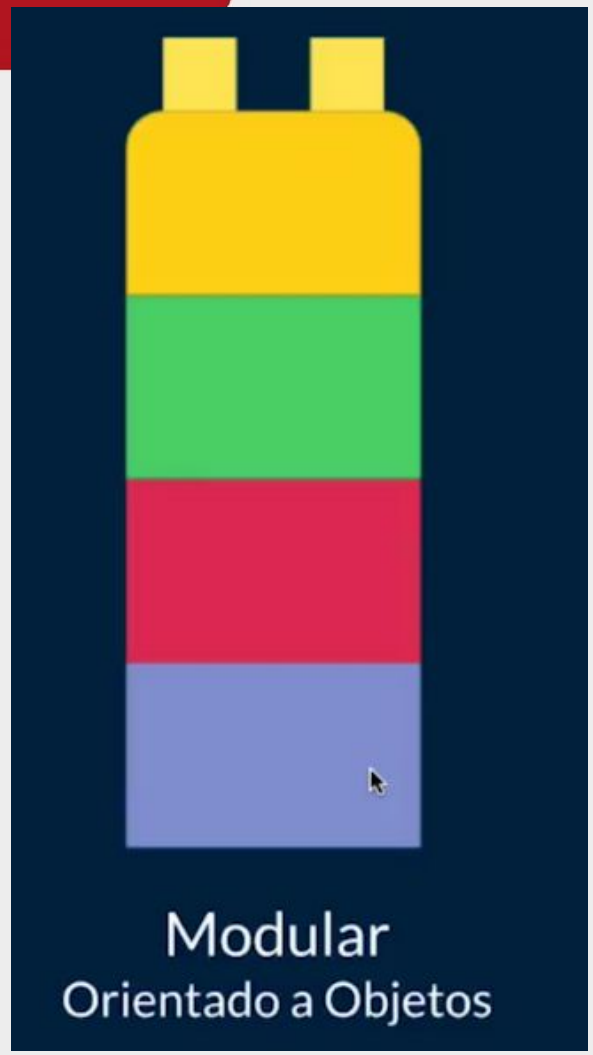
Modularidad



"Divide y vencerás".

UNIVERSIDAD DE LOS LLANOS

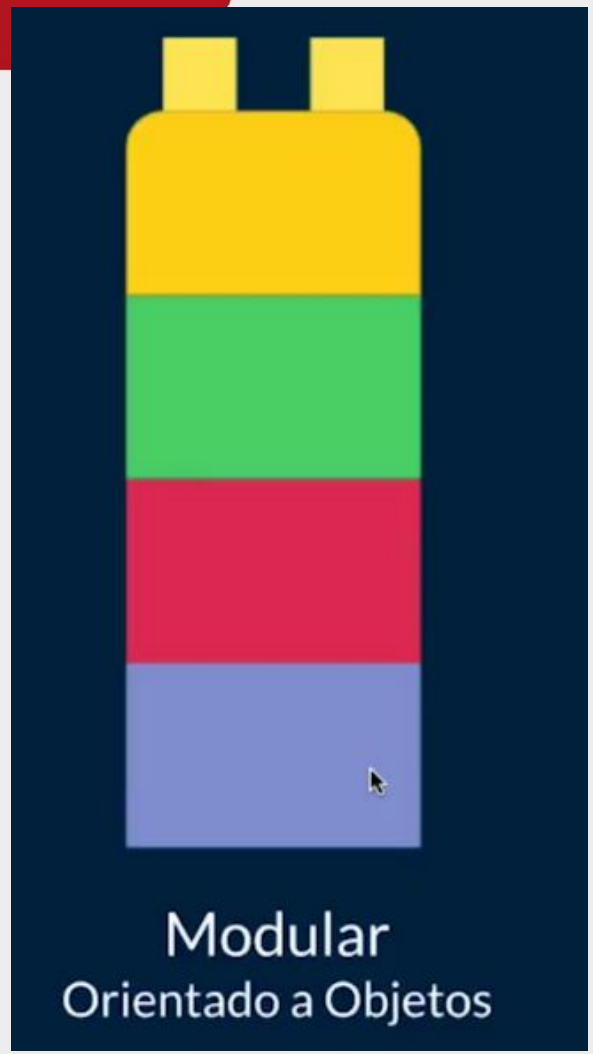
Modularidad



- Reutilizar
- Evitar colapsos
- Mantenible
- Legibilidad
- Resolución rápida de problemas.

UNIVERSIDAD DE LOS LLANOS

Modularidad



Clase

- Modularidad
- Divide el programa en diferentes partes o módulos (clases)
- Separa las clases en archivos

UNIVERSIDAD DE LOS LLANOS

POO

- Clases

```
class Computer{
...
};
```

- Métodos

```
class Computer{
public:
    int suma(int a, int b){
        return a + b;
    }
    int resta(int a, int b){
        return a - b;
    }
};
```

- Objetos

```
Computer p1 = Computer();
```

UNIVERSIDAD DE LOS LLANOS

Bibliografía

- Platzi
- CodigoFacilito

UNIVERSIDAD DE LOS LLANOS

Gracias...

UNIVERSIDAD DE LOS LLANOS®