What is a language?

Webster:

A systematic means of communicating ideas or feelings by the use of conventionalized signs, sounds, gestures, or marks having understood meanings.

Linguist’s Definition

(Charles Yang)

A language is:

A description of pairs ($S$, $M$), where $S$ stands for sound, or any kind of surface forms, and $M$ stands for meaning.

A theory of language must specify the properties of $S$ and $M$, and how they are related.

Languages and Formal Systems

What is the difference between a formal system and a language?

With a language, the surface forms have meaning.

Caveat: computer scientists often use language to mean just a set of surface forms.
What are languages made of?

• **Primitives** (almost all languages have these)
  - The simplest surface forms with meaning
• **Means of Combination** (all languages have these)
  - Like Rules of Production for Formal Systems
  - Ways to make new surface forms from ones you already have
• **Means of Abstraction** (all powerful languages have these)
  - Ways to use simple surface forms to represent complicated ones

Does English have these?

• **Primitives**
  - Words (?)
• **Means of combination**
  - ?

Does English have these?

• **Primitives**
  - Words (?)
• **Means of combination**
  - e.g., “antifloccipocinilipilification” - not a primitive
  - Morphemes - smallest units of meaning
    - e.g., anti- (“opposite”)
• **Means of abstraction**
  - e.g., Sentence ::= Subject Verb Object
  - Precise rules, but not the ones you learned in grammar school

The “**these**” in the slide title is an abstraction for the three elements of language introduced 2 slides ago.

The “**they**” in the confusing sentence is an abstraction for pronouns.

How should we describe (Formal) Languages?

Backus Naur Form

symbol ::= replacement

We can replace symbol with replacement

A ::= B means anywhere you have an A, you can replace it with a B.

**nonterminal** - symbol that appears on left side of rule

**terminals** - symbol that never appears on the left side of a rule
BNF Example

Sentence ::= NP Verb
NP ::= Noun
Noun ::= Wes
Noun ::= Scheme
Verb ::= rocks
Verb ::= sucks

What are the terminals?
How many different things can we express with this language?

4, but only 2 are true.

BNF Example

Sentence ::= NP Verb
NP ::= Noun
NP ::= Noun and NP
Noun ::= Wes
Noun ::= Scheme
Verb ::= rocks
Verb ::= sucks

How many different things can we express with this language?

Infinitely many!
Recursion is powerful.

Liberal Arts Trivia: Art History

Q. Name the type of painting in which pigment is mixed with water on a thin layer of mortar or plaster. Because of the chemical makeup of the plaster, a binder is not required, as the pigment mixed solely with the water will sink into the intonaco, which itself becomes the medium holding the pigment. The technique was popular during the European Renaissance.

Liberal Arts Trivia: Music

Q. This Hong Kong singer is one of the original four cantopop Heavenly Kings ( 大天王 ), and possesses a rich baritone/tenor. He is sometimes called the God of Songs ( 歌神 ). His most famous work is perhaps Goodbye Kiss ( 吻别 ) - one of the best-selling albums of all time, with over 3 million copies sold in 1993 alone. Give the English or Romanized name of this singer.
Most Essential Scheme

Expr ::= PrimitiveExpr
PrimitiveExpr ::= Number
PrimitiveExpr ::= + | * | <= | ...
Expr ::= Name
Expr ::= ApplicationExpr
ApplicationExpr ::= (Expr MoreExprs)
MoreExprs ::= Expr MoreExprs
MoreExprs ::= Expr MoreExprs

This is everything you need to write for PS1!

Rules of Evaluation & People

ENIAC: Electronic Numerical Integrator and Computer

- Early WWII computer
  - But not the world’s first (PS4)
- Built to calculate bombing tables

Memory size:
- twenty 10 decimal digit accumulators = 664 bits
- ENIAC (1946): 3 mm
- Apollo Guidance Computer (1969): 1 inch
- You: 2.2 miles

Directions for Getting 6

1. Choose any regular accumulator (ie. Accumulator #9).
2. Direct the Initiating Pulse to terminal 5i.
3. The initiating pulse is produced by the initiating unit’s Io terminal each time the Eniac is started. This terminal is usually, by default, plugged into Program Line 1-1 (described later). Simply connect a program cable from Program Line 1-1 to terminal 5i on this Accumulator.
4. Set the Repeat Switch for Program Control 5 to 6.
5. Set the Operation Switch for Program Control 5 to ADD.
6. Set the Clear-Correct switch to C.
7. Turn on and clear the Eniac.
8. Normally, when the Eniac is first started, a clearing process is begun. If the Eniac had been previously started, or if there are random neons illuminated in the accumulators, the “Initial Clear” button of the Initiating device can be pressed.
9. Press the “Initiating Pulse Switch” that is located on the Initiating device.
10. Stand back.

Admiral Grace Hopper (1906-1992)

- Mathematics PhD Yale, 1934
- Entered Navy, 1943
- First to program Mark I (first “large” computer, 51 feet long)
- Wrote first compiler (1952) - program for programming computers
- Co-designer of COBOL (most widely used programming language until a few years ago)

"Nobody believed that I had a running compiler and nobody would touch it. They told me computers could only do arithmetic.”

USS Hopper
Code written by humans

Compiler translates from code in a high-level language to machine code

Code machine can run

DrScheme uses an interpreter. An interpreter is like a compiler, except it runs quickly and quietly on small bits of code at a time.

John Backus

- Chemistry major at UVA (entered 1943)
- Flunked out after second semester
- Joined IBM as programmer in 1950
- Developed Fortran, first commercially successful programming language and compiler

John Backus

- Chemistry major at UVA (entered 1943)
- Flunked out after second semester
- Joined IBM as programmer in 1950
- Developed Fortran, first commercially successful programming language and compiler

Describing Languages

- Fortran language was described using English
  - Imprecise
  - Verbose, lots to read
  - Ad hoc
    
    \[
    \text{DO 10 I=1,10} \\
    \text{DO 10 I=1,10} \\
    \] (Often incorrectly blamed for loss of Mariner-I)
- Wanted a more precise way of describing a language

IBM 704 Fortran manual, 1956

<table>
<thead>
<tr>
<th>STATEMENT</th>
<th>NORMAL SEQUENCING</th>
</tr>
</thead>
<tbody>
<tr>
<td>a = b</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>GO TO n</td>
<td>Statement n</td>
</tr>
<tr>
<td>GO TO n, n, ...</td>
<td>Statement last assigned</td>
</tr>
<tr>
<td>ASSIGN i TO n</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>GO TO n, n, ...</td>
<td>Statement ni</td>
</tr>
<tr>
<td>IF (a) n, n, ...</td>
<td>Statement ni as a logical condition</td>
</tr>
<tr>
<td>SENSE LIGHT i</td>
<td>Next executable statement</td>
</tr>
<tr>
<td>IF SENSE LIGHT i</td>
<td>Statement as Sense</td>
</tr>
<tr>
<td>IF SENSE SWITCH i</td>
<td>Statement as Sense</td>
</tr>
</tbody>
</table>

Recall: Backus Naur Form

symbol ::= replacement

We can replace symbol with replacement

A ::= B means anywhere you have an A, you can replace it with a B.

nonterminal - symbol that appears on left side of rule

terminals - symbol that never appears on the left side of a rule

Language Elements

When learning a foreign language, which elements are hardest to learn?
- Primitives: lots of them, and hard to learn real meaning
- Means of Combination
  - Complex, but, all natural languages have similar ones [Chomsky]
    
    SOV (45% of all languages) - Sentence ::= Subject Object Verb (Korean)
    SVO (42%) - Sentence ::= Subject Verb Object
    VSO (9%) - Sentence ::= Verb Subject Object (Welsh)
    
    "Lladdodd y ddraig y dyn." (Killed the dragon the man.)
    
    OSV (<1%): Tobati (New Guinea)
    Schemish: Expression ::= (Verb Object)
- Means of Abstraction: few of these, but tricky to learn differences across languages
  
  English: I, we
  Tok Pisin (Papua New Guinea): mi (I), mitupela (he/she and I), mitripela (both of them and I), mipela (all of them and I), yumitupela (you and I), yumitripela (both of you and I), yumipela (all of you and I)
### Primitives
- Standard Procedures
- Primitive expressions
- Identifiers, numerals

### Means of Combination
- Expressions
- Program structure

### Means of Abstraction
- Definitions

#### Pages in Revised 5 Report on the Algorithmic Language Scheme
- 48 pages total (includes formal specification and examples)

- 18

#### Pages in Revised 5 Report on the Algorithmic Language Scheme
- 18

- 356

#### English
- Morphemes
  - Words in Oxford English Dictionary
    - 500,000
- Grammar Rules
  - English Grammar for Dummies Book
    - 100s (?)
  - 384 pages
- Pronouns
  - ~20

#### Liberal Arts Trivia: Architecture
- Q. Name the distinctive architectural features of Islamic mosques that is typically a spire or onion-shaped dome, and is usually either free-standing or much taller than all surrounding structures. There are six in this picture of the Sultan Ahmed Mosque (Blue Mosque) in Istanbul:

![Image of the Sultan Ahmed Mosque (Blue Mosque) in Istanbul](image-url)
Liberal Arts Trivia: Egyptology

• Q. Name the last effective pharaoh of Egypt’s Ptolemaic dynasty. She originally shared power with her father and her brothers, whom she also married, but eventually ruled alone. As pharaoh, she allied with Gaius Julius Caesar that solidified her grip on the throne. After Caesar’s assassination in 44 BC, she aligned with Mark Antony in opposition to Caesar’s legal heir Augustus. After losing the Battle of Actium to Octavian’s forces, Antony committed suicide, and she followed suit, according to tradition killing herself by means of an asp bite on August 12, 30 BC.

Liberal Arts Trivia: Philosophy

• Q. Name this 19th century philosophical work by John Stuart Mill. To the Victorian readers of the time it was radical, advocating moral and economic freedom of individuals from the state. Mill argues against the “tyranny of the majority” and articulates the harm principle: people can do anything they like as long as it does not harm others.

Expressions and Values

• (Almost) every expression has a value
  - Have you seen any expressions that don’t have values?
• When an expression with a value is evaluated, its value is produced

Five Types of Expression

1. Primitives (if #t 3 5)
2. Names (define (square n) (* n n))
3. Application (square 4)
4. Lambda ((lambda (q) (- 0 q)) 7)
5. If (+ (if true 3 5) 10)

Primitive Expressions

Expression ::= PrimitiveExpression

PrimitiveExpression ::= Number

PrimitiveExpression ::= #t | #f

PrimitiveExpression ::= Primitive Procedure
Evaluation Rule 1: Primitives

If the expression is a primitive, it evaluates to its pre-defined value.

> 3
3
> #t
#t
> +
#<primitive:+>

Name Expressions

Expression ::= NameExpression
NameExpression ::= Name

Evaluation Rule 2: Names

If the expression is a name, it evaluates to the value associated with that name.

> (define two 2)
> two
2

Application Expressions

Expression ::= Application Expression
ApplicationExpression ::= (Expression MoreExpressions)
MoreExpressions ::= ε
MoreExpressions ::= Expression MoreExpressions

Evaluation Rule 3: Application

3. If the expression is an application:
   a) Evaluate all the subexpressions (in any order)
   b) Apply the value of the first subexpression to the values of all the other subexpressions.

   (Expression₀ Expression₁ Expression₂ … )

Rules for Application

I. Primitives. If the procedure to apply is a primitive, just do it.

II. Constructed Procedures. If the procedure is a constructed procedure, evaluate the body of the procedure with each formal parameter replaced by the corresponding actual argument expression value.
Eval and Apply are defined in terms of each other.

Without Eval, there would be no Apply, Without Apply there would be no Eval!

**Making Procedures**

`lambda` means “make a procedure”

Expression ::= ProcedureExpression
ProcedureExpression ::= (lambda (Parameters) Expression)
Parameters ::= ε
Parameters ::= Name Parameters

**Evaluation Rule 4: Lambda**

4. Lambda expressions evaluate to a procedure that takes the given parameters and has the expression as its body.

Lambda is the English name for the Greek letter written \( \lambda \).

**Lambda Example: Tautology Function**

(lambda () #t)
> ((lambda () #t) 1120)
#<procedure>: expects no arguments, given 1: 1120
> ((lambda () #t))
#t
> ((lambda (x) x) 1120)
1120

Now You Know All of Scheme!

- Once you understand Eval and Apply, you can understand all Scheme programs!
- Except:
  - There are a few more special forms (like if)
  - We have not define the evaluation rules precisely enough to unambiguously understand all programs (e.g., what does “value associated with a name” mean?)
**Example: Nanostick**

- How far does light travel in 1 nanosecond?
  
  ```lisp
  > (define nanosecond (/ 1 (* 1000 1000 1000))) ;; 1 billionth of a s
  > (define lightspeed 299792458) ; m / s
  > (* lightspeed nanosecond)
  149896229/500000000
  > (exact->inexact (* lightspeed nanosecond))
  0.299792458
  = just under 1 foot
  
  Some Dell machines in Thornton have “1.8-GHz Pentium 4 CPU”s.
  GHZ = GigaHertz = 1 Billion times per second
  They must finish a step before light travels 6.6 inches!

**Homework**

- Read Structured Lab Guide (Today)
- Complete the Honor Pledge (due Wednesday in class, signed)
- Start PS 1 (due Mon Sep 06)