

Class 28: Goto Considered Harmful

Held: Monday, 11 April 2011

Summary: Today we consider the basics of structured programming and the reasons it supplanted goto-ridden programming.

Related Pages:

- EBoard.
- Reading: Dijkstra: Go To Statement Considered Harmful; Bratley, P. (1967). Algorithm 306: Permutation with Repetitions [G6]. *Communications of the ACM* 10(7), July 1967, pp. 450-451..

Notes:

- Are there questions on the exam?
- Reading for Wednesday: ???
- EC for either of Tuesday's Town-Hall Meetings on the Forum [academic]
- EC for Thursday's Convocation on Intellectual Property [academic]
- EC for Thursday's CS Extra [academic]
- EC for Friday's CS Table [academic]

Overview:

- Context.
- Dijkstra's Famous Letter.
- Permutation with Repetitions.

The Context of GTSCH

Some of these notes come from Dijkstra's "What Led to 'Notes on Structured Programming'" (EWD1308). Available at <http://www.cs.utexas.edu/users/EWD/transcriptions/EWD13xx/EWD1308.html>.

- Late 1960's; about a decade after the first major high-level programming languages.
- Why did languages include go to statements?
 - Part of the underlying Von Neumann architecture.
 - At least initially, seemed like the obvious control structure. (Non-recursive procedure call also available.)
- If control was based on go to statements, why were these languages "closer to the problem"?
 - Programmers didn't have to lay out memory for values and variables.
 - Programmers didn't have to decompose formulae into individual operations.
 - Programmers didn't have to worry about temporaries in such decompositions.
 - Mnemonics for many key operations.

- Programmers didn't have to worry about data representation.
- ...
- But people were seeing problems writing larger programs
 - As the previous reading tells us, many proto-software-engineers wanted more abstraction
 - A large community also wanted to be able to formally verify programs.
- The result: New control structures
 - Conditionals of various forms (including case)
 - Recursive functions
 - Loops
- Key idea of most of these control structures (usuauully implicit; eventually explicit): One entry point, one exit point.
- Dijkstra, a few years earlier, had been working on parallel programming.
- Dijkstra, in some sense, was serving as (unsigned) spokesperson for a much larger group.
- The original title of the article was "A Case Against Go to"

The Letter: Go To Statement Considered Harmful

- Thesis?
- Structure of argument?
- What does he mean by "a (mixed) sequence of textual and/or dynamic indices"?
- What does he mean by "coordinate"?
- Why would one care about indices or coordinates?
- What does he mean by the "number of people in the room" example?
- Possible counter arguments?

Algorithm 306: The Interface

- What is the purpose of this algorithm?
- What distinguishes it from other algorithms for computing permutations?
- Are there other goals of the algorithm?
- Does it meet those goals?

Algorithm 306: The Implementation

- What are the arguments, the local variables, the "own" variables?
- What general purposes do they seem to serve?
- What's going on at L2?
- Why isn't the sequence of L4 and L5 represented as a loop?

```

L4: if a[i] = p then go to L2;
    if a[i] < p then r := i;
L5: i := i-1;
    go to L4;
L2: ...

```