PhUSE GPP Steering Board
Good Programming Practice at a Glance

The GPP Steering Board is a voluntary industry group with representatives from a diverse array of health and life sciences organizations:

- Working to develop common good programming practices applicable to statistical programming for the development of high quality and efficient programming code.
- Focus on analysis and reporting of clinical trial data: data integration; preparation of clinical data for e-submissions.
- This poster contains an overview of PhUSE GPP guidance Version 1: Consolidated industry guideline with potential to replace/extend current industry guidance and serve as a set of business rules when working on standard scripts for public use.

PhUSE GPP Guidance is found on PhUSE Wiki


---

**Before you begin**

Understand what you need to do; plan how to get there; collect what you need:
- SOPs & training
- Study Protocol
- Standards
- Annotated CRF
- SAP & Shells
- Specifications
- Tools and macros

---

**GPP Principles**

Write code that is:
- Clear and easy to read and review
- Easy to maintain, modify and debug
- Robust and needs minimal code maintenance
- Can be reused or easily adapted.
- Efficient and uses minimal computer processing resource
- Written in such a way to reduce logical and syntax errors

---

**Coding conventions**

- Do not overwrite existing datasets
- Use of all uppercase should be avoided
- Separate data steps and procedures with at least one blank line – Use whitespace
- Use ‘data=dataset’ option in procedure statements
- End data steps and procedures with run or quit
- Put each statement on a separate line
- Indent statements belonging to a level by 2 to 5 columns (use the same number of spaces throughout the program), i.e. every nesting level should be visibly indented from the previous level.
- Do loops and if then else statements start and end with same level of indentation
- Comments explain the purpose of what is being done
- Indented Code
- Keep only the variables you need
- Data sets not overwritten
- Where clauses do not overwrite
- What else?

---

**Examples of Good and bad code**

Review the 2 code snippets below: Both produce the same output. Why is code on the right better?

---

**Layout & Header**

1. Always have a program header!
2. Clean up work area from previous program
3. Read in external data
4. For multiple data sources, pre-process, then combine before common derivations are applied
5. Process and manipulate data, merge and derive variables
6. Perform analysis
7. Report data: Produce tables, figures and listings
8. Clean up work area

---

**DO**

- Use a program Header
- Use informative comments in appropriate places
- Use unique and informative dataset names
- Use one statement per line
- Indent consistently
- Program defensively for missing and unexpected data
- Check log for ERRORS, WARNING, and NOTES that indicate data issues

**DON’T**

- Use a header but forget to fill
- Use comments that simply say what the code does (not why)
- Use inconsistent or random naming for datasets and variables
- Program around data issues instead of fixing them
- Ignore Notes in the Log that indicate potential issues E.G. Automatic Number/character conversions, Merge by multiple records

---

**Check your Log!**

Note(s): just ERRORS and WARNING s!

---

Many thanks to the GPP Steering Board for their constructive reviews and contributions.

Mark Foxwell, PRA Health Sciences UK

Beate Hientzsch Accovion, Germany

Shafi Chowdhury Shafi Consultancy, UK

Dean Grundy Roche, UK

Jennifer Chin Eisai, UK

Cindy Song Sanofi, US

Kate Peaceck Quintiles, UK

Alexandra Marquart Boehringer-Ingelheim Germany

Marina Dalton GSK, US

Gayathri Janssen R&D (J&J), US

Kolandavelu Jane Merrer, US

Ninan Luke Novartis, India

Art Collins Biogen Idec, US

Ralf Gessensch Vollmar, Germany

Mark Foxwell, PRA Health Sciences, Reading, UK