Copyright Warning

Object Oriented Design (OOP - History)

Week 3-1

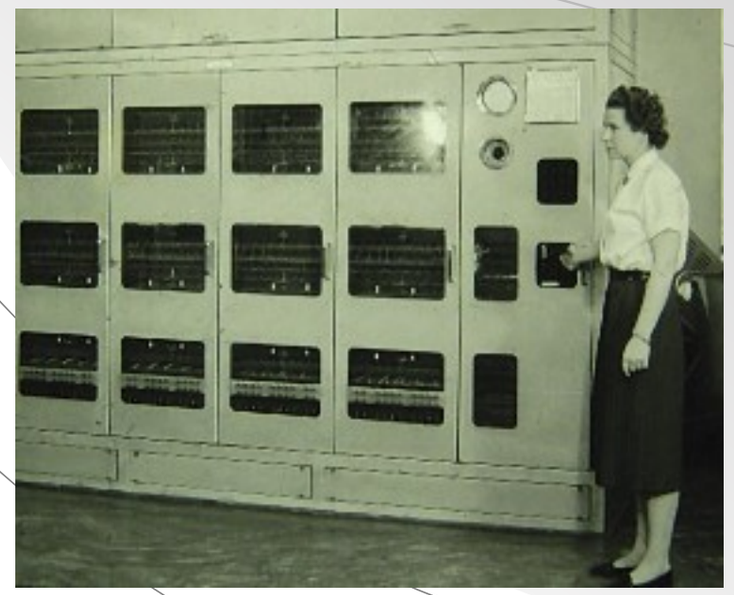
OOTech is not to replace Old Soft Dev Tech

- OOP is based on the previous programming technologies
- Compensate shortcomings of prev. approach
- OOTs also advancement of prev. approach.

Bit of History

- from
- Machine Language to
- Structural Programming

Machine Lang.



SILLIAC - the University of Sydney's first computer

- Computer can understand
  - Machine Language : Binary Numbers
- We need to have commands written in Machine Language
- around 40's and 50's....programmers wrote programs with machine lang.

Example

```
0xeb, 0x4e, 0x90, 0x48, 0x45, 0x4c, 0x4c, 0x4f
0x49, 0x50, 0x4c, 0x00, 0x02, 0x01, 0x01, 0x00
0x02, 0xe0, 0x00, 0x40, 0x0b, 0xf0, 0x09, 0x00
0x12, 0x00, 0x02, 0x00, 0x00, 0x00, 0x00, 0x00
```

- very first part of my old OS...
- move a head of FDD to find a bootloader...
- very inefficient....not may people can read

First Step: Assembly Lang.

- Replaced Machine Lang. with Symbols.

```
A10010          MOV   AX, X
8B160210        MOV   DX,Y
01D0            ADD   AX, DX
A10410          MOV   Z,AX
```

- Still hard to read, but you can guess.
- "Assembler" reads code in Assembly Lang and produce Machine Lang.

First Step: Assembly Lang.

- Computer is for doing some tasks.
  - Computer needs programs
- Let's use a computer to help making computer.
  - Programming Lang is for make coding process easy!

2nd Step: High-Level Lang.

- Can we make it more easier?
  - instead of describing each command (for a machine)
    - use representation understood by human
- FORTRAN (1957)

  z=x+y

2nd Step: High-Level Lang.

- Even those old rather primitive High-Level Langs
  - helped improving the process of programming
- but...by the end of 20th cent. we have “Software Crisis”
  - not enough programmers.
- Software Engineering
  - Prog. Lang, Prog. Tech., Soft Dev. Processes...

3rd Step: Structural Prog.

- Proposed by Edsger W. Dijkstra
- To make a program, which behave correctly
- important to give a easy to understand structure to the program.

3rd Step: Structural Prog.

- example
  - don't use "GO TO" statement
- use
  - Sequence
  - Selection : if..then..else..endif, switch/case
  - Repetition : for, while, do...repeat, repeat

Spaghetti Code

- in 70's 80's
  - very small memory
  - slow CPU
- Programmers tried to write programs that are
  - small (byte), small (steps)
- This made programming very tricky.

Spaghetti Code

- Program with very tricky code plus
- GOTO statement

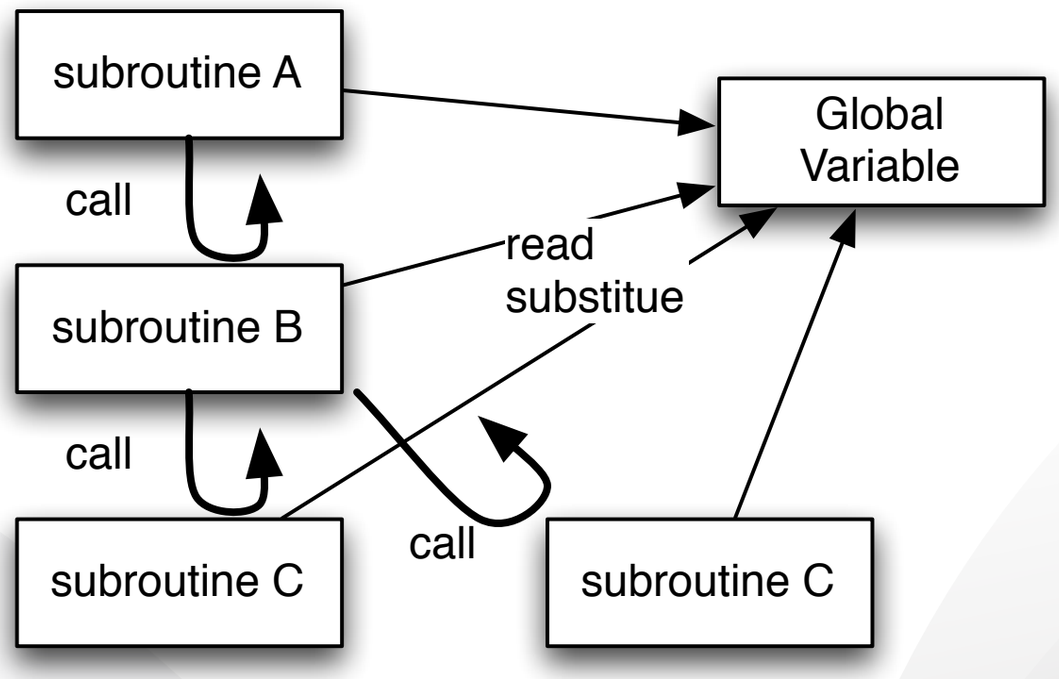
- very difficult to see the overall structure of the code....Spaghetti code
- criticism: structural programming...slow

Subroutine

- To support easy maintenance
  - Increase independence of subroutines
- subroutine itself was there in 40's
  - group common commands together
  - to make the code small.
- people realised that independent subroutine helps software maintenance.....HOW?

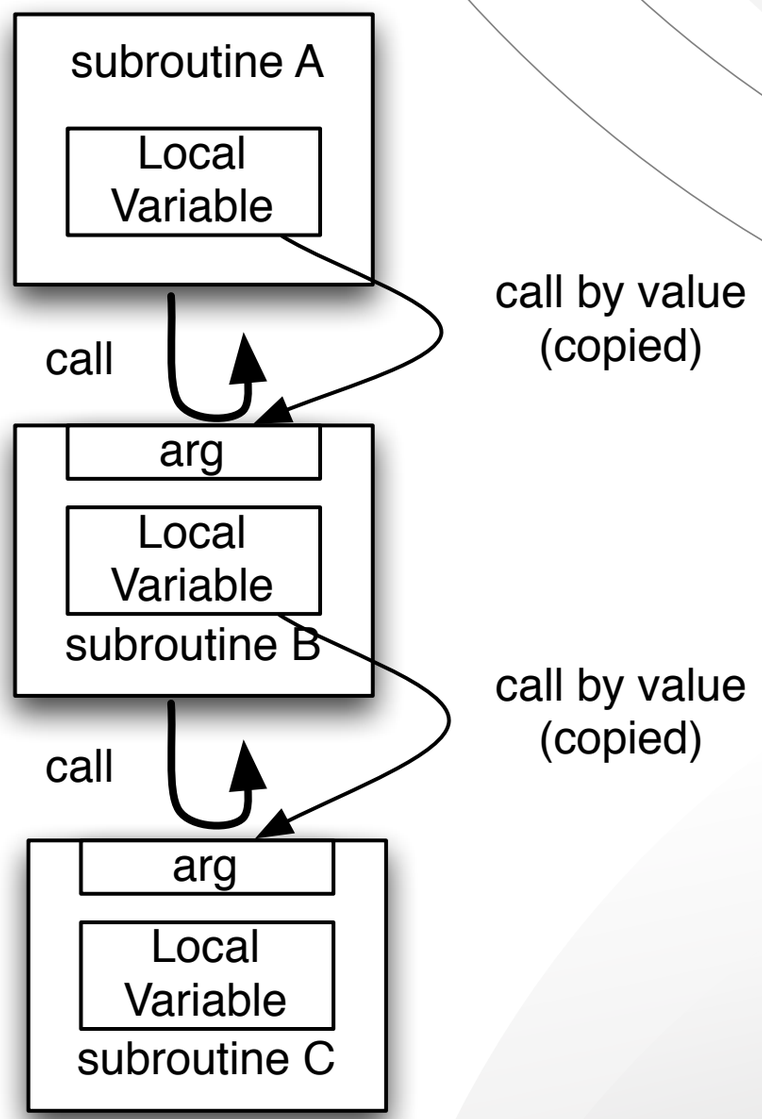
Independent Subroutine

- To increase independency
  - decrease shared pieces of info. (variables)among subroutines
- Shared pieces of info.....Global Variables



Avoid Global Variables

- To avoid the negative effect of Global Variables
  - local variables
  - Call by Value



Prog. Lang. for GOTO-less Programming

- C, Pascal, ALGOL...
  - if statement
  - case statement
  - for statement
  - while statement....
- old FORTRAN, COBOL had difficulty in using those. (modern ones do support them)

Prog. Lang. for GOTO-less Programming

- C, Pascal, ALGOL...
- local variables
- call by value

but...

- I've seen "GOTO" in C, Pascal,...
  - for getting out from a deeply nested loop structure
  - increase the execution efficiency...(long time ago..)

C is good.

- supports
  - Structural Programming
  - bit operation
  - pointer for memory manipulation/ management
  - add capabilities by libraries of functions.
- C++, Java, C#, etc. inherits C's capabilities.

Global Variables / Reusability

- These are still big problems after Structural Programming
- OOP came to our rescue.