Class 03: Parameter-Passing Strategies

Held: Wednesday, 31 August 2011

Summary: We consider parameter-passing strategies in programming languages.

Related Pages:

- EBoard.
- Reading: Aho et al., Chapter 1.

Notes:

- I checked, and Pascal is case insensitive.
- Reading for Friday: Chapter 2. (Yes, it’s a lot. It’s okay to skim.)
- You can find most of today’s Pascal code examples in the Examples/Pascal page or the Examples/Wirth page.
- I realize that I have overprepared for today’s class. Some topics will probably move to Friday’s class.
- EC for Thursday’s Thursday extra on HCI for Freedom.

Overview:

- Parameter Passing.
- Procedures in Pascal.
- Parameter Passing, Revisited.
- Questions on the Reading.

Parameter Passing Policies

- Terminology:
  - “formal parameters” (also “formals”): the parameters that appear in the function/procedure declaration.
  - “actual parameters” (also “actuals” also “actual arguments” also “arguments”): the parameters that appear in the function/procedure call
  - “parameter”: ambiguous
- Consider declaration

```plaintext
int fun(int f) {
    ...
}
```

- And call
What is the relationship between formal $f$ and actual $a$?

- **Pass-by-value**: Each of $f$ and $a$ have their own memory location. The value stored in $a$ is copied into the location associated with $f$. Nothing gets copied back.
- **Pass-by-reference**: $f$ and $a$ share a memory location. Changes to $f$ affect $a$.
- **Pass-by-text**: The text of $a$ is substituted for $f$ in the body of `fun`.
- **Pass-by-name**: Call-by-text with sensible scoping rules.

We will focus on pass-by-value and pass-by-reference, which are two of the most common parameter passing strategies.

### Procedures in Pascal

- Pascal differentiates procedures, which do not return values, from functions, which do.
- The form looks much like that of a program:

  ```pascal
  procedure (typed-parameters);
  var
      variables
  local-procedures
  begin
      statement-list
  end
  ```

  One big difference: You can pass parameters by value or by reference.
  - By value: What you know from C. Changes to the formal do not affect the actual.
  - By reference: Changes to the formal do affect the actual.

- **Examples**
  - `swap.p`
  - `munge.p`

- For functions, you must also specify a return type.
- Return values from functions are treated strangely.
  - You must assign to “the function variable” (which has the same name as the function).
  - The value of that variable when the function terminates is the value of the function.
  - But it can only be used on the lhs of an assignment.

- **Examples**
  - `functions.p`

### Parameter Passing Revisited

- Why would we choose one parameter passing method over another?
- Do you agree with the assertion that “Java behaves as if it used call-by-reference for anything other than a basic type”? (p. 34) Why or why not?
Questions on Aho et al., Chapter 1

- Do you have any questions on the first chapter of the text?