

SAS Programming Style

Tyler Cole

SECTIONS

General Concepts

Program Style

SAS Techniques

Summary Points

General Concepts

GENERAL CONCEPTS

Minimalism

- Architecture and design
- Simplest and fewest elements are used to create the maximum effect

GENERAL CONCEPTS

“Minimum”

- Achieved when it is no longer possible to improve by deletion
 - Reduce program code to the fewest steps necessary
 - Keep the number of PROC and DATA steps to a minimum
 - Compactness does not take precedence over clarity

GENERAL CONCEPTS

Programming Goal

- Code that can be repaired “on the fly” by the average programmer

GENERAL CONCEPTS

Basic examples

- | | |
|------------------------|--------------------------------------|
| • Sort and MERGE: | Use SQL join |
| • Numerous DATA steps: | Consolidate and use fewer DATA steps |
| • User-defined macros: | Reorganize code if called only once |

Program Style

PROGRAM STYLE

Typical Layout

One-off programs typically contain three distinct sections of code

- 1) Process records
- 2) Summarize records
- 3) Output results

PROGRAM STYLE

process

summarize

output results

PROGRAM STYLE

EXAMPLE: Macro with %IF-%THEN and other macro statements

%macro...

%if-%then

process

summarize

output results

%mend...

%...

PROGRAM STYLE

EXAMPLE: Macro called only once (can move SAS code outside macro)

`%macro...`

process

summarize

output results

`%mend...`

`%...`

PROGRAM STYLE

EXAMPLE: Macro called multiple times

`%macro...`

process

summarize

output results

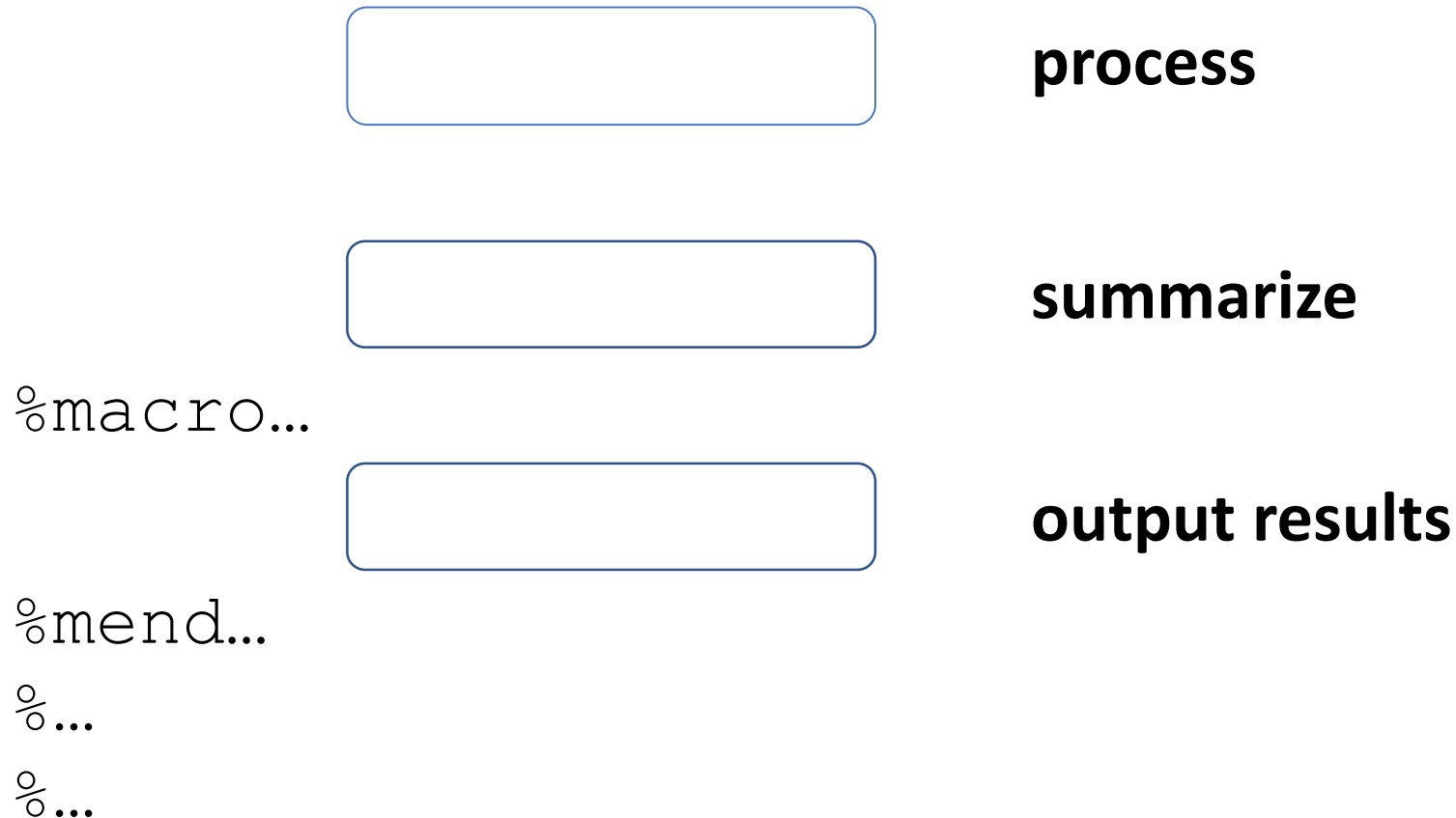
`%mend...`

`%...`

`%...`

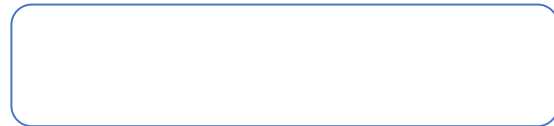
PROGRAM STYLE

EXAMPLE: Use group-by processing to minimize code repeated in macro



PROGRAM STYLE

Sample program

An empty rectangular box with a thin blue border and rounded corners, intended for a sample program.

**process and create final.sas7bdat
(one record per PT per...)**

An empty rectangular box with a thin blue border and rounded corners, intended for a sample program.

**summarize and subset
final.sas7bdat**

An empty rectangular box with a thin blue border and rounded corners, intended for a sample program.

output results

PROGRAM STYLE

- Updates much easier if process/summarize/output are kept separate
- Use group-by processing to compute statistics in one pass
- Limit macro only to SAS code that must be repeated

SAS Techniques

SAS TECHNIQUES

IF-THEN statement

`IF [junk] THEN [statement if true];`

<code>[junk]</code> evaluates to missing	→ FALSE
<code>[junk]</code> evaluates to zero	→ FALSE
All other values	→ TRUE

SAS TECHNIQUES

IF-THEN statement

```
if nobs then...;  
if indexc(valc, '<') then...;
```

SAS TECHNIQUES

Align IF-THEN/ELSE statements

```
if pos='RIGHT' then val=scrsp1_std;  
else if pos='LEFT' then val=scrps2_std;
```

```
if          pos='RIGHT' then val=scrsp1_std;  
else if pos='LEFT'      then val=scrps2_std;  
                                ↑ ↑
```

SAS TECHNIQUES

Implicit array

```
array tmp $ note1 - note3  
do over tmp;  
    if not missing(tmp) then...;  
end;
```

SAS TECHNIQUES

PUT function

``*' || put(1.5, best.) || '*';` → * 1.5 *

``*' || put(1.5, best. -1) || '*';` → *1.5 *

``*' || put(1.5, best. -c) || '*';` → * 1.5 *

`OPTIONS MISSING='.';`

``*' || put(., best.) || '*';` → * . *

SAS TECHNIQUES

Distinct count

```
proc sql;  
select count(distinct pt) from adverse;  
quit;
```

Summary Points

SUMMARY POINTS

- MINIMUM: No longer possible to strengthen by deletion
- Reduce program code to the minimum steps necessary
- Don't let compactness take precedence over clarity
- Maintain distinct sections for process, summarize, and output results