The notion of algorithm

General definition:

An algorithm is the specification of a process to be carried out by a computer
Not quite an algorithm

5 properties of an algorithm

Algorithm vs program
What makes up an algorithm

Basic steps:
- Feature call \( x.f(a) \)
- Assignment
- ...

Sequencing of these basic steps:

CONTROL STRUCTURES

Control structures

Definition: program construct that describes the scheduling of basic actions

Three fundamental control structures:
- Sequence
- Loop
- Conditional

They are the "Control structures of Structured Programming"

Control structures as problem-solving techniques

Sequence: "To achieve \( C \) from \( A \), first achieve an intermediate goal \( B \) from \( A \), then achieve \( C \) from \( B \)"

Loop: solve the problem on successive approximations of its input set

Conditional: solve the problem separately on two or more subsets of its input set
The sequence (or Compound)

\[
\text{instruction}_1 \\
\text{instruction}_2 \\
\vdots \\
\text{instruction}_n
\]

Semicolon as optional separator

\[
\text{instruction}_1 ; \\
\text{instruction}_2 ; \\
\vdots \\
\text{instruction}_n
\]

Not quite an algorithm

![Image of a recipe label with cooking times]
Correctness of a Compound

Precondition of instruction$_i$ must hold initially

Postcondition of each instruction$_i$ must imply precondition of each instruction$_{i+1}$

Final effect is postcondition of instruction$_n$

Conditional instruction

if Condition
then Instructions
else Other_instructions
end

Computing the greater of two numbers

if $a > b$
then $\text{max} := a$
else $\text{max} := b$
end
As a function

```plaintext
maximum(a, b: INTEGER): INTEGER
  do
    if a > b then
      Result := a
    else
      Result := b
    end
  end
```

The conditional as problem-solving technique

PROBLEM SPACE

Region 1
Region 2

Use technique 1
Use technique 2

Basic form

```plaintext
if Condition then
  Instructions
else
  Other_instructions
end
```
A variant of the conditional

\[
\text{if } \text{Condition} \text{ then} \\
\text{Instructions} \\
\text{end}
\]

Nesting

\[
\text{if } \text{Condition}_1 \text{ then} \\
\text{Instructions}_1 \\
\text{else} \\
\quad \text{if } \text{Condition}_2 \text{ then} \\
\quad \text{Instructions}_2 \\
\quad \text{else} \\
\quad \\
\quad \text{...} \\
\quad \text{end} \\
\text{end} \\
\]

(Means the same as

\[
\text{if } \text{Condition} \text{ then} \\
\text{Instructions} \\
\text{else} \\
\text{Instructions} \\
\text{end}
\]

with an empty "else" clause)
Intro. to Programming, lecture 9: Control structures I

**Nested structure**

**Comb-like structure**

**Comb-like conditional**

```
if Condition_1 then
  Instructions_1
elsif Condition_2 then
  Instructions_2
elsif Condition_3 then
  Instructions_3
elsif ...
else
  Instructions_0
end
```
Comb-like structure

What we have seen

- The notion of algorithm
  - Basic properties
  - Difference with "program"
- The notion of control structure
- Correctness of an instruction
- Control structure: sequence
- Control structure: conditional
- Nesting, and how to avoid it
End of lecture 9