Object-Oriented Programming (OOP)
vs
Procedural Programming (PP)

CS356 Object-Oriented Design and Programming
http://cs356.yusun.io
September 29, 2014

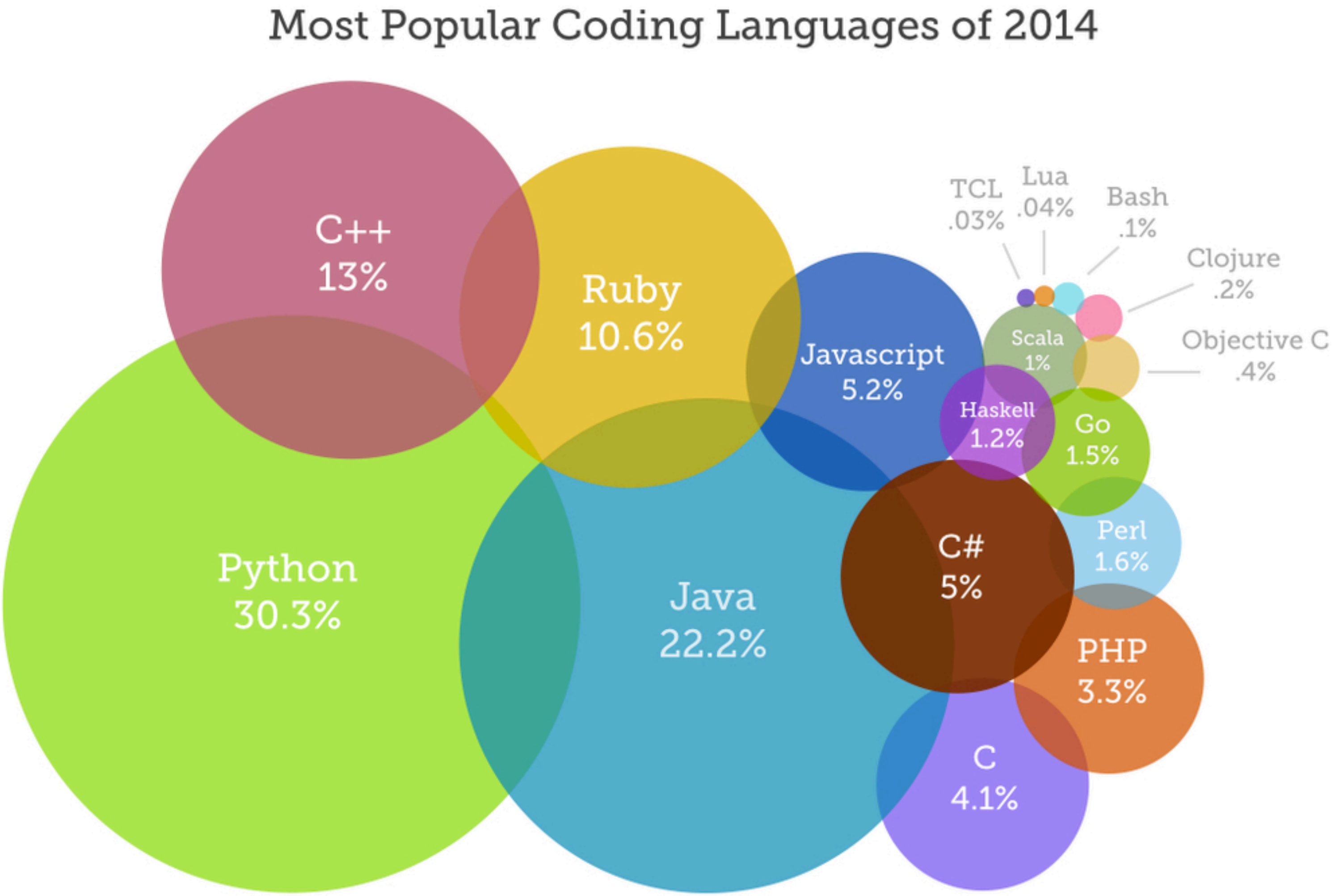
Yu Sun, Ph.D.
http://yusun.io
yusun@csupomona.edu

CAL POLY POMONA

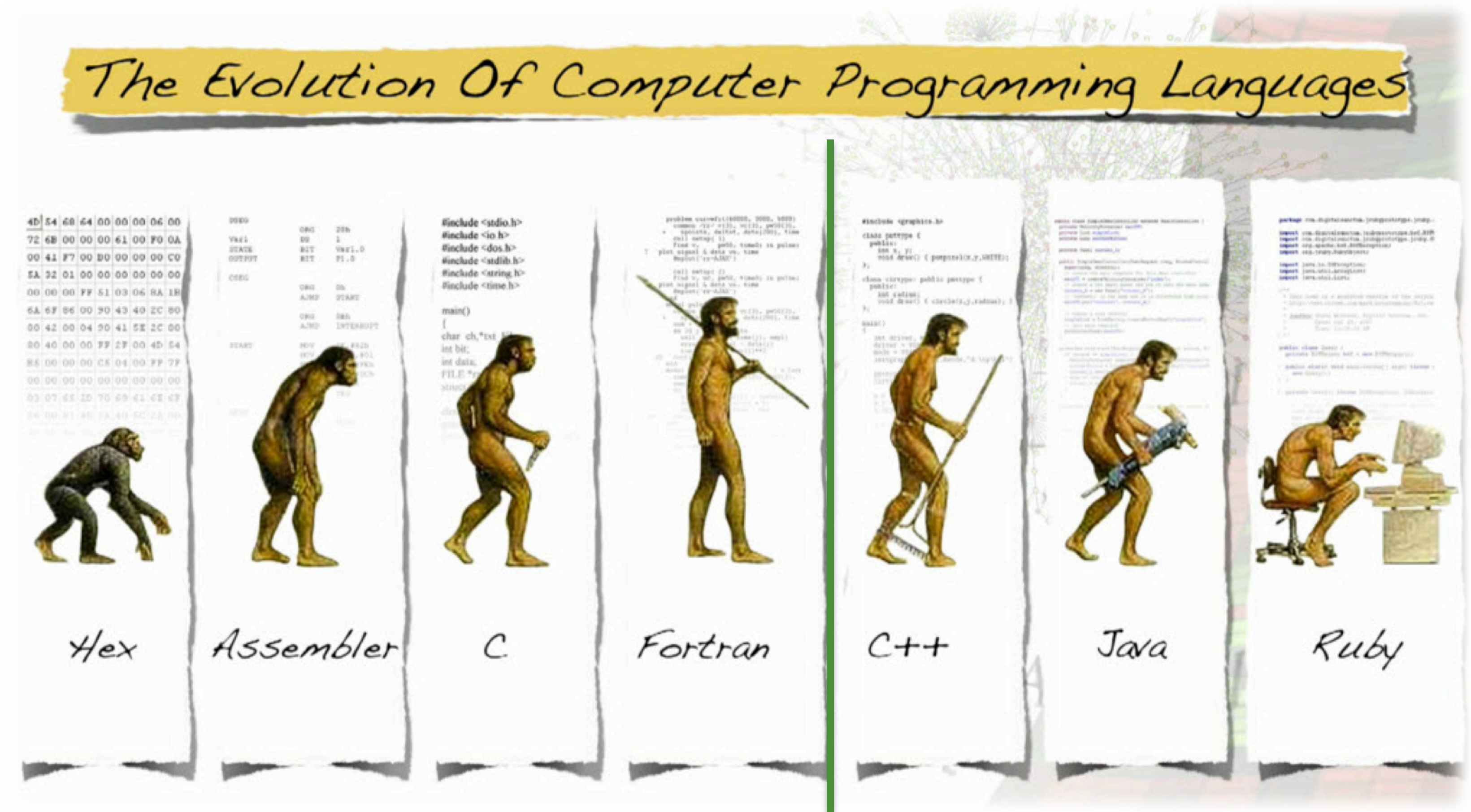
Announcements

- Practice Java
- Office hours: Tue/Thu 2-4pm; or email me for other time
- Move tech interview lecture to the week of Oct 13
- Create GitHub account
  - https://github.com/
  - Email me your GitHub ID and your name by Friday
- Learn Git and Github
  - https://help.github.com/
  - https://try.github.io/
  - https://www.youtube.com/watch?v=r63f51ce84A

Modern Programming Languages



OOP Came After PP



Procedural Programming (PP)

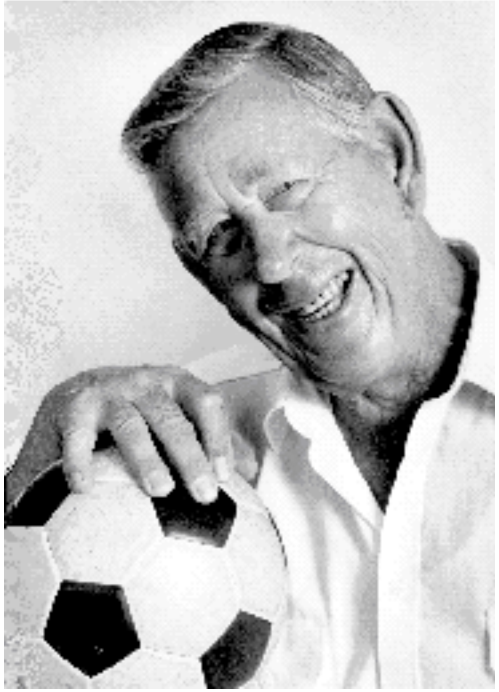
Object-Oriented Programming (OOP)

Soccer Penalty Cards History

Soccer Penalty Cards History

Soccer Penalty Cards History

The penalty cards were used for the first time in 1970 World Cup in Mexico



Ken Aston (1915 – 2001)

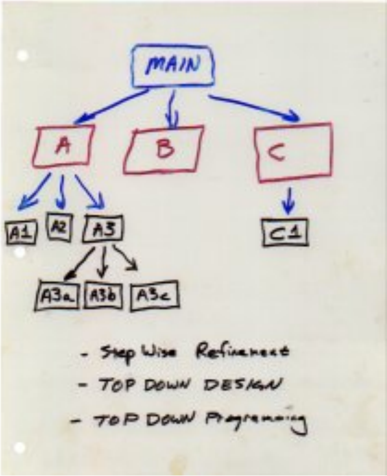
"As I drove down Kensington High Street, the traffic light turned red. I thought, 'Yellow, take it easy; red, stop, you're off'."

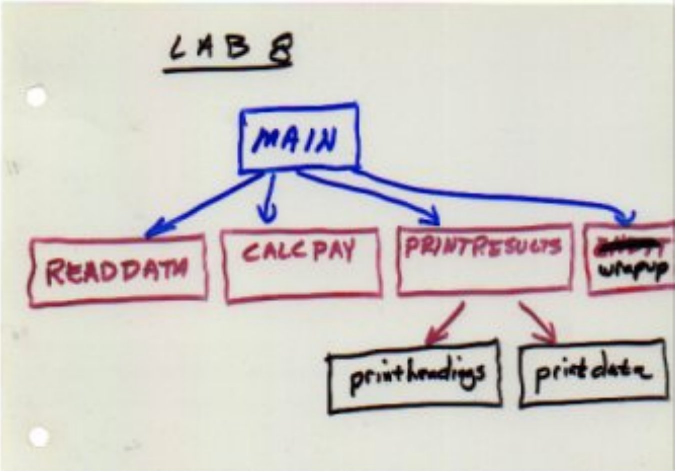
What is Procedural Programming

- *Procedural Programs:*
  - are made up of procedures (e.g., routines, subroutines, methods, or functions)
  - represent a list of instructions telling a computer what to do step by step and how to perform from the first code to the second code
- Traditional programming languages were procedural
  - C, Pascal, BASIC, FORTRAN

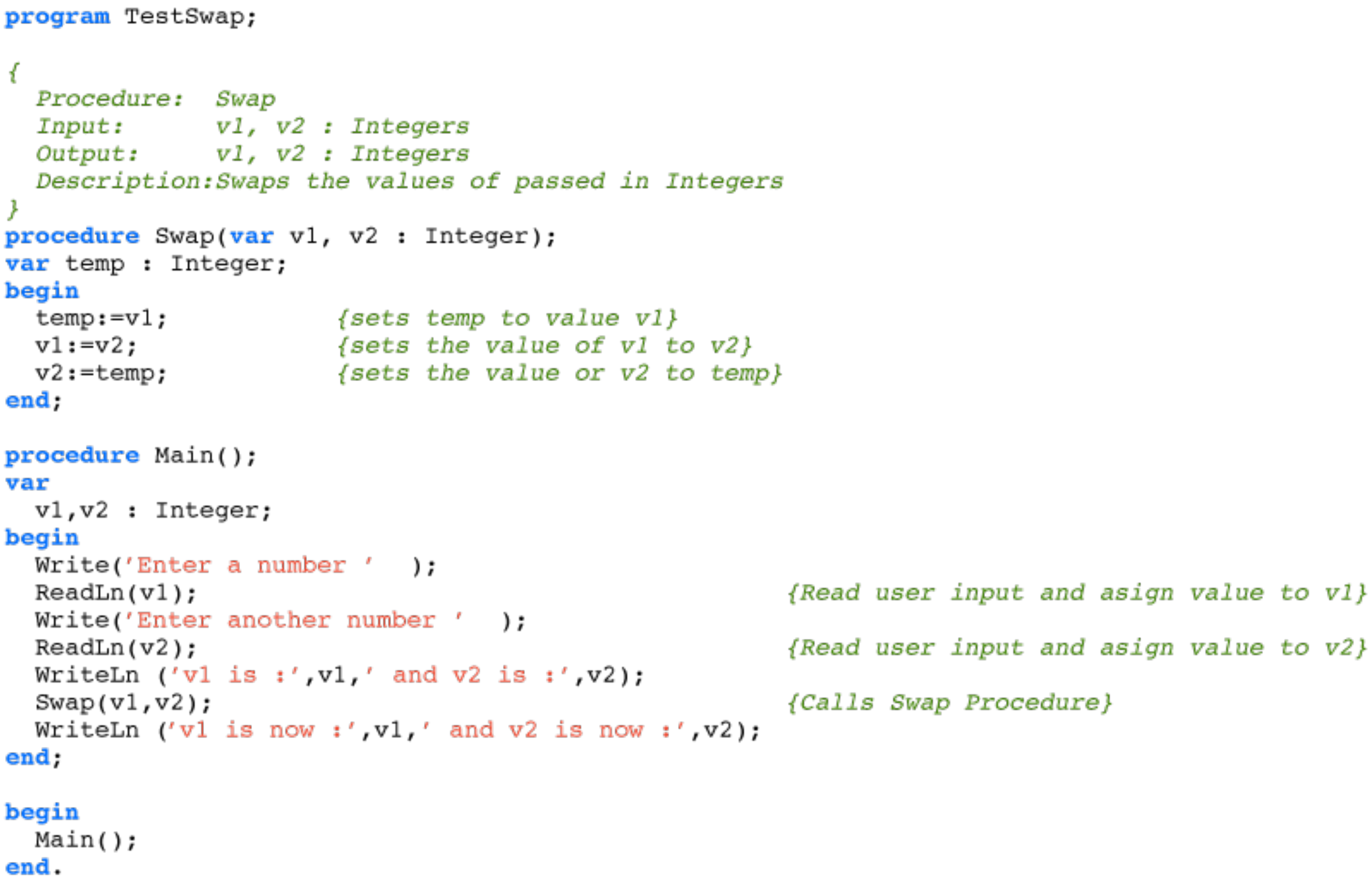
Procedural Programming Design

- *Top Down Design*
  - Functional decomposition – a problem (procedure) is systematically broken down into sub problems (sub procedures)
  - Functional decomposition continues until a sub problem is straightforward enough to solve





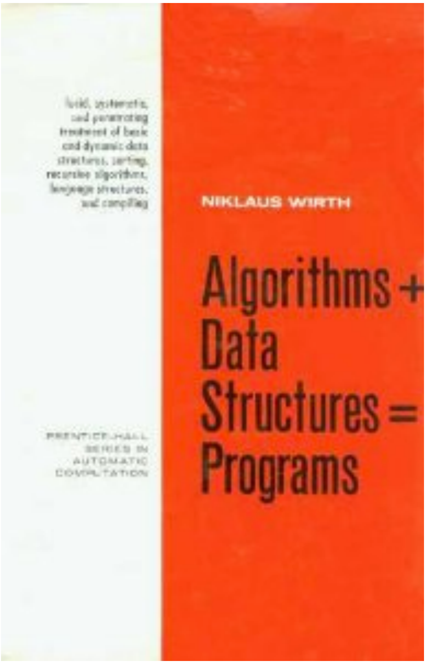
Sample Code in Pascal

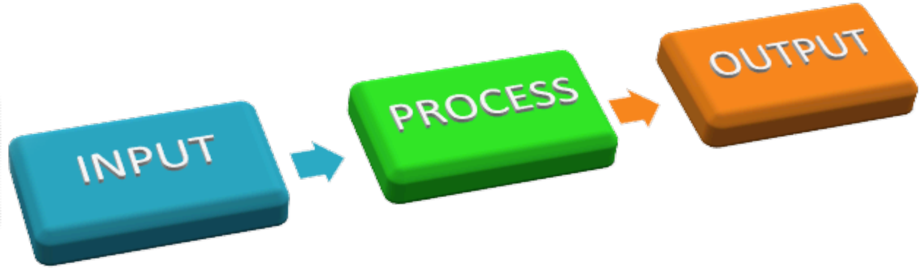


```
program TestSwap;

{
  Procedure:  Swap
  Input:      v1, v2 : Integers
  Output:     v1, v2 : Integers
  Description:Swaps the values of passed in Integers
}
procedure Swap(var v1, v2 : Integer);
var temp : Integer;
begin
  temp:=v1;           {sets temp to value v1}
  v1:=v2;             {sets the value of v1 to v2}
  v2:=temp;           {sets the value or v2 to temp}
end;

procedure Main();
var
  v1,v2 : Integer;
begin
  Write('Enter a number '   );
  ReadLn(v1);                                        {Read user input and asign value to v1}
  Write('Enter another number '   );
  ReadLn(v2);                                        {Read user input and asign value to v2}
  WriteLn ('v1 is :',v1,' and v2 is :',v2);
  Swap(v1,v2);                                       {Calls Swap Procedure}
  WriteLn ('v1 is now :',v1,' and v2 is now :',v2);
end;

begin
  Main();
end.
```

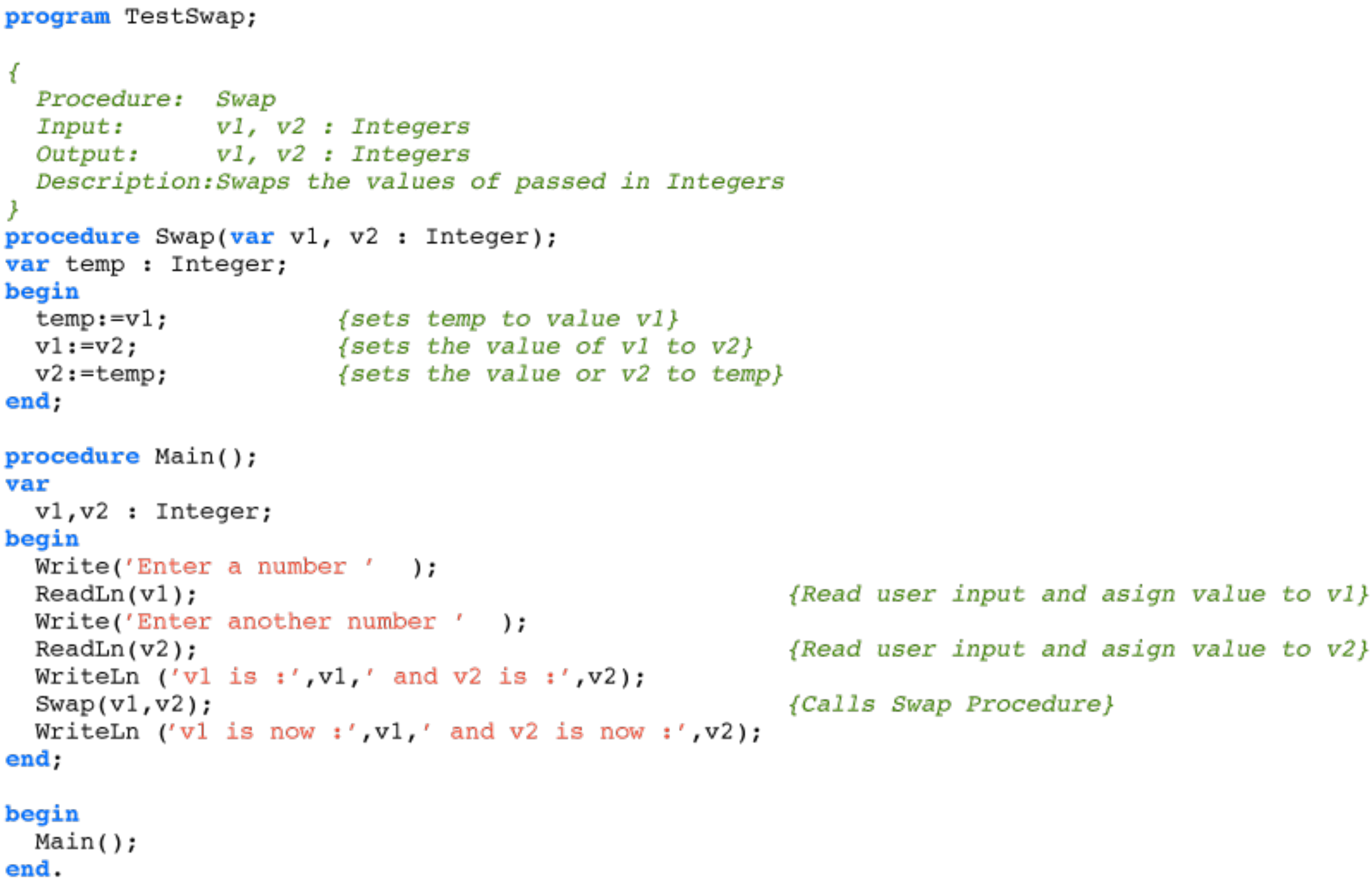
Procedural Programming Design

- *Algorithms + Data Structures = Programs*
  - Algorithms are implemented in procedures
  - Procedures take specific data structures as input and generate other types of data as output


lucid, systematic, and penetrating treatment of basic and dynamic data structures, sorting, recursive algorithms, language structures, and compiling
NIKLAUS WIRTH
Algorithms + Data Structures = Programs
PRENTICE-HALL SERIES IN AUTOMATIC COMPUTATION

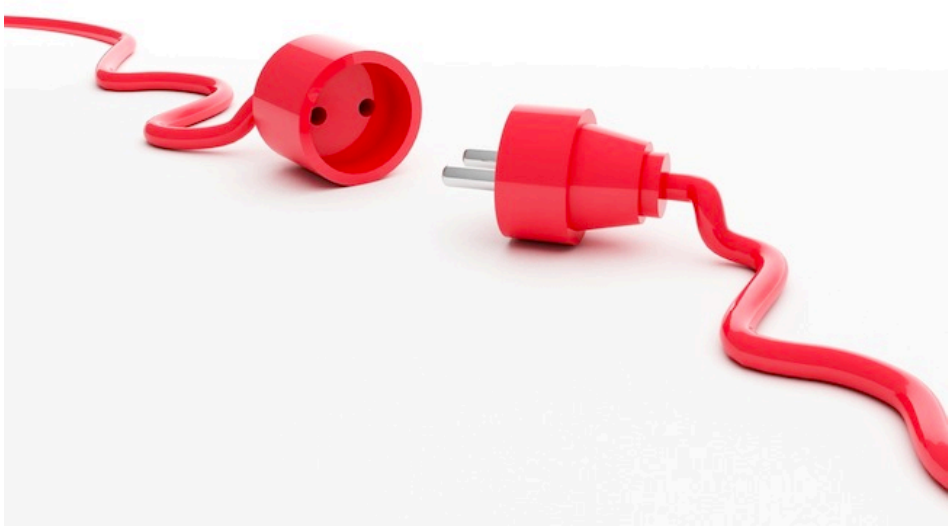


Sample Code in Pascal



```
program TestSwap;

{
  Procedure:  Swap
  Input:      v1, v2 : Integers
  Output:     v1, v2 : Integers
  Description:Swaps the values of passed in Integers
}
procedure Swap(var v1, v2 : Integer);
var temp : Integer;
begin
  temp:=v1;           {sets temp to value v1}
  v1:=v2;             {sets the value of v1 to v2}
  v2:=temp;           {sets the value or v2 to temp}
end;

procedure Main();
var
  v1,v2 : Integer;
begin
  Write('Enter a number '  );
  ReadLn(v1);                                       {Read user input and asign value to v1}
  Write('Enter another number '  );
  ReadLn(v2);                                       {Read user input and asign value to v2}
  WriteLn ('v1 is :',v1,' and v2 is :',v2);
  Swap(v1,v2);                                      {Calls Swap Procedure}
  WriteLn ('v1 is now :',v1,' and v2 is now :',v2);
end;

begin
  Main();
end.
```

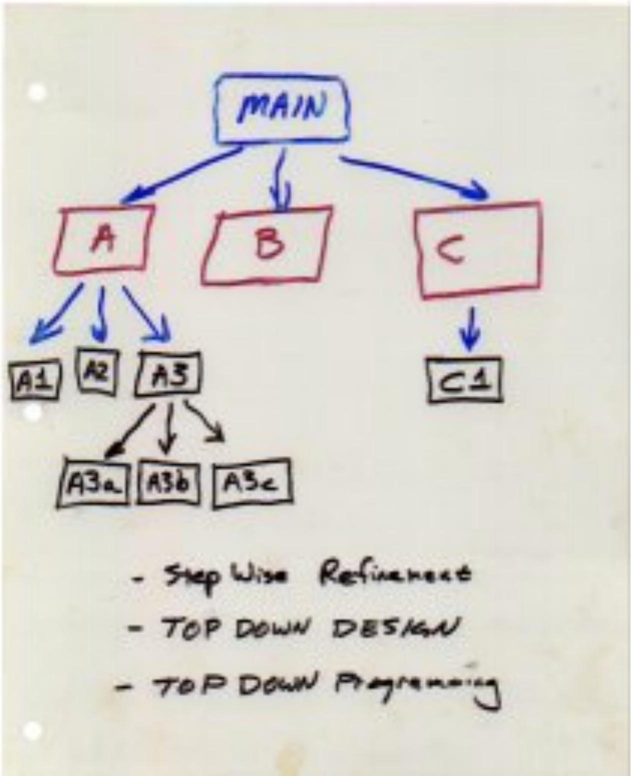
Problems with PP - Expressiveness



- Procedural languages are difficult to relate with the real world objects.
- Disconnected from real world problems.

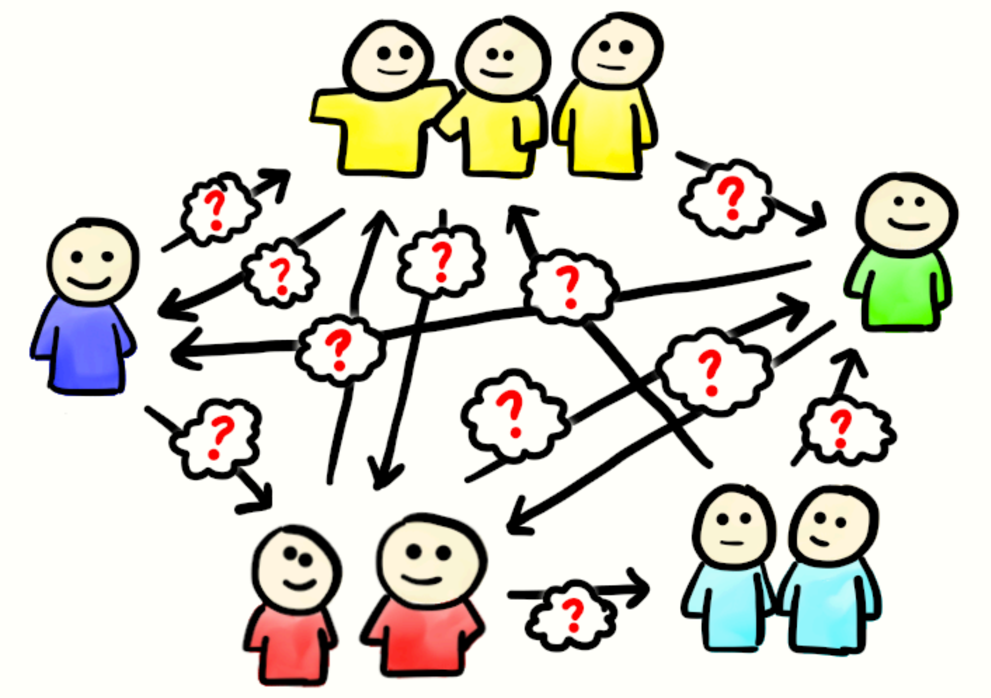
Problems with PP - Extensibility

- Top-down design does not scale with top-level changes
- Changes in algorithm or data structures affect each other



INPUT
PROCESS
OUTPUT

Problems with PP – Security



- Data used in procedural languages are exposed to the whole program (procedures)
- No security for the data

Build a SMS (Short Message Service)

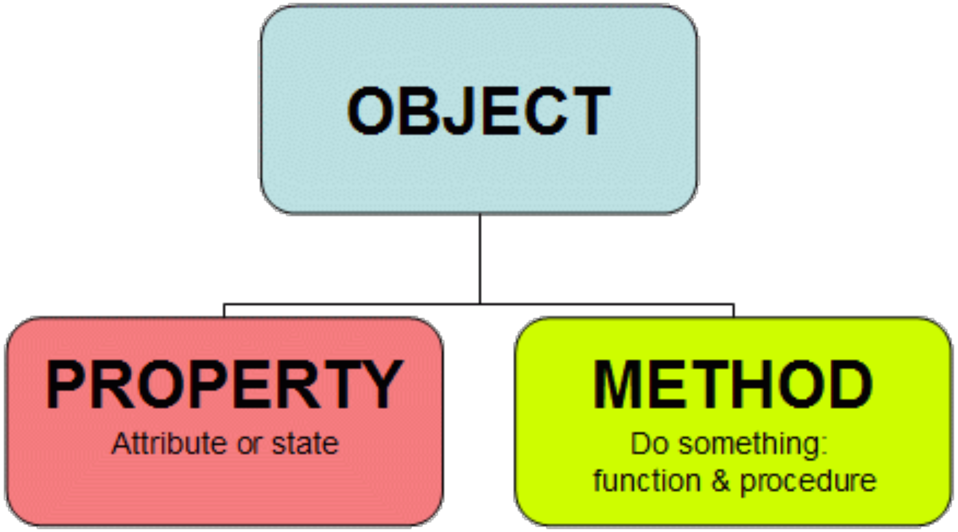
- Send text messages
  - Message input component
  - Send the message
- Receive test messages
  - Receive the message
  - Display the message in the inbox

Motivation for OOP

- Real world connection
  - We build software to solve real world problems.
  - People have real world problems.
  - *Therefore, we build software for people.*
- Extensible
  - Good software not only solves immediate problems, but it can be maintained and modified to address the inevitable changes that the customer will want.
- Secure
  - Exposing data to everywhere prevents the software from being the extensible good software. More importantly, it is not secure.

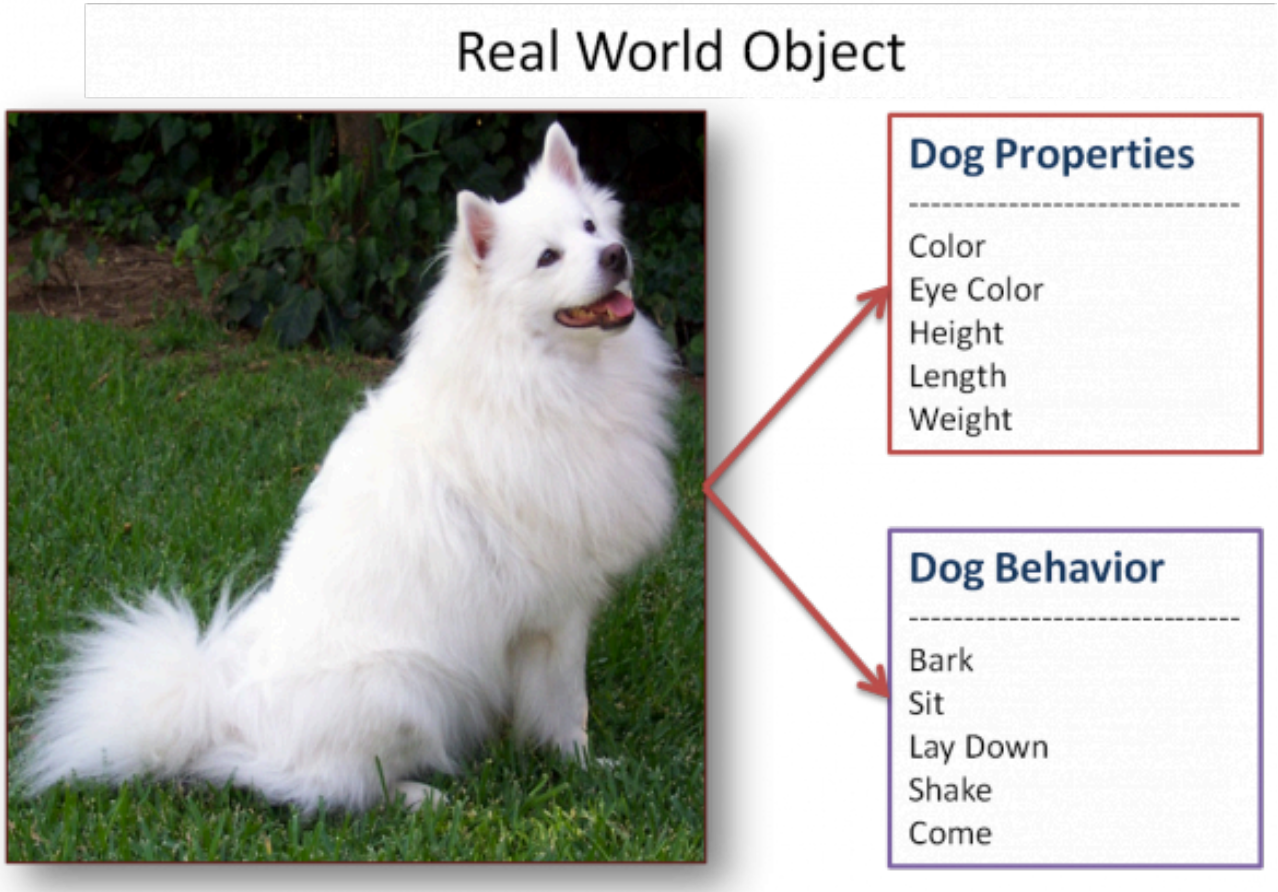
OOP

- A class is a combination of state (properties) and behavior (methods)
- An object is an instance of a class



OOP Connects Real World Objects

- An inheritance solution requires 15 subclasses to represent each type of view



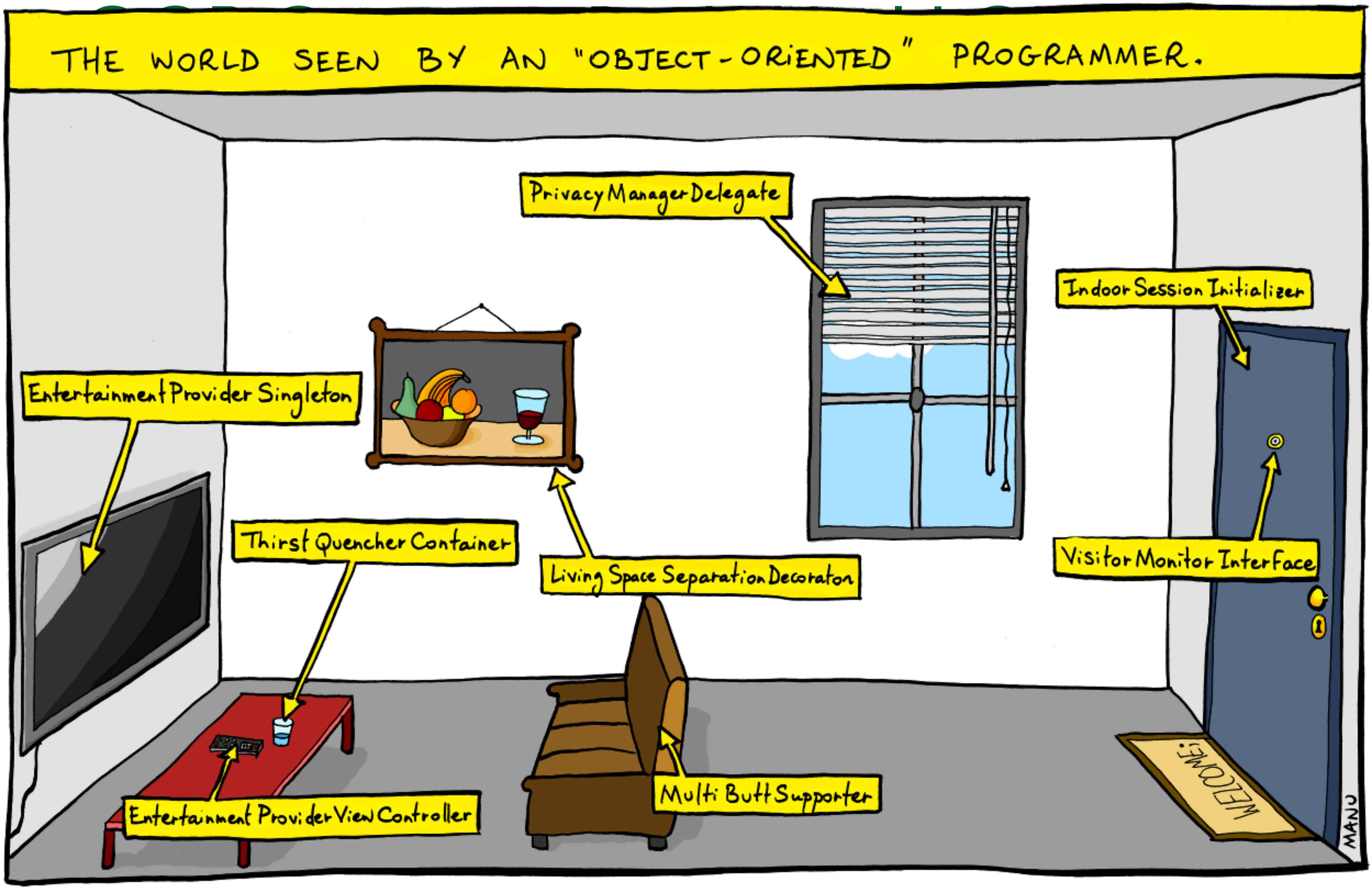
Real World Object

Dog Properties

Color
Eye Color
Height
Length
Weight

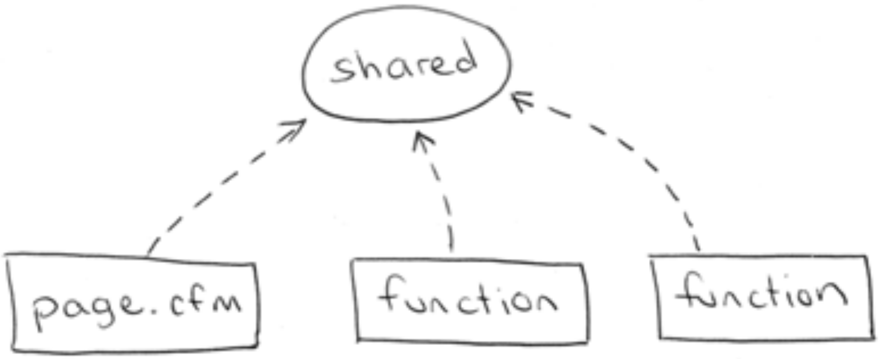
Dog Behavior

Bark
Sit
Lay Down
Shake
Come

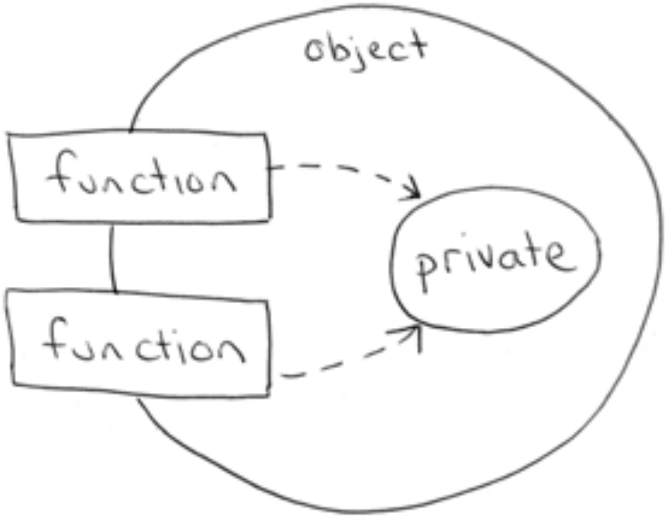

THE WORLD SEEN BY AN "OBJECT-ORIENTED" PROGRAMMER.
Privacy Manager Delegate
Indoor Session Initializer
Entertainment Provider Singleton
Thirst Quencher Container
Living Space Separation Decorator
Visitor Monitor Interface
Multi Butt Supporter
Entertainment Provider View Controller
WELCOME!
MANU

OOP Encapsulates Data

- The ability of an object to hide its data and methods from the rest of the world



PP shares the data globally



OOP encapsulates data privately

OOP Supports Better Modularity

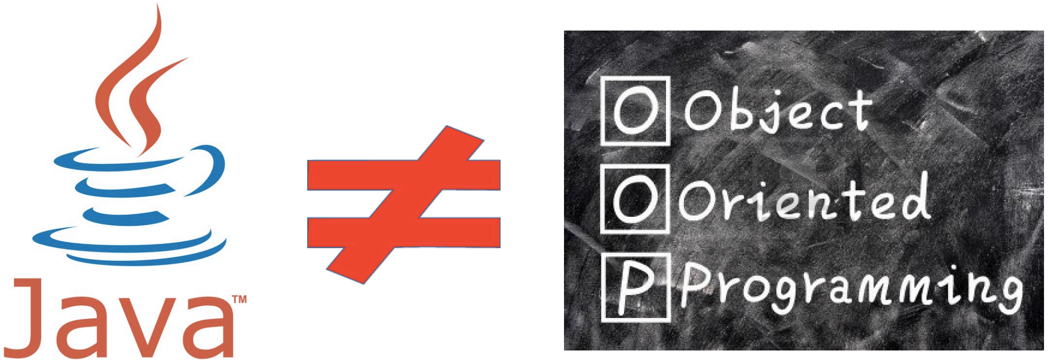
- The source code for a class can be written and maintained independently of the source code for other classes
- Once created, an object can be easily passed around inside the system

OOP Improves Reuse and Extensibility

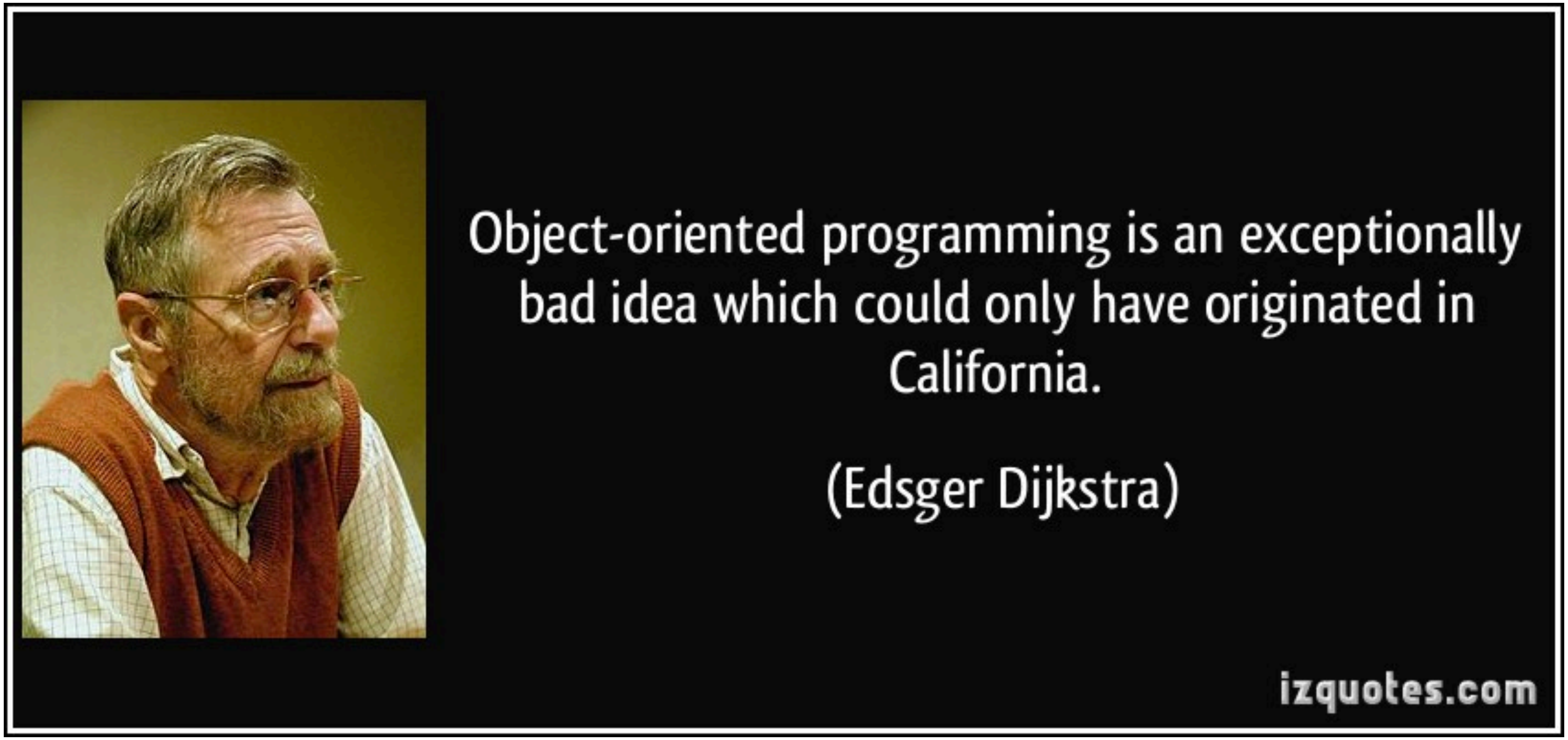
- If a class already exists, you can use objects from that class in your program.
- Object oriented programming languages allow classes to inherit commonly used state and behavior from other classes

Using Java Doesn't Mean Using OOP

- Using OOP means:
  - Data encapsulation
  - Information Hiding
  - Model problems using objects
  - Polymorphism



Disadvantages of OOP



Object-oriented programming is an exceptionally bad idea which could only have originated in California.

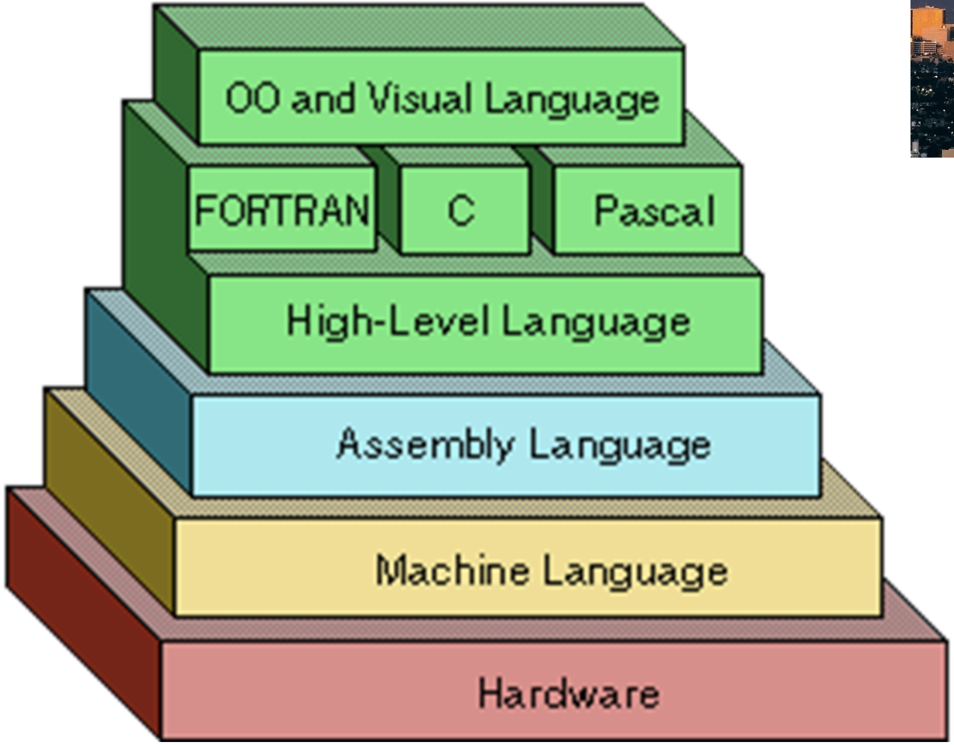
(Edsger Dijkstra)

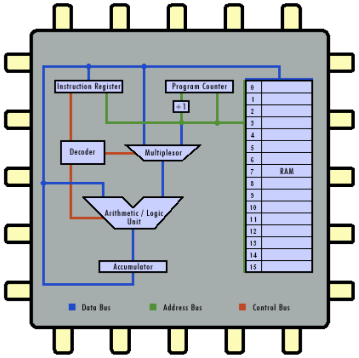
izquotes.com

Disadvantages of OOP

- Not every problem can be considered as objects
  - e.g., Procedure is not an object
- Steep learning curve
  - PP (Top-down) is more straightforward for us to think
- Large program size
- Slower programs / Less efficient

Level of Abstraction


OO and Visual Language
FORTRAN
C
Pascal
High-Level Language
Assembly Language
Machine Language
Hardware


Instruction Register
Program Counter
+1
Decoder
Multiplexor
Arithmetic / Logic Unit
Accumulator
RAM
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Data Bus
Address Bus
Control Bus