Sisyphus
Frank Williams (b. 1947), USA
2003. Mixed media on canvas. 143x195cm.

*In Greek myth, Sisyphus was condemned to roll a boulder up hill over and over again.*
Control Flow

- Sequencing, selection, case/switch statement
- Iteration and recursion
- Iterators, generators in Icon
- Nondeterminacy
Control flow. A computer program directs the computer to execute a stream of instructions from among many possible sequences of instructions. This directed, continuous stream of actions is the control flow. Normally, each statement—left to right, top to bottom—in the program is executed one after the other. This is sequential flow of control.
Unstructured

Machine like:

    if (A .lt. B) goto 10
10       ...

EDSGER W. DIJKSTRA

Fundamental contributions to programming as a high, intellectual challenge.

A.M. TURING AWARD 1972
Dijkstra was born in Rotterdam, Netherlands in 1930. ACM Turing Award in 1972, the 1982 IEEE Computer Pioneer Award. He held the Schlumberger Centennial Chair in Computing Sciences at the University of Texas at Austin, 1984-1999, and retired as Professor Emeritus in 1999. Died 6 August 2002 at his home in Nuenen, the Netherlands.

Dijkstra, “Go to statement considered harmful,” 1968. Ignited the goto controversy


(I have always heard it pronounced “dikeistra,” and I have been told this is close to the Dutch pronunciation.)
Eliminating GOTO


“At the IFIP Congress in 1971 I had the pleasure of meeting Dr. Eichi Goto of Japan, who cheerfully complained that he was always being eliminated.”
10 J = 1
11 COME FROM 20
12 WRITE (6,40) J
   STOP
13 COME FROM 10
20 J = J+2
40 FORMAT (I4)
This was picked up in INTERCAL

(1) PLEASE

.  
.  
.  

(2) DO COME FROM (1)

which has included other dubious features including output in Roman Numerals and the “ignore” variables.
Structured Programming

**if, while**
Single entry point and single exit point. Note also the case/switch statment. Also exception handling does have an impact on the flow of control; we cover this topic later.
Structured Programming

Even the most convoluted control rarely calls for a goto.

```plaintext
loop              for (;;) {
    ...            ...
    exit;         break;
    ...            ...
end loop; }  ```
Midtest Loops

Modula (the precursor to Modula-2) introduced a midtest loop with one or more exit points as a supplement to the pretest (while) and post-test (repeat-until) loops.

```
LOOP
  ⟨statement list⟩₀
WHEN ⟨condition⟩₁ EXIT;
  ⟨statement list⟩₁
WHEN ⟨condition⟩₂ EXIT;
  ⟨statement list⟩₂
:
END;
```

In Modula-2 this gave way to the less structured, but more convenient EXIT statement. A loop may have several exit statements.

The exit statements may be nested in other control constructs (WHILE, CASE, IF, etc.), but they break out of only the most deeply nested LOOP statement.

```
LOOP
  ⟨statement list⟩₀
IF ⟨condition⟩₁ break l;
  ⟨statement list⟩₀
END;
```

Ada has an exit-when statement that breaks out of any nested loop.

```
l: loop
  ⟨statement list⟩₀
exit l when ⟨condition⟩₁;
  ⟨statement list⟩₀
end loop l;
```

Java is much the same but has no special loop construct; one must use the while or the for loop.

```
l: for (;;) {
  ⟨statement list⟩₀
if ⟨condition⟩₁ break l;
  ⟨statement list⟩₀
}
```
Without a label break (and continue) in Java refer to the most closely enclosing for, while, do, or switch statement.
Structured Programming

ada/programs/goto/main.adb
ada/programs/goto/alt.adb
ada/programs/goto/fun.adb
(Recall this definition.) *Statement*. A **statement** is a construct in a program that performs some action or governs the control flow. *Block*. A **block** is a group of statements usually with a single entrance and exit. The block is like parentheses in expressions, only it is used in control flow. But blocks may also control scope of declarations (as may other compound statements).
Iteration

Modula-2

FOR i := first TO last BY step DO
  ...
END

Most languages, including Algol 68, Pascal, Ada, Fortran 77 and 90/95, and Modula-3 prohibit changes to the loop index within the body of the loop. In Ada, step is 1 or -1.

Tip: Use a for loop when you know the number times a block of code needs to be repeated.
Iterators

Sebesta, “Section 8.4.4 Iteration Based on Data Structures,” pages 336–337. A Perl example:

```perl
@names = {"Bob", "Carol", "Ted", "Alice"}
foreach $name (@names) {
    print $name;
}
```

A Python example of iterating over a list:

```python
names = ["Bob", "Carol", "Ted", "Alice"]
for name in names:
    print name
```
Iterators

Clu examples, Scott, Figure 6.5, page 288 and Figure 6.6, page 289.
Iterators

Idiom for stepping through all the key-value pairs in a Java Map collection:

```java
for (Iterator i=m.entrySet().iterator();i.hasNext();)
{
    final Map.Entry e = (Map.Entry) i.next();
    System.out.println(e.getKey()+" ": "+e.getValue());
}
```

With the enhanced for loop introduced with in the revised Java language introduced in 2004:

```java
for (Map.Entry e: m.entrySet()) {
    System.out.println(e.getKey()+" ": "+e.getValue());
}
```
Icon Generators

... to ... by ...
find (substr,str)
upto (chars,str) # all positions at which any char in chars

The prefix operator ! generates all elements of a string, list, record, file, or table.

every i := first to last by step do { write(i) }
every i:=1 + upto(’ ’,s) do { write(i) }
every write (1+upto(’ ’,s))

if 2>6 then { ... }
if (i:=find(‘ab’,s)>6 then { ... }

Scott, Section 6.5.3 Iterators, pages 288-291.
TPK in Icon
def gensquares(N):
    for i in range(N):
        yield i ** 2

for i in gensquares(5):
    print i, "; "
List Comprehension

List Comprehension Python (also in Haskell)

[expression for name in sequence [if conditional] ...]

>>> [x**2 for x in range(10)]

[0, 1, 4, 9, 16, 25, 36, 49, 64, 81]

>>> noprimes = [j for i in range(2,8) for j in range(2*i, 50, i)]

>>> primes = [x for x in range(2,50) if x not in noprimes]

>>> print primes

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47]

composite = [j | i <- [2..8], j <- [2*i, 3*i .. 50]]

primes = [x | x <- [2..50], not (x `elem` composite)]
Recursion

Theory: Don’t need iteration, recursion is enough.
Practice: Can be as efficient as iteration *tail recursion*. (Sebesta, 9th ed., Section 15.4, page 660.) Optimizing transformations like *continuation-passing style* is an advanced topic in functional programming.

*We really did not understand the implications of recursion, or its value, at the ALGOL 60 meeting in Paris. McCarthy did, but the rest of us didn’t.*

Perlis, 1978
Nondeterminacy

Compute the maximum:

```plaintext
if x ≥ y → m := x
  y ≥ x → m := y
fi.
```

Compute the gcd:

```plaintext
x := X; y := Y;
do x > y → x := x − y
  y > x → y := y − x
od.
```

Used in Hoare’s CSP and for concurrency in Occam, Ada and SR.
Parallelism

FORALL a HPF (High-Performance FORTRAN) extension to Fortran 90 found in the Fortran 95 standard.

```
FORALL (i=1:size, j=i:size)
    hilbert(i,j) = 1.0 / real (i+j+1)
END FORALL
```

Significance of Nondeterminacy

Usually programming languages prescribe exactly the order of execution for all statements even though the algorithm may not depend on the order.

\[ X := 3; \quad Y := 7; \quad // \text{one order} \]
\[ Y := 7; \quad X := 3; \quad // \text{another} \]

Computer hardware today is highly parallel and it requires great effort on the part of compiler to find some of the parallelism hidden in the unnecessarily, over-constrained program sequence.
Exception handling, see Sebesta, seventh edition, Chapter 11. Call-backs, Java GUI event handling, see Sebesta, seventh edition, Chapter 11. Where is the main? Applets (run by the browser), serverlets (run by the server), GUI programs (reactive to user gestures).
Scott 6.9, Review Questions