

Control Flow

To make computation feasible, flow of control among statements required.

- Selecting among alternative control flow paths
- Repeated execution of certain collection of statements

Different types of control statements :

- Compound statement
- Selection statement
- Iterative statement
- Unconditional branching
- Guarded commands

Compound Statement

Way to specify a collection of statements; can be abstracted to a single statement

Example :

```
begin  
  statement_1  
  ....  
  statement_n  
end
```

- Missing in early version of FORTRAN
- ALGOL 60 first introduced it
- Data declarations can be added to the beginning, forming a *block*; e.g. C, C++, Ada.
- Pascal no blocks, but allows compound statement

Selection Statement

Way of choosing between two or more execution paths in a program

Categories :

- Two-way (or n -way)
- Multiple selection

Two-way (**if-stmt**)

- Form and type of expression that controls the selection
- Select a single, a sequence of or a compound statement ?
- Nesting selectors

Examples of Two-way Selectors

Degenerate case is FORTRAN's single-way selector or logical if-stmt.

```
if (Boolean expression) statement
```

ALGOL 60 allowed single-way selector with a compound statement

```
if (Boolean expression) then
  begin
    statement_1
    ....
    statement_n
  end
```

ALGOL 60 introduced the first two-way selector

```
if (Boolean expression)
  then statement
  else statement
```

Any of the statements can be a compound statement. Statement following the **then** is called the *then clause*

Statement following the **else** is called the *else clause*

Nesting Selectors

Two-way selection constructs can be nested. This causes the famous the *dangling-if* problem

```
if (sum = 0) then
  if (count = 0)
    then result := 0
  else result := 1
```

Solution :

- Use static semantic rules (Pascal); **else** clause is always paired with the most recent unpaired **then** clause.
- Use syntax to connect **else** clauses to **then** clauses (ALGOL 60)

```
if sum = 0 then
  begin
    if (count = 0)
      then result := 0
    else result := 1
  end
```

Another solution : Use reserved words for end of clauses; Ada, FORTRAN 77, Modula-2

Multiple Selection Constructs

This allows the selection of one of any number of statements or statement groups.

Early multiple selectors :

- FORTRAN's 3-way selector; arithmetic **if-stmt**.
- FORTRAN computed **GO TO**

Modern multiple selectors : **case-stmt**

- Introduced in ALGOL-W; Can select a single branch.
- Pascal's case statement is similar to ALGOL-W's, except the branches are labeled (constant lists).
- C has the **switch** statement. There is no implicit branch after each selectable code segment.
- In Ada's case statements the constant lists have to be exhaustive. Also allows subranges in the constant list. Can use **others** clause to make list exhaustive.

Iterative Statements

This construct causes a statement or collection of statements to be executed 0, 1 or more times.

Different types of loops :

- Counter-Controlled
- Logically Controlled
- User-Located loop Control
- User-defined iteration control

Counter-Controlled :

- Loop parameters; initial value, terminal value and stepsize
- FORTRAN IV do-loop :- posttest loop construct; loop variable undefined upon loop termination; loop variable and loop parameters cannot be changed inside loop body; extended loop body.
- FORTRAN 77 do-loop :- loop variable can be integer, real and double precision; loop parameters can be positive or negative; loop controlled by iteration count, so changing loop parameters inside body does not make any difference; pretest loop construct; single-entry structure.

- ALGOL 60 for-statement :- Excessively complex; can combine counter and boolean expression for loop control; expression in the **for** lists are evaluated every iteration of the loop statement; pretest loop construct.
- Ada for-statement :- Similar to Pascal; scope of loop variable is the range of the loop and it is implicitly declared; Loop variable cannot be changed inside body; Loop parameters can be changed but does not effect iteration count.
- C, C++ for-statement :- pretest loop; all variables can be changed inside loop; it is legal to branch into the body of a for statement; In C++, counter can be defined inside loop control.