Chapter 1

Environment and Guiding Principles

This chapter provides the context and universal guidelines for the material in this book. It is best to begin by describing the environment in which a statistical programmer works in the pharmaceutical industry. Then we explore the fundamental principles that should guide you in your day-to-day work. These principles permeate all of the tasks that you do on a daily basis and, if kept in mind, they will keep you from going astray in your statistical programming duties.

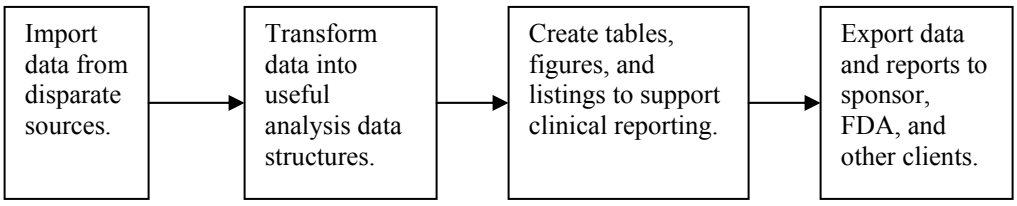
The Statistical Programmer's Working Environment

Pharmaceutical Industry Vocabulary

Like many industries, the pharmaceutical industry has a vocabulary and language all its own. Our industry is full of acronyms, medical terminology, and jargon that you must become familiar with to be an effective statistical programmer. To assist you in identifying some of these terms, this book *italicizes* the first occurrence of terminology specific to SAS programming in the pharmaceutical industry. At the end of the book is a glossary that you can refer to for definitions of these terms.

Statistical Programmer Work Description

The statistical programmer usually works in the statistics department of a pharmaceutical research and development group or *contract research organization* (*CRO*). The role of statistical programmers is to use their superior technical and programming skills in order to allow *clinical trial* statisticians to perform their statistical analysis duties more efficiently. This may involve importing and exporting data, working with other information technology professionals on site and at other companies, deriving variables and creating analysis data sets, and creating *clinical study report* (*CSR*) materials consisting of tables, figures, and listings (*TFL*). Here is a simplified illustration of the general work processes of the statistical programmer:



The Drug/Device Development Process

The clinical trial industry is primarily concerned with bringing new drugs, devices, or therapies to the general population. In the United States, most clinical trials are funded by pharmaceutical companies wanting to bring a new treatment to market or by the National Institutes of Health (NIH), which funds research to improve the health of all Americans. Because the majority of clinical trials are conducted with the idea to bring a new drug or device to market, we will briefly look at the U.S. *Food and Drug Administration* (*FDA*) approval process. Further details about the drug and device approval process can be found at www.fda.gov.

Drug Approval Process

The FDA is charged with making sure that all new drugs brought to market are both safe and effective. The FDA helps to do this with a drug approval process that can easily cost hundreds of millions of dollars and can take a decade or more to move a drug from discovery to a pharmacy near you. There are several progressive levels of studies that are conducted as part of the drug approval process.

1. **Pre-clinical studies** are the experiments that are conducted in the laboratory and with animals long before a new drug is ever introduced for use by humans. If these studies are promising, the drug maker usually pursues an *Investigational New Drug* (*IND*) application. The IND application allows the drug maker to conduct clinical trials of the new compound on human subjects.
2. **Phase 1 trials** are the "first in man" studies of a new drug in humans. These studies are usually carried out on small samples of subjects. The idea here is to determine the safety of the drug in a small and usually healthy volunteer study population.
3. **Phase 2 trials** go beyond phase 1 studies in that they begin to explore the efficacy of a drug. Phase 2 studies have larger (100–200 patients) study populations than phase 1 studies and are aimed at narrowing the dose range for the new medication. Safety is monitored at this stage as well, and phase 2 trials are generally conducted in the target study population.
4. **Phase 3 trials** are large-scale clinical trials on populations numbering in the hundreds to thousands of patients. These are the critical trials that the drug maker runs to show that its new drug is both safe and efficacious in the target study population. If the phase 3 trials are successful, they will form the keystone elements of a *New Drug Application* (*NDA*).
5. **Phase 4 trials**, or post-marketing trials, are usually conducted to monitor the long-term safety of a new drug after the drug is already available to consumers.

Device Approval Process

The FDA is also in charge of regulating new medical devices. The device approval process at the FDA varies based on the degree of risk inherent in the device. Class 1 devices carry little risk for the patient; they include devices such as elastic bandages and surgical instruments. Class 2 devices carry slightly higher risk for the patient; they include such devices as infusion pumps and motorized wheelchairs. Class 3 devices are high-risk devices and thus require the most regulatory scrutiny. Class 3 devices include replacement heart valves and implantable defibrillators. Obviously, the approval requirements for a class 3 device are much higher than for a class 1 device.

Clinical Trial Study Designs

There are many types of clinical trials, and there are some general trial design concepts that you need to understand. One key concept is the *randomization* of study therapy. When you randomly assign patients to study therapy, you reduce potential treatment *bias*. Another key concept is treatment *blinding*. Blinding a patient to treatment means the patient does not know what treatment is being administered. In a *single-blind trial*, only the patient does not know what treatment is being administered. In a *double-blind trial*, neither the patient nor the patient's doctor knows which treatment is being given. On occasion there may even be a *triple-blind trial*, where the patient, the patient's doctor, and the staff analyzing the trial data do not know what therapy is being given.

There are other trial design concepts for you to be aware of. A clinical trial can be carried out at a single *site* or it can be a *multi-center trial*. In a single-site trial all of the patients are seen at the same clinical site, and in a multi-center trial several clinical sites are used. Multi-center trials are needed sometimes to eliminate site-specific bias or because there are more patients required than a single site can *enroll*.

Trials may be designed to determine equivalence or superiority between therapies. An *equivalence trial* is designed to show that there is no clinically significant difference between therapies, and a *superiority trial* is intended to show that one therapy is significantly better than another.

Finally, trials can follow parallel or crossover study designs. In a *parallel trial,* patients are assigned to a therapy that they remain on, and they are compared with patients in alternate therapy groups. In a *crossover trial*, patients switch or change therapy assignments during the course of the trial.

Industry Regulations and Standards

Regulatory authorities govern and direct much of the work of the statistical programmer in the pharmaceutical industry. It is important for you to know about the following regulations, guidance, and standards organizations.

International Conference on Harmonization (ICH)

The *International Conference on Harmonization* (*ICH*) is a non-profit group that works with the pharmaceutical regulatory authorities in the United States, Europe, and Japan to develop common regulatory guidance for all three. The goal of the ICH is to define a common set of regulations so that a pharmaceutical regulatory application in one country can also be used in another. Over time the Food and Drug Administration (FDA) usually adopts the guidelines developed by the ICH, so you can watch the development of guidance at the ICH to see what FDA requirements may be forthcoming.

Clinical Data Interchange Standards Consortium (CDISC)

The *Clinical Data Interchange Standards Consortium* (*CDISC*) is a non-profit group that defines clinical data standards for the pharmaceutical industry. CDISC has developed numerous data models that you should familiarize yourself with. Four of these models are of particular importance to you:

- *Study Data Tabulation Model* (*SDTM*). The SDTM defines the data tabulation data sets that are to be sent to the FDA as part of a regulatory submission. The FDA has endorsed the SDTM in its *Electronic Common Technical Document* (*eCTD*) guidance. The SDTM was originally designed to simplify the production of *case report tabulations* (*CRTs*), and therefore the SDTM is listing friendly, but not necessarily friendly for creating statistical summaries and analysis.
- *Analysis Dataset Models* (*ADaM*). The CDISC ADaM team defines data set definition guidance for the analysis data structures. These data sets are designed for creating statistical summaries and analysis.
- *Operational Data Model* (*ODM*). The ODM is a powerful *XML*-based data model that allows for XML-based transmission of any data involved in the conduct of clinical trials. SAS has provided support for importing and exporting ODM files via the CDISC procedure and the XML LIBNAME engine.
- *Case Report Tabulation Data Definition Specification (Define.xml).* Define.xml is the upcoming replacement for the data definition file (define.pdf) sent to the FDA with electronic submissions. Define.xml is based on the CDISC ODM model and is intended to provide a machine-readable version of define.pdf. Because define.xml is machine readable, the metadata about the submission data sets can be easily read by computer applications. This allows the FDA to work more easily with the data submitted to it.

You will be exporting, importing, and creating data for these models, so it is important that you learn about them. The FDA has begun to formally endorse the use of these data models in their guidance. Eventually the FDA will probably require data to be formatted to the CDISC model standards for regulatory submissions.

Food and Drug Administration (FDA) Regulation and Guidance

The FDA is the department within the United States Department of Health and Human Services that is charged with ensuring the safety and effectiveness of drugs, *biologics*, and devices marketed in the United States. Any work that you perform that contributes to a submission to the FDA is covered by these federal regulations. There are a number of specific regulations and guidance you must know.

"21 CFR – Part 11 Electronic Records; Electronic Signatures"

21 CFR – Part 11 is a federal law that regulates the submission of electronic records and electronic signatures to the FDA. Of particular interest to the statistical programmer are the following requirements of Part 11:

> "Validation of systems to ensure accuracy, reliability, consistent intended performance, and the ability to discern invalid or altered records."
>
> "Determination that persons who develop, maintain, or use electronic record/electronic signature systems have the education, training, and experience to perform their assigned tasks."
>
> "Adequate controls over the distribution of, access to, and use of documentation for systems operation and maintenance."
>
> "Revision and change control procedures to maintain an audit trail that documents time-sequenced development and modification of systems documentation."

21 CFR – Part 11 means that you must be qualified to do your work, your programming must be validated, you must have system security in place, and you must have change control procedures for your SAS programming. The current additional FDA guidance on 21 CFR – Part 11 is titled "Guidance for Industry Part 11, Electronic Records; Electronic Signatures—Scope and Application."

"E3 Structure and Content of Clinical Study Reports"

The "*E3*" describes in detail what reporting goes into a clinical study report for an FDA submission. This guidance is of major importance, as you are often required to generate tables, figures, case report tabulations, and perhaps *clinical narrative* support for the clinical study report.

"E9 Statistical Principles for Clinical Trials"

The "*E9*" discusses the statistical issues in the design and conduct of a clinical trial. It details trial design, trial conduct, and data analysis and reporting. Although most useful

to the statistician, this guidance gives an excellent overview of how a clinical trial should be conducted.

"E6 Good Clinical Practice: Consolidated Guidance"

The "*E6*" (or *GCP*s) discusses the overall standards for implementing a clinical trial. Anyone who works on a clinical trial needs to understand this document. Of particular interest to the statistical programmer are the following parts of E6. The italics have been added for emphasis.

> "5.1.1 The sponsor is responsible for implementing and maintaining quality assurance and quality control systems with written SOPs [standard operating procedures] to ensure that trials are conducted and data are generated, documented (recorded), *and reported* in compliance with the protocol, GCP, and the applicable regulatory requirement(s)."
>
> "5.5.1 The sponsor should *utilize appropriately qualified individuals* to supervise the overall conduct of the trial, to handle the data, to verify the data, to *conduct the statistical analyses*, and to *prepare the trial reports.*"
>
> "5.5.4 *If data are transformed during processing, it should always be possible to compare the original data and observations with the processed data.*"

"Part 312.33 of Title 21 of the Code of Federal Regulations; Annual Reports"

21 CFR – Part 312.33 discusses what is required for an Investigational New Drug (IND) application. Part 312.33 discusses the requirements for the annual reporting for the IND. This reporting requires you to create adverse event, death, and subject dropout summaries annually for any drug under an IND application.

"Providing Regulatory Submissions in Electronic Format – General Considerations"

This guidance document governs how electronic files should be sent to the FDA. Currently, the FDA requests that electronic documents be submitted as Portable Document Format (PDF) files. The PDF page should be a standard 8.5" × 11" page with 1" margins and 12-point font. Data sets are currently to be sent to the FDA as SAS XPORT transport format files. In the future it is likely that data sets will be required to be sent as XML files, probably formatted in the CDISC ODM.

"Providing Regulatory Submissions in Electronic Format – NDAs"

This guidance document describes how a New Drug Application (NDA) may be sent electronically to the FDA. The guidance defines how the files in the electronic submission should be structured for FDA review.

"Electronic Common Technical Document Specification"

The Electronic Common Technical Document (eCTD) is the vision for future electronic submissions to the FDA. This specification was developed by the International Conference on Harmonization (ICH) as an open-standards solution for electronic submissions to worldwide regulatory authorities. The FDA has adopted the eCTD as the future replacement for its other e-submission guidance, although for now the older guidance is still in effect. Note that the eCTD still depends largely on submitting text documents as PDF files and submitting data sets as SAS XPORT transport format files.

Your Clinical Trial Colleagues

Within any pharmaceutical company or contract research organization, there are groups and individuals outside the statistics department that you work with. Let's take a look at the functional groups a statistical programmer interacts with most.

Site Management

The site management group is responsible for clinical site relations. They recruit doctors at clinics to participate in clinical trials, train their staff in trial conduct, and monitor the sites for protocol compliance while serving as an all-around advocate for the clinical site. Site management can be your ally in helping to get the data entered in a clean and readily usable form. Clean data at the start of the data collection process precludes the need for extra data queries from data management and helps prevent subsequent data analysis problems. With the arrival of *electronic data capture* (*EDC*) technology, the importance of site management has grown, because data entry has moved from the data management group to the clinical site itself.

Data Management

Next to the clinical trial statistician, the statistical programmer works most closely with the data management group. The data management group is usually responsible for *case report form* (*CRF*) design, database design and setup, data entry, data cleaning, data coding, data quality control, and providing the clinical trial data for analysis by the statistics group. Cleaning the data involves scouring the data for problems by using programmatic and manual checks of the data. Coding the data entails applying generic codes to freely entered text fields such as adverse events, medications, and medical histories. Quality control of the data involves auditing the data to make sure that it was entered properly. Finally, the data management group typically provides the data to the statistical programmer via some kind of *relational database management system* (*RDBMS*), which can then be imported into SAS. You save time when data management provides a well-cleaned and well-coded clinical database, because this means you do not have to program around dirty data.

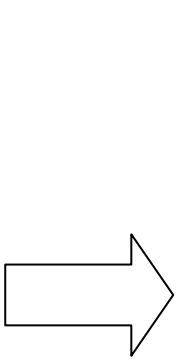
Information Technology

The information technology (*IT*) group has varying responsibilities, depending on the size of your organization. IT is usually responsible for computer systems infrastructure, maintenance, and general computer help desk support. The IT group may also perform some level of software development. In small to midsize organizations IT may simply make *application program interfaces* (*APIs*) between off-the-shelf systems, while at large organizations IT may be responsible for full software applications architecture and development.

You need to work with the IT department within your organization as well as with external sponsors and vendors. Internally, you may work with IT for SAS configuration management and installation qualification, encryption technologies, and desktop publishing or report distribution concerns. The most common reason for you to work with external IT staff is usually in regard to information exchange technologies such as *FTP* and encryption tools.

Project Management

Most contract research organizations and pharmaceutical companies are organized in a matrix management structure. This structure is called a matrix because there are project teams that span various functional departments. It may help to visualize the relationship like this:



		Functional Departments		
		Statistics	Data Management	Site Management
Clinical Trial Projects	Project A			
	Project B			
	Project C			

The project manager provides operational oversight in a clinical trial. The project manager is responsible for meeting the trial needs by enlisting the support of the various functional departments. He or she also works with the primary investigator, as well as with external vendors such as laboratories, pharmaceutical companies, and contract research organizations. The project manager needs to work with the statistical programmer over the course of a clinical trial. As a statistical programmer, you may find that you answer to (at least) two managers during a trial. The statistical programming

functional management serves as your skill-specific manager, while the project manager serves as your project-specific manager.

Quality Assurance

The quality assurance (QA) group is your internal regulatory reference, and they are there to help you. The primary goal of QA is to see that operations in your organization meet regulatory standards. They can assist you in interpreting the various regulations and help you to prepare for customer and regulatory audits. Quality assurance may also perform internal audits to make sure that your business processes meet regulatory standards. Finally, the QA group typically maintains all of your company's standard operating procedures (SOPs).

Medical Writing

The medical writing group may assist in creating various documents for your organization. Medical writers may help with the writing of clinical study reports for the FDA. Medical writers may also get involved in writing an NDA submission. Clinical narrative safety reporting is another task that medical writers help with. On occasion you will have to respond to requests for additional data from the medical writing group as they compile their reporting. Finally, a good medical writer can be a staunch ally in statistical reporting, as he or she may find any last-minute inconsistencies in your analysis before sending it along to the authorities.

Guiding Principles for the Statistical Programmer

The following are specific guiding principles for SAS programming in the pharmaceutical industry. These are high-level concepts that you should keep in mind while performing any of a broad range of tasks.

Understand the Clinical Study

A good statistical programmer takes time to understand the subject matter. If you were going to perform open-heart surgery and you were handy with a knife, you would not just roll up your sleeves and get to work. You would get formal training and obtain a medical degree first so that you understood what you were doing. The same can be said of SAS programming in the pharmaceutical industry. Just because you are a SAS expert does not mean you know all there is to know about a particular drug or device or the disease state it intends to cure.

There are several areas of study that will help you understand the research topic. As a first step you will want to read the clinical *protocol*. The protocol describes the device or medication to be used, the patient populations under study, the statistical plan of the clinical trial, and the details of the disease state. If you want to understand the disease state or indication further, you may want to seek out a clinical *investigator* of the clinical trial or do some further reading about the disease. Understanding the patient population is a good way to understand the data that you will see and whether there is reason for concern when viewing the data. For example, if you were studying a medication to reduce hypertension, you would not be as worried if patient blood pressure data were elevated at *baseline*. You would expect to see this because you understand that hypertensive patients have high blood pressure.

The next step in understanding the topic of study is to read the *statistical analysis plan* (*SAP*). The SAP is a very detailed document, separate from the protocol, describing how the clinical trial data will be analyzed. Although the protocol usually has only a few paragraphs on the statistical analysis, the SAP presents the entire statistical analysis in considerable detail. The SAP describes what inferential analyses will be done, defines the study population, presents data windowing or other special data handling rules, and often includes draft *output shells* that show precisely what tables, listings, and graphs will be provided in the reporting. The SAP is where the majority of your work is defined. Thus, you need to understand the SAP in exquisite detail, so it is beneficial to study it well in advance of programming.

There are additional documents describing the operations of the clinical trial that you may want to review. The site monitoring plan describes how the site management staff ensures that the clinical sites are conducting the protocol and completing the CRF properly. The clinical data management plan used by the data management staff may be useful to review. The clinical data management plan contains data entry instructions, data coding instructions, data review instructions, and a data quality control plan. Finally, there is the very important *annotated CRF,* which shows you where the variables in the clinical database come from on the CRF. The following is an example of an annotated medical history CRF page:

<data set name = MEDHX>

Protocol Name *<study>* Patient Number: _ _ _ - _ _ _ *<patid>* Visit Identifier *<visit>*

Medical History:

	Historical Condition	Diagnosis Date
1	_____	__/__/____
2	_____	__/__/____
3	_____	__/__/____
4	_____	__/__/____

<seq> ***<condition>*** ***<diagdt>***

Note that in this example the data set and variable names are in italics and are enclosed in angle brackets (< >). Also note that there may be external data (from the laboratory, ECG Holter monitor, etc.) loaded into your clinical data management system, and you will want the specifications for those data as well.

Program a Task Once and Reuse Your Code Everywhere

One of the main reasons that you use computers is to perform repetitive tasks for you. If you apply that line of reasoning to your statistical programming, it will serve you well by preventing you from "reinventing the wheel." Another way to describe this is that you should strive to make your statistical programming modular in nature. We can look at a demonstration of modular programming by examining the SAS libref. Many if not most SAS programs begin with SAS LIBNAME statements that look something like these.

Program 1.1 Librefs That Commonly Appear at the Start of a SAS Program

```
libname trialdata "c:\mytrial\sasdata";
libname library "c:\mytrial\mysasformats";
libname otherdata "c:\someotherdata";
```

For a clinical trial you may have anywhere from a few dozen to a couple hundred different SAS programs, depending on the nature of the project. If you were to copy the LIBNAME statements above into 200 different programs, you might realize two things:

1. You had to copy three common lines of SAS code into 200 different places.
2. You now have a code maintenance problem.

The code maintenance problem surfaces when you realize that you need to change one of those SAS librefs. Then you have to edit many SAS programs to make this simple change. An alternative to having those three SAS librefs everywhere is to have them in a single location. The SAS macro facility provides two simple ways to do this. You could place those three LIBNAME statements in a single SAS program and use a %INCLUDE macro statement like this:

```
%include "c:\mylibrefs.sas";
```

Another approach would be to wrap a SAS macro around the three LIBNAME statements and call it with a simple SAS macro call. First, set up a SAS macro.

Program 1.2 Using a SAS Macro to Define Common Librefs

```
%macro mylibs;
   libname trialdata "c:\mytrial\sasdata";
   libname library "c:\mytrial\mysasformats";
   libname otherdata "c:\someotherdata";
%mend mylibs;
```

Then, call the SAS macro in another SAS program like this:

```
%mylibs
```

With either approach, what you have done is take a piece of SAS code common to many programs and put it in one place. If you ever have to make a change to one of those SAS librefs across all programs, you can easily change it in a single place. This practice is fundamental to good programming, and although it is possible to be overly modular, it is better to err on the side of making your SAS code more modular than to create SAS code maintenance problems over the long term.

Clinical Trial Data Are Dirty

People and their behavior are unique, and that is a wonderful thing. Unfortunately, the data that describe a patient's activities during a clinical trial tend to be unique as well. The clinical trial protocol and clinical trial staff make the best effort to guide the patient through a common treatment protocol, but this is often not enough to control the data coming from the patient. It is also often the case that the case report form used to collect the data turns out to be a less than perfect instrument for collecting what is needed for analyses. Finally, despite the best efforts of data management to provide a clean database, not all data fields are scoured. Therefore, you may be faced with a sometimes deviant and heterogeneous clinical trial database, so you need to be on guard for dirty or discrepant data.

The best way to protect yourself against dirty clinical trial data is to use good defensive programming techniques. In other words, you should write SAS code that accounts for all possible data permutations. Imagine you have a SAS data set that contains adverse event data for patients in a trial. Assume that the data set has only three fields: the subject ID (subjectid), a "yes or no" field describing whether the subject had an adverse event or not (aeyn), and the text description of the adverse event (aetext). To extract data for the patients who had an adverse event, you might set up a SAS data set as in the following program.

Program 1.3 Subsetting a Data Set for Patients with an Adverse Event

```
data aes;
   set aes;
      by subjectid;
         where aeyn = "YES";
run;
```

Now, consider what would happen if the SAS data set "aes" looks like this:

subjectid	aeyn	aetext
101	YES	Rash
102		Hives
103	NO	
104	YES	Headache

The SAS code you wrote would eliminate the observation for subjectid=102. This is because the "aeyn" field is not populated for that row and is therefore eliminated by the WHERE clause in SAS. This is a classic "*parent-child*" data problem in clinical trial data, where the "parent" question is left unanswered but the "child" response is given. A way to handle this problem would be either to include the "aetext" field in the WHERE clause or to add a warning to the SAS log. The code in Program 1.4 does both.

Program 1.4 Subsetting a Data Set for Patients with an Adverse Event Using Defensive Programming Techniques

```
data aes;
   set aes;
      by subjectid;

      **** PARENT-CHILD WARNING;
      if (aeyn ne "YES" and aetext ne "") or
         (aeyn = "YES" and aetext = "") then
         put "WARN" "ING: ae parent-child bug " aeyn= aetext=;
```

```
      **** GET AES;
      if aeyn = "YES" or aetext ne "";
run;
```

This SAS program first warns you when the "parent-child" data fields are out of synchrony and subsequently keeps all observations that could possibly indicate an adverse event. (Note that when the PUT statement is triggered, the "WARN" and "ING" are concatenated in the log file and signal a warning condition to SAS. The same trick can be used with "ERR" "OR" conditions as well. The benefit of breaking the "WARNING" and "ERROR" text in half in the SAS program is that it will be missed during text searching of SAS log files for warning and error conditions if none exist.)

Anywhere you have conditional logic is another place for defensive programming techniques. When there is conditional logic, there should be a catch-all follow-up statement. Assume you have SAS code such as the following.

Program 1.5 Example of Simple Conditional Logic IF-THEN/ELSE

```
if a > b then
   a + b;
else if a < b then
   a - b;
```

There should always be a follow-up ELSE statement to trap any potentially unforeseen conditions like the following.

Program 1.6 Example of Simple Conditional Logic IF-THEN/ELSE Using Defensive Programming Techniques

```
if a > b then
   a = a + b;
else if a < b then
   a = a - b;
else
   put "How does a relate to b? " a= b=;
```

The SAS SELECT statement is great for conditional processing because it has a mandatory OTHERWISE clause built into it to help catch unforeseen conditions. In a SAS SELECT statement, the code above would look like the code in the following program.

Program 1.7 Example of Simple Conditional Logic SELECT

```
select;
   when(a > b) a = a + b;
   when(a < b) a = a - b;
   otherwise put "What am I missing? " a= b=;
end;
```

In an optimal world, the CRF is perfectly designed to answer the questions of the study and the clinical data management group will have cleaned the data to perfection. However, to be a good statistical programmer in the clinical trial arena, you must always keep a lookout for errant data and program defensively. Defensive programming lets you account for all possible clinical data permutations.

Use SAS Macros Judiciously

The SAS macro language is a very powerful tool. With SAS macros you can write dynamic SAS applications that are in essence SAS programs that write other SAS programs. Unfortunately, with such great power comes the potential for great abuse. The SAS macro language can be abused when it is used to such an extent that a SAS program becomes unreadable. A SAS macro can become unreadable when it is too dense with macro invocations, is poorly documented, or involves too many nested macro calls. For instance, examine the following SAS code.

Program 1.8 Example of SAS Macro Code That You Should Not Write

```
data &&some&i;
   &getfile &subopt;
      &subset;
      %makecod
%run
```

Perhaps this is valid SAS code, but there is no code documentation to tell the user what any of those macro variables or macro calls actually do. Upon further investigation, we may find that the %MAKECOD macro calls six other SAS macros in a nested fashion. Also, is a %RUN really necessary, or has the programmer developed one too many macros?

The SAS macro language can also be abused when it is used in place of a built-in facility of SAS designed to solve the given task. A classic example is using the SAS macro language when simple SAS BY statement processing would work in its place. Examine the following SAS code, which prints out demographic data patient by patient.

Program 1.9 Reinventing SAS BY Processing with a SAS Macro

```
proc sort
   data = demog;
      by subjectid;
run;

**** SAS MACRO TO PRINT MY DEMOGRAPHIC DATA BY PATIENT;
%macro printpt(subjectid);
   proc print
      data = demog;
      where subjectid = "&subjectid";
      var subjectid age sex race;
   run;
%mend printpt;

%printpt(101-001)
%printpt(101-002)
%printpt(101-003)
```

Although the SAS code is valid and gets the job done, the following SAS code is better because it can handle unlimited "subjectid" while at the same time being less cumbersome to read.

Program 1.10 Using SAS BY Processing Instead of a SAS Macro

```
proc sort
   data = demog;
      by subjectid;
run;

**** PRINT MY DEMOGRAPHIC DATA BY PATIENT;
proc print
   data = demog;
      by subjectid;
         var subjectid age sex race;
run;
```

When the SAS macro language is used judiciously, it can be a powerful ally. SAS macros should be written so that they are not overly complicated, and they should always be the best-documented SAS programming code in any application. There is no worse fate than to be handed a complex SAS macro program with insufficient documentation. For more information about sound SAS macro programming practices, you can refer to the SAS Press books *Carpenter's Complete Guide to the SAS Macro Language*, by Art Carpenter, and *SAS Macro Programming Made Easy*, by Michele Burlew.

A Good Programmer Is a Good Student

Anyone who has programmed in any language has discovered that one of the best ways to learn to program is to learn from the examples of others. We all innovate when we author programs, but when faced with a new problem it is natural to look to others to see how they may have approached it. When you look to others for SAS programming help you will pick up new SAS tricks and insights. Each insight you gain from studying the efforts of others can be extrapolated into creating your own programs. This is essential to becoming a good programmer. So, where can you find a teacher or mentor?

Your search for the knowledge of SAS programming should begin locally. Look around in your department or organization for other statistical programmers who might serve as mentors and resources. These sage individuals have much to offer and can provide hands-on help to you in enhancing your programming skill. There is a wealth of knowledge outside your organization as well. SAS and SAS Publishing provide many papers and books that show how to maximize your use of SAS software. Local and national SAS users groups are also an excellent source of published papers on how to innovate in SAS. Finally, the Internet contains a treasure trove of SAS tricks and tips. There are many individual Web sites devoted to SAS, as well as the SAS-L listserv, which is a major source of SAS help. If you have a question related to SAS, it has probably been asked on SAS-L at some point. If it has not already been asked, then someone reading SAS-L will likely have an answer for you. Chapter 10, "Further Resources," contains a section that explains in detail how to use these resources.

Strive to Make Your Programming Readable

It is important that you make your programs readable. It is guaranteed that someone other than you will have to read your program at some point, and you want to make that task easy. A well-written SAS program has an ample supply of comments, employs white space to make the code pleasing to the eye, and uses a consistent case and indention style. For more information about basic SAS programming style, you may want to refer to *The Little SAS Book: A Primer*, by Lora Delwiche and Susan Slaughter.