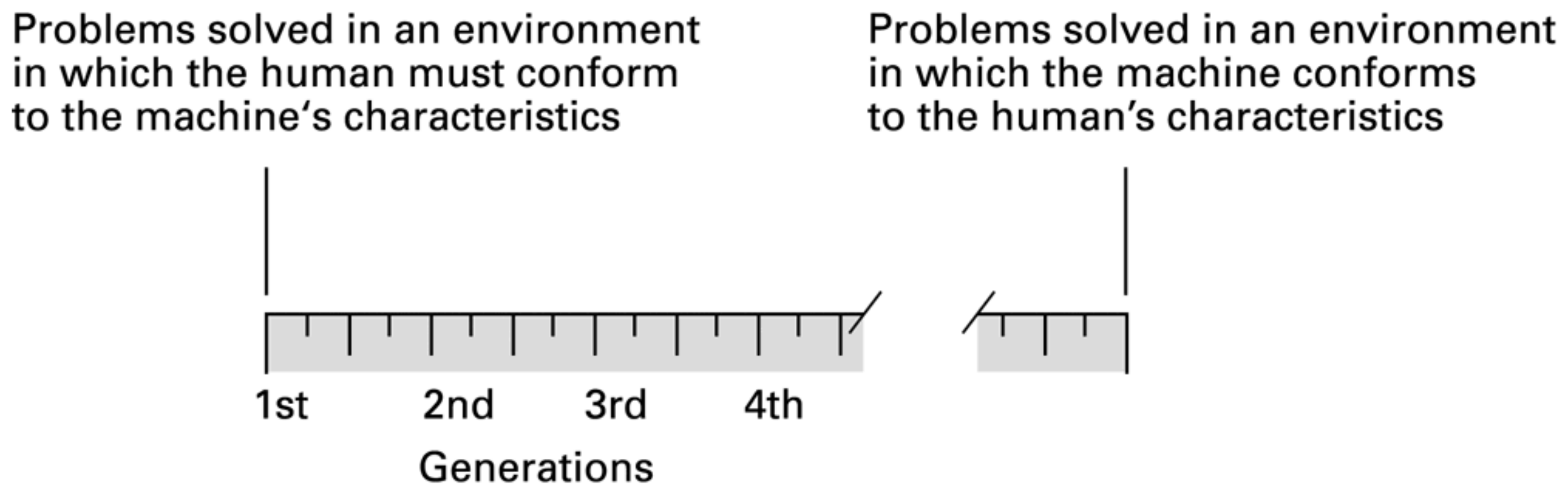
Programming and Software

http://www.techshownetwork.com/

History of programming languages


Problems solved in an environment in which the human must conform to the machine's characteristics
Problems solved in an environment in which the machine conforms to the human's characteristics
1st
2nd
3rd
4th
Generations

First-generation: Machine code

- The programmer enters the binary / hexadecimal codes that the CPU / ALU will run directly

See the video recreation of programming a PDP- 11 in the late 1970s.

http://www.youtube.com/watch?v=XV-7J5y1TQc

Jump to 3’30”. *Don’t look at anything before.*

Second-generation: Assembly language

- A mnemonic system for representing machine instructions
  - Mnemonic names for op-codes
  - Identifiers: Descriptive names for memory locations, chosen by the programmer

Assembly Language Characteristics

- One-to-one correspondence between machine instructions and assembly instructions
  - Programmer must think like the machine
- Inherently machine-dependent
- Converted to machine language by a program called an **assembler**

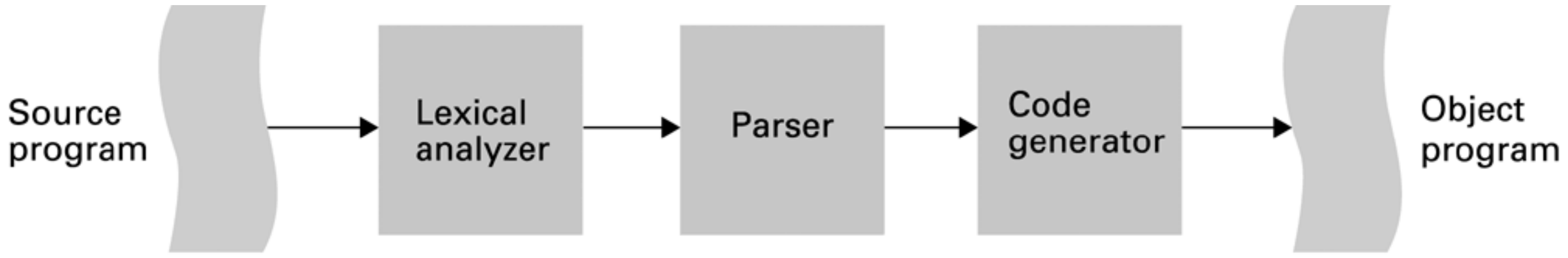
Program Example

Machine language	Assembly language
156C	LD R5, Price
166D	LD R6, ShippingCharge
5056	ADDI R0, R5 R6
30CE	ST R0, TotalCost
C000	HLT

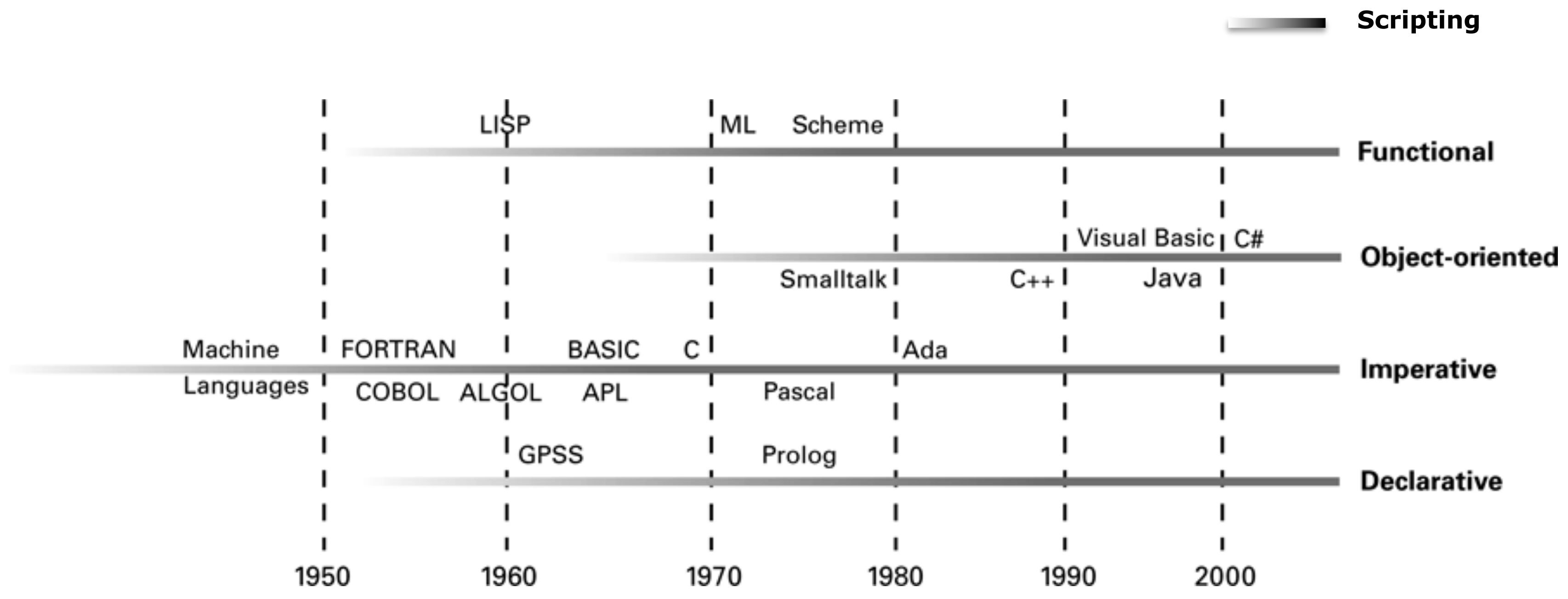
Third Generation Language

- Uses high-level primitives
  - Similar to our pseudocode in Chapter 5
- Machine independent (mostly)
- Examples: FORTRAN, COBOL
- Each primitive corresponds to a sequence of machine language instructions
- Converted to machine language by a program called a **compiler**

What a compiler does



The historical evolution of programming paradigms



BASIC Program Example

```
10 REM  Sample BASIC Program
20 REM  By Les Carr
30 REM
40 PRINT "Hello World!"
45 LET S=0
50 FOR C=1 TO 10
60 PRINT C, C*C
65 LET S=S+C
70 NEXT C
80 PRINT "The sum of 1 to 10 is", C
90 STOP
100 END
```

BASIC allows you to jump around the line numbers using 'GOTO' commands. This is now considered EVIL.

- Line numbers used to order the statements
  - dates back to punched cards
- Rather than named locations, talk about *variables* with *values*
- REM is *remark*
- Iteration is achieved by FOR var = firstval to lastval

Prolog Program Example

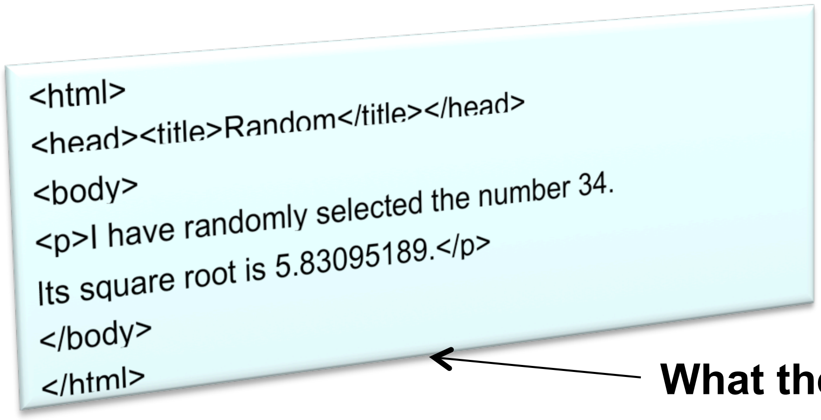
```
parent(les, daisy). parent(les, joel).
parent(les, sam). parent(les, ruby).
parent(peter, les).
parent(herbert, peter).
parent(adam, herbert).

child(X,Y) :- parent(Y,X).
grandparent(X,Y):-
  parent(X,Someone), parent(Someone,Y).

ancestor(X,Y):-
  parent(X,Y).
ancestor(X,Y):-
  parent(X,Someone),ancestor(Someone,Y).
```

- Logic programming
  - Predicate logic
- Assert facts
  - relation(a,b,c).
- Define *rules* about the relationships

PHP Program Example

```
<html>
<head><title>Random</title></head>
<body>
<p>I have randomly selected the number
<?php $x=rand(1,100); echo $x;?>.
Its square root is <?php echo sqrt($x);?>.</p>
</body>
</html>
```

What PHP on the server sees



```
<html>
<head><title>Random</title></head>
<body>
<p>I have randomly selected the number 34.
Its square root is 5.83095189.</p>
</body>
</html>
```

What the client browser sees

- A scripting language used on HTML servers
- Builds web pages to send pure HTML to web browser
- Calculates values, talks to databases…
- Similar to JSP, ASP etc

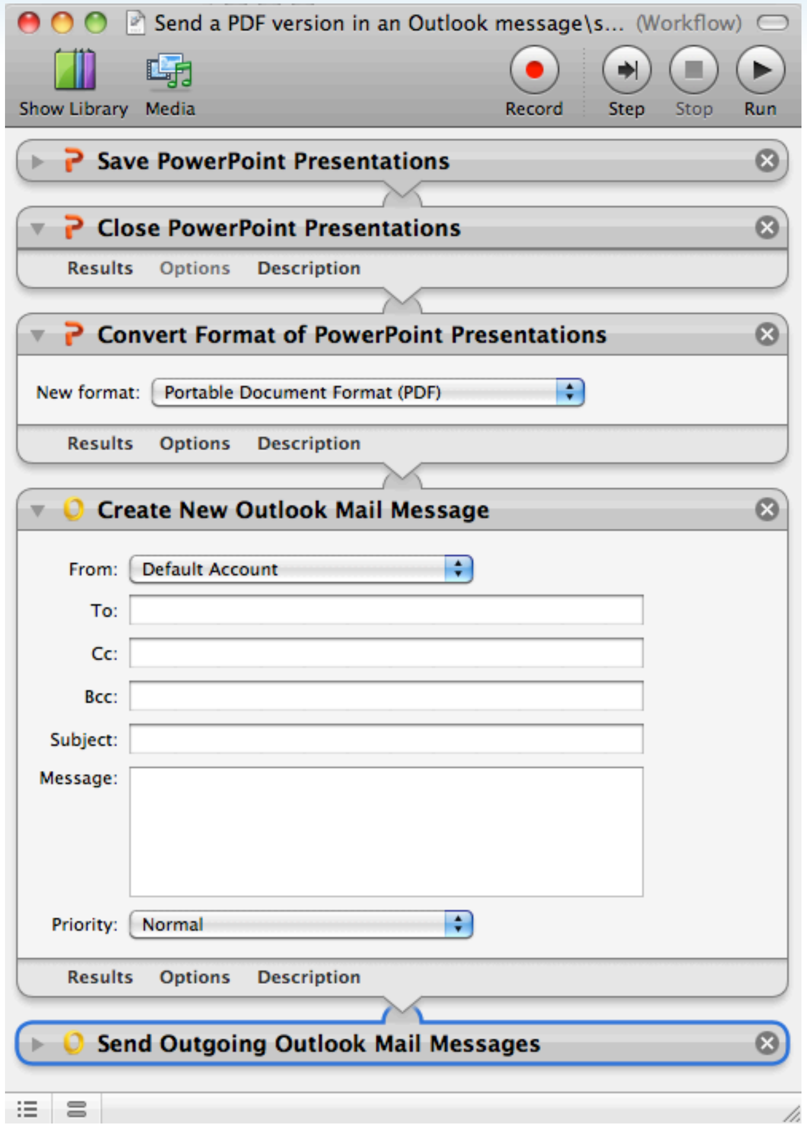
Applescript Program Example

```
tell application "iTunes" to play playlist
        named "My Top Rated”
```

A *scripting language* used on the Apple Mac to control other applications

Automator Program Example

A *visual scripting language* used on the Apple Mac to control other applications



Save PowerPoint Presentations

Close PowerPoint Presentations

Convert Format of PowerPoint Presentations

New format: Portable Document Format (PDF)

Create New Outlook Mail Message

From: Default Account

To:

Cc:

Bcc:

Subject:

Message:

Priority: Normal

Send Outgoing Outlook Mail Messages

Object-oriented programming

- The most recent paradigm

- **Programming** = modeling the real world
- The **real world** is composed of objects
- **Objects** are a combination of
  - state (properties, values, data)
  - behaviour (capabilities, functionality)

Object-oriented Concepts

- **Classes:** similar objects are instances of a single class
- **Inheritance:** new classes can be defined in terms of previously defined classes
  - Saves hard work, avoids duplication
- **Polymorphism:** allows very different classes of object to have same functionality e.g.
  - Students take a module
  - Lecturers take a module

Programming: Increasing Scale and Complexity

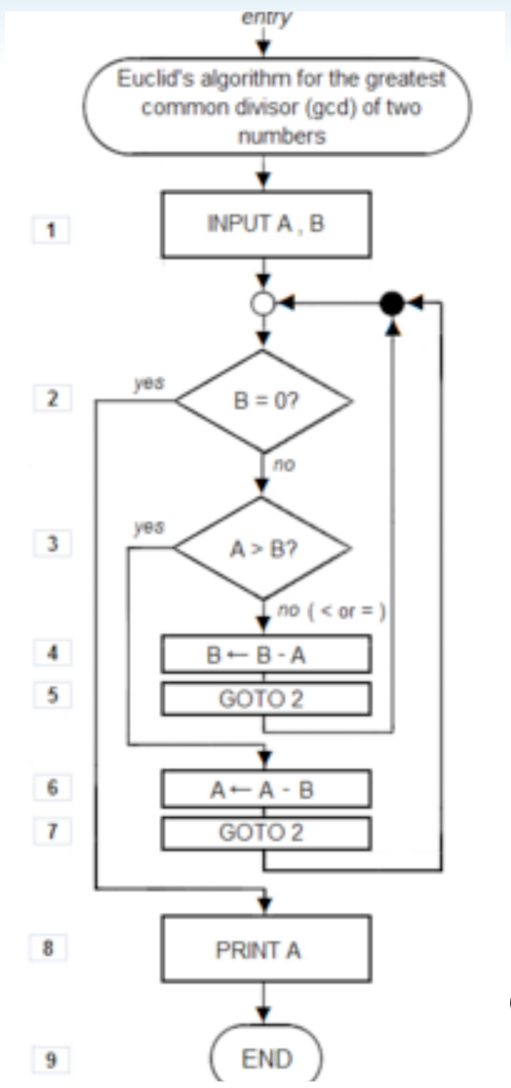
- Machine code: a human can manage little functionality with small amount of state
  - A modern laptop has 100,000x as much memory
  - 10^5 times as much complexity

- OO programming makes it easier for human software developers (and teams of developers) to control 10^5 complexity

Algorithms

- An algorithm is a set of rules that defines the sequence of operations to do something
- Usually defined either by Flowcharts or by “Pseudocode”.

Flowchart



Oblongs are terminators
Rectangles are actions
Diamonds are binary decision points

(From Wikipedia)

Pseudocode

E.g. Binary “Chop” – searches *ordered* array of N integers for the integer x

```
min := 1
max := N
repeat
    mid := (min+max) div 2
    if x > A[mid] then min := mid + 1
    else max := mid – 1
until (A[mid] = x) or (min > max);
if A[mid] = x
    output x
else
    output ‘Not Found’
```