Object Oriented Programming

Assembly

lea bx, string

mov ch, 0

mov cl, [bx+1]

jcxz null

add bx, 2

BASIC

```
i$ = Input$()
If i$ = ""
   Print "SORRY, BUT I DIDN'T UNDERSTAND."
Goto userinputsection
EndIf
i$ = " " + i$ + " "
```

Structured Programming

- Form
  - Instances of scalar or matricial types
  - Control structures (if, while, …)
- No modularization
- Improvements (over test-and-jump programs)
  - Better control flow
  - Formal reasoning about programs
  - Readability

Not possible to hide implementation and use an interface

Procedural Programming

- Form
  - Instances of scalar or vector types
  - Control structures
  - Routines
- Routines as modules
  - Functions – Return a result
  - Procedures – Change the data
- Improvements
  - Encapsulation (of control flow)
  - Reutilization
  - Easier to debug
  - Can maximize cohesion and minimize connections

Object (or data) oriented programming

- Form
  - Instances of scalar types and vectors and ADT (abstract data types)
  - Control structures
  - Routines
  - Routines related with ADT
- Modularization
  - ADT – Data and operations on it together
  - Operations – Routines that operate over ADT
- Improvements
  - Better encapsulation (data and operations associated)
  - Data hiding
  - Changes programming perspective

Object Oriented Programming

- Form
  - Instances of classes (objects), scalar types and vectors and ADT
  - Control Structures
  - Routines
  - Routines related with ADT
  - Operations organized in classes
- Modularization
  - Classes – Models for objects with a given behavior
  - ADT – Data and operations on it
  - Operations – Routines that operate over ADT
  - Method – Operation implementations
- Improvements
  - Extension and specialization
  - Changes programming perspective

Object Oriented Programming

- Uses
  - Structured Programming – Flow Control
  - Procedural Programming – Routines
  - Data Centered Programming – ADT and operations

- Advantages
  - Modularization
  - Encapsulation
  - Reusability
  - Extension and specialization
  - More expressive
  - Flexibility
  - Robust

Object Oriented Programming

- Interface:
  - Operations – Implemented in one or more methods
  - Properties – May be implemented using attributes or not
- Implementation
  - Methods – Operation implementation
  - Attributes – Data is part of class implementation

Object Oriented Programming

- Everything is an object
- Objects have responsibilities, behaviors and properties
- Organization of programs reflects reality …

Object Oriented Programming

- Analyzing a problem
  - Which objects?
  - What are their responsibilities?
  - How do they cooperate?
  - How are objects classified?

- Solution design
  - What classes to define?
  - What objects to build?
  - What responsibilities to give them?
  - How to make them cooperate?

More information / References

- Y. Daniel Liang, "Introduction to Java Programming" 7th Ed. Prentice-Hall, 2010

Summary

- Object Oriented Programming