

Appendix 1

Chapter Quiz Answer Keys

Chapter 2: Basic Concepts	377
Chapter 3: Accessing Your Data	378
Chapter 4: Creating SAS Data Sets	379
Chapter 5: Identifying and Correcting SAS Language Errors	380
Chapter 6: Creating Reports	381
Chapter 7: Understanding DATA Step Processing	383
Chapter 8: BY-Group Processing	384
Chapter 9: Creating and Managing Variables	384
Chapter 10: Combining SAS Data Sets	386
Chapter 11: Processing Data with DO Loops	387
Chapter 12: SAS Formats and Informats	388
Chapter 13: SAS Date, Time, and Datetime Values	389
Chapter 14: Using Functions to Manipulate Data	390
Chapter 15: Producing Descriptive Statistics	390
Chapter 16: Creating Output	391

Chapter 2: Basic Concepts

- Correct answer: c
Rows in the data set are called observations, and columns are called variables. Missing values do not affect the structure of the data set.
- Correct answer: a
When it encounters a DATA, PROC, or RUN statement, SAS stops reading statements and executes the previous step in the program. This program contains one DATA step and two PROC steps, for a total of three program steps.
- Correct answer: b
It must be a character variable, because the values contain letters and underscores, which are not valid characters for numeric values.

4. Correct answer: a

It must be a numeric variable, because the missing value is indicated by a period rather than by a blank. Missing values in numeric variables are indicated by a period while character values are indicated by a blank. Numeric values are also right justified while character values are left justified.

5. Correct answer: d

If you use `VALIDVARNAME=ANY`, the name can begin with or contain any characters, including blanks, national characters, special characters, and multi-byte characters. The name can be up to 32 bytes long. The name must contain at least one character, and the variable name can contain mixed-case letters.

6. Correct answer: d

To store a file permanently in a SAS data library, you assign it a libref. For example, by assigning the libref `Cert` to a SAS data library, you specify that files within the library are to be stored until you delete them. Therefore, SAS files in the `Cert` and `Certxl` libraries are permanent files.

7. Correct answer: d

To reference a temporary SAS file in a `DATA` step or `PROC` step, you can specify the one-level name of the file (for example, `Forecast`) or the two-level name using the libref `Work` (for example, `Work.Forecast`).

8. Correct answer: d

The numeric variable `Balance` has a default length of 8. Numeric values (no matter how many digits they contain) are stored in 8 bytes of storage unless you specify a different length.

9. Correct answer: c

The five statements are 1) the `PROC PRINT` statement (two lines long); 2) the `VAR` statement; 3) the `WHERE` statement (on the same line as the `VAR` statement); 4) the `LABEL` statement; and 5) the `RUN` statement (on the same line as the `LABEL` statement).

10. Correct answer: d

Every SAS file is stored in a SAS library. A SAS library is a collection of SAS files, such as SAS data sets and catalogs. In some operating environments, a SAS library is a physical collection of files. In others, the files are only logically related. In the Windows and UNIX environments, a SAS library is typically a group of SAS files in the same folder or directory.

Chapter 3: Accessing Your Data

1. Correct answer: d

Librefs remain in effect until the `LIBNAME` statement is changed, canceled, or until the SAS session ends.

2. Correct answer: b

When you are using the default engine, you do not have to specify the engine name in the `LIBNAME` statement. However, you do have to specify the libref and the SAS data library name.

3. Correct answer: a

A SAS engine is a set of internal instructions that SAS uses for writing to and reading from files in a SAS library. Each engine specifies the file format for files that are stored in the library, which in turn enables SAS to access files with a particular format. Some engines access SAS files, and other engines support access to other vendors' files.

4. Correct answer: a

To print a summary of library contents with the CONTENTS procedure, use a period to append the `_ALL_` option to the libref. Adding the NODS option suppresses detailed information about the files.

5. Correct answer: e

All the programs listed violate the rules for assigning a libref. Librefs must be 1 to 8 characters long, must begin with a letter or underscore, and can contain only numbers, letters, or underscores. After you assign a libref, you specify it as the first level in the two-level name for a SAS file.

6. Correct answer: c

The CONTENTS procedure creates a report that contains the contents of a library or the descriptor information for an individual SAS data set.

7. Correct answer: c

The LIBNAME statement is global, which means that librefs stay in effect until changed or canceled, or until the SAS session ends. Therefore, the LIBNAME statement assigns the libref for the current SAS session only. You must assign a libref before accessing SAS files that are stored in a permanent SAS data library.

8. Correct answer: d

The LIBNAME statement does not specify an engine. Therefore, it uses a default engine to create the States library.


Chapter 4: Creating SAS Data Sets

1. Correct answer: a

You assign a fileref by using a FILENAME statement in the same way that you assign a libref by using a LIBNAME statement.

2. Correct answer: b

By default, the IMPORT procedure reads delimited files as varying record-length files. If your external file has a fixed-length format, use the OPTIONS statement before the PROC IMPORT statement that includes the RECFM=F and LRECL= options.

3. Correct answer: ~~a~~ 

Use the OBS= option in the OPTIONS statement before the IMPORT procedure to limit the number of observations that SAS reads from the external file. When you use the OBS= option in the PROC PRINT statement, the whole file is imported but printing is limited to the number of observations specified. Use DELIMITER='.' to indicate that the delimiter is a period (.) and use GETNAMES=YES to read in the first line, which contains the variable names.

4. Correct answer: d

To read an Excel workbook file, SAS must receive the following information in the DATA step: a libref to reference the Excel workbook to be read, the name and location (using another libref) of the new SAS data set, and the name of the Excel worksheet that is to be read.

5. Correct answer: d

The GETNAMES statement specifies whether the IMPORT procedure generates SAS variable names from the data values in the first row in the input file. The default is GETNAMES=YES. NO specifies that the IMPORT procedure generates SAS variable names as VAR1, VAR2, and so on.

6. Correct answer: b

When you associate a fileref with an individual external file, you specify the fileref in subsequent SAS statements and commands.

7. Correct answer: d

The IMPORT procedure reads data from an external data source and writes it to a SAS data set. In delimited files, a delimiter (such as a blank, comma, or tab) separates columns of data values. You can also have a delimiter other than blanks, commas, or tabs. In those cases, PROC IMPORT reads the data from the external data source as well. You can have a delimiter such as an ampersand (&).

8. Correct answer: b

Placing an explicit OUTPUT statement in a DATA step overrides the automatic output, so that observations are added to a data set only when the explicit OUTPUT statement is executed. The OUTPUT statement overrides the default behavior of the DATA step.

Chapter 5: Identifying and Correcting SAS Language Errors

1. Correct answer: a

To correct errors in programs when you use the Editor window, you usually need to recall the submitted statements from the recall buffer to the Editor window. After correcting the errors, you can resubmit the revised program. However, before doing so, it is a good idea to clear the messages from the SAS log so that you do not confuse the old error messages with the new messages. Remember to check the SAS log again to verify that your program ran correctly.

2. Correct answer: d

The missing quotation mark in the LABEL statement causes SAS to misinterpret the statements in the program. When you submit the program, SAS is unable to resolve the PROC step, and a PROC PRINT running message appears at the top of the active window.

3. Correct answer: c

Syntax errors generally cause SAS to stop processing the step in which the error occurred. When a program that contains an error is submitted, messages regarding the problem also appear in the SAS log. When a syntax error is detected, the SAS log

displays the word ERROR, identifies the possible location of the error, and gives an explanation of the error.

4. Correct answer: c

Syntax errors occur because the program statements did not conform to the rules of the SAS language. Syntax errors, such as misspelled keywords, generally prevent SAS from executing the step in which the error occurred.

5. Correct answer: b

When you submit a SAS statement that contains an invalid option, a log message notifies you that the option is not valid or not recognized. You should recall the program, remove or replace the invalid option, check your statement syntax as needed, and resubmit the corrected program.

6. Correct answer: b

The DATA step contains a misspelled keyword (dat instead of data). However, this is such a common (and easily interpretable) error that SAS produces only a warning message, not an error.

7. Correct answer: d

The `_N_` and `_ERROR_` temporary variables can be helpful when debugging a DATA step. The `_N_` variable displays debugging messages for a specified number of iterations of the DATA step. The `_ERROR_` displays debugging messages when an error occurs.

8. Correct answer: d

You can use the PUTLOG statement or the PUT statement to help identify errors and print messages in the SAS log. You can use the PUTLOG statement in a DATA step to write messages to the SAS log to help identify logic errors. You can also use temporary variables in the PUTLOG to assist in debugging. You can use the PUT statement to examine variables and print your own message to the SAS log.

9. Correct answer: c

A logic error occurs when the program statements follow the rules and execute, but the results are not correct. You can use the PUTLOG statement in the DATA step to help identify logic errors.

Chapter 6: Creating Reports

1. Correct answer: c

The DATA= option specifies the data set that you are listing, and the ID statement replaces the Obs column with the specified variable. The VAR statement specifies variables and controls the order in which they appear, and the WHERE statement selects rows based on a condition. The LABEL option in the PROC PRINT statement causes the labels that are specified in the LABEL statement to be displayed.

2. Correct answer: a

You use the DATA= option to specify the data set to be printed. The LABEL option specifies that variable labels appear in output instead of in variable names.

3. Correct answer: d

In the WHERE statement, the IN operator enables you to select observations based on several values. You specify values in parentheses and separated by spaces or commas. Character values must be enclosed in quotation marks and must be in the same case as in the data set.

4. Correct answer: c

In a PROC SORT step, you specify the DATA= option to specify the data set to sort. The OUT= option specifies an output data set. The required BY statement specifies the variable or variables to use in sorting the data.

5. Correct answer: d

You do not need to name the variables in a VAR statement if you specify them in the SUM statement, but you can. If you choose not to name the variables in the VAR statement as well, then the SUM statement determines their order in the output.

6. Correct answer: c

The BY statement is required in PROC SORT. Without it, the PROC SORT step fails. However, the PROC PRINT step prints the original data set as requested.

7. Correct answer: a

Column totals appear at the end of the report in the same format as the values of the variables, so b is incorrect. Work.Loans is sorted by Month and Amount, so c is incorrect. The program sums both Amount and Payment, so d is incorrect.

8. Correct answer: c

To ensure that the compound expression is evaluated correctly, you can use parentheses to group the observations:

```
account='101-1092' or rate eq 0.095
```

OBS	Account	Amount	Rate	Months	Payment
1	101-1092	\$22,000	10.00%	60	\$467.43
2	101-1731	\$114,000	9.50%	360	\$958.57
3	101-1289	\$10,000	10.50%	36	\$325.02
4	101-3144	\$3,500	10.50%	12	\$308.52
5	103-1135	\$8,700	10.50%	24	\$403.47
6	103-1994	\$18,500	10.00%	60	\$393.07
7	103-2335	\$5,000	10.50%	48	\$128.02
8	103-3864	\$87,500	9.50%	360	\$735.75
9	103-3891	\$30,000	9.75%	360	\$257.75

For example, from the data set above, a and b select observations 2 and 8 (those that have a rate of 0.095); c selects no observations; and d selects observations 4 and 7 (those that have an amount less than or equal to 5000).

9. Correct answer: d

By default, PROC PRINT prints all observations and variables. An Obs column is generated to identify the observation number, and variables and observations appear in the order in which they occur in the data set.

Chapter 7: Understanding DATA Step Processing

1. Correct answer: b

During the compilation phase, the program data vector is created. The program data vector includes the two automatic variables `_N_` and `_ERROR_`. The descriptor portion of the new SAS data set is created at the end of the compilation phase. The descriptor portion includes the name of the data set, the number of observations and variables, and the names and attributes of the variables. Observations are not written until the execution phase.

2. Correct answer: a

Syntax checking can detect many common errors, but it cannot verify the values of variables or the correctness of formats.

3. Correct answer: c

The DATA step executes once for each record in the input file, unless otherwise directed.

4. Correct answer: d

The remaining variables are initialized to missing. Missing numeric values are represented by periods, and missing character values are represented by blanks.

5. Correct answer: b

The default value of `_ERROR_` is 0, which means there is no data error. When an error occurs, whether one error or multiple errors, the value is set to 1.

6. Correct answer: d

By default, at the end of the DATA step, the values in the program data vector are written to the data set as an observation. Then, control returns to the top of the DATA step, the value of the automatic variable `_N_` is incremented by one, and the values of variables that were created in programming statements are reset to missing. The automatic variable `_ERROR_` is reset to 0 if necessary.

7. Correct answer: a

The order in which variables are defined in the DATA step determines the order in which the variables are stored in the data set.

8. Correct answer: c

When SAS cannot detect syntax errors, the DATA step compiles, but it does not execute.

9. Correct answer: d

The variable type for Bonus is incorrect. When there is an incorrect variable type, SAS attempts to automatically convert to the correct variable type. If it cannot, SAS continues processing and produces output with missing values.

10. Correct answer: c

The FREQ procedure detects invalid character and numeric values by looking at distinct values. You can use PROC FREQ to identify any variables that were not given an expected value.

11. Correct answer: d

At the bottom of the DATA step, the compilation phase is complete, and the descriptor portion of the new SAS data set is created. There are no observations because the DATA step has not yet executed.

Chapter 8: BY-Group Processing

1. Correct answer: d

When you use the BY statement with the SET statement, the DATA step creates the temporary variables FIRST. and LAST. They are not stored in the data set.

2. Correct answer: d

Before you can perform BY-group processing, your data must follow a pattern. If your data is not ordered or grouped in some pattern, BY-group processing results in an error.

3. Correct answer: a

In the DATA step, during BY-group processing only, the temporary variables FIRST.variable and LAST.variable are available for DATA step programming, but they do not appear in the output data set.

4. Correct answer: c

The SORT procedure sorts the data Cert.Credit by the variable Type in ascending order. You do not have to specify the order in the BY statement in PROC SORT unless you are sorting in DESCENDING order.

5. Correct answer: b

A BY group includes all observations with the same BY value. If you use more than one variable in a BY statement, a BY group is a group of observations with the same combination of values for these variables. Each BY group has a unique combination of values for the variables.

6. Correct answer: c

SAS determines FIRST.variable by looking at each observation. When an observation is the first in a BY group, SAS sets the value of the FIRST.variable to 1. This happens when the value of the variable changed from the previous observation. For all other observations in the BY group, the value of FIRST.variable is 0.

7. Correct answer: a

The SORT procedure sorts the data Cert.Choices by the variable Day first, then Flavor in ascending order, and finally writes the sorted data set to Work.Choices.

Chapter 9: Creating and Managing Variables

1. Correct answer: c

Program c correctly deletes the observation in which the value of Finish is oak and the value of Price is less than 200. It also creates TotalPrice by summing the variable Price down observations, and then drops Price by using the DROP statement in the DATA step.

2. Correct answer: c

Logical comparisons that are enclosed in parentheses are evaluated as true or false before they are compared to other expressions. In the example, the AND comparison within the nested parentheses is evaluated before being compared to the OR comparison.

3. Correct answer: b

You must enclose character values in quotation marks, and you must specify them in the same case in which they appear in the data set. The value **OK** is not identical to **ok**, so the value of Count is not changed by the IF-THEN statement.

4. Correct answer: d

The length of a variable is determined by its first reference in the DATA step. When creating a new character variable, SAS allocates as many bytes of storage space as there are characters in the reference to that variable. The first reference to a new variable can also be made with a LENGTH statement or an assignment statement.

5. Correct answer: a

You can write multiple ELSE statements to specify a series of mutually exclusive conditions. The ELSE statement must immediately follow the IF-THEN statement in your program. An ELSE statement executes only if the previous IF-THEN/ELSE statement is false.

6. Correct answer: a

The length of a new variable is determined by the first reference in the DATA step, not by data values. In this case, the length of Type is determined by the value **Fixed**. The LENGTH statement is in the wrong place; it must occur before any other reference to the variable in the DATA step. You can run PROC CONTENTS on the data set to see the length of each variable.

7. Correct answer: b

To select variables, you can use a DROP or KEEP statement in any DATA step. You can also use the DROP= or KEEP= data set options following a data set name in any DATA or PROC step. However, you cannot use DROP or KEEP statements in PROC steps.

8. Correct answer: b

The variables Age, Weight, and Group are specified using the KEEP= option in the SET statement. When Cert.Fitness is being read, Age, Weight, and Group are the variables that create Work.Cardiac. The variables Age and Group are specified in the DROP= option in the DATA statement. Age and Group are dropped from Work.Cardiac.

9. Correct answer: c

You specify the data set to be created in the DATA statement. The DROP= data set option prevents variables from being written to the data set. Because you use the variable OrdTime when processing your data, you cannot drop OrdTime in the SET statement. If you use the KEEP= option in the SET statement, then you must list OrdTime as one of the variables to be kept.

Chapter 10: Combining SAS Data Sets

1. Correct answer: a

This example is a case of one-to-one matching, which requires multiple SET statements. Where same-named variables occur, values that are read from the second data set replace those that are read from the first data set. Also, the number of observations in the new data set is the number of observations in the smallest original data set.

2. Correct answer: b

This is a case of concatenation, which requires a list of data set names in the SET statement and one or more BY variables in the BY statement. Notice that observations in each BY group are read sequentially, in the order in which the data sets and BY variables are listed. The new data set contains all the variables from all the input data sets, as well as the total number of records from all input data sets.

3. Correct answer: a

Concatenating appends the observations from one data set to another data set. The new data set contains the total number of records from all input data sets, so b is incorrect. All the variables from all the input data sets appear in the new data set, so c is incorrect.


4. Correct answer: a

The concatenated data sets are read sequentially, in the order in which they are listed in the SET statement. The second observation in Work.Reps does not contain a value for Sale, so a missing value appears for this variable. (Note that if you merge the data sets, the value of Sale for the second observation is \$30,000.)

5. Correct answer: b

If you have variables with the same name in more than one input data set, values of the same-named variable in the first data set in which it appears are overwritten by values of the same-named variable in subsequent data sets.

6. Correct answer: a

The DATA step uses the IN= data set option, and the subsetting IF statement excludes unmatched observations from the output data set. So, answers ~~a and b~~  which contain unmatched observations, are incorrect.

7. Correct answer: d

Match-merging overwrites same-named variables in the first data set with same-named variables in subsequent data sets. To prevent overwriting, rename variables by using the RENAME= data set option in the MERGE statement.

8. Correct answer: c

The two input data sets are not sorted by values of the BY variable, so the DATA step produces errors and stops processing.

9. Correct answer: c

In this example, the new data set contains one observation for each unique value of ID. The new data set is shown below.

Obs	ID	Name	Dept	Project	Hours
1	000	Miguel	A12	Document	.
2	111	Fred	B45	Survey	35
3	222	Diana	B45	Document	40
4	777	Steve			0
5	888	Monique	A12	Document	37
6	999	Vien	D03	Survey	.

10. Correct answer: a

In the new data set, the third observation is the second observation for ID number 2 (Kelly Windsor). The value for Bonus is retained from the previous observation because the BY variable value did not change. The new data set is shown below.

Obs	ID	Name	Sale	Bonus
1	1	Nay Rong	\$28,000	\$2,000
2	2	Kelly Windsor	\$30,000	\$4,000
3	2	Kelly Windsor	\$40,000	\$4,000
4	3	Julio Meraz	\$15,000	\$3,000
5	3	Julio Meraz	\$20,000	\$3,000
6	3	Julio Meraz	\$20,000	\$3,000
7	3	Julio Meraz	\$25,000	\$3,000
8	4	Richard Krabill	\$35,000	\$2,500

Chapter 11: Processing Data with DO Loops

1. Correct answer: c

DO loops are DATA step statements and cannot be used in conjunction with PROC steps.

2. Correct answer: c

The number of iterations is determined by the DO statement's stop value, which in this case is 12.

3. Correct answer: a

Use a DO loop to perform repetitive calculations starting at 1 and looping 15 times.

4. Correct answer: d

At the end of the 15th iteration of the DO loop, the value for Year is incremented to 2005. Because this value exceeds the stop value, the DO loop ends. At the bottom of the DATA step, the current values are written to the data set.

5. Correct answer: b

The OUTPUT statement overrides the automatic output at the end of the DATA step. On the last iteration of the DO loop, the value of Year, 2004, is written to the data set.

6. Correct answer: d

The number of observations is based on the number of times the OUTPUT statement executes. The new data set has 20 observations, one for each iteration of the DO loop.

7. Correct answer: b

Place the monthly calculation in a DO loop within a DO loop that iterates once for each year. The DO WHILE and DO UNTIL statements are not used here because the number of required iterations is fixed. A non-iterative DO group would not be useful.

8. Correct answer: a

The DO UNTIL condition is evaluated at the bottom of the loop, so the enclosed statements are always executed at least once.

9. Correct answer: c

Because the DO WHILE loop is evaluated at the top of the loop, you specify the condition that must exist in order to execute the enclosed statements.

10. Correct answer: a

The WHILE expression causes the DO loop to stop executing when the value of Distance becomes equal to or greater than 250.

Chapter 12: SAS Formats and Informats

1. Correct answer: c

If you do not specify the LIBRARY= option, formats are stored in a default format catalog named Work.Formats. The libref Work signifies that any format that is stored in Work.Formats is a temporary format; it exists only for the current SAS session.

2. Correct answer: a

To store formats in a permanent catalog, you first write a LIBNAME statement to associate the libref with the SAS data library in which the catalog will be stored. Then add the LIB= (or LIBRARY=) option to the PROC FORMAT statement, specifying the name of the catalog.

3. Correct answer: d

The name of a format that is created with a VALUE statement must begin with a dollar sign (\$) if it applies to a character variable.

4. Correct answer: b

A semicolon is needed after the PROC FORMAT statement. The VALUE statement begins with the keyword VALUE and ends with a semicolon after all the labels have been defined.

5. Correct answer: d

You can list values separated by commas, but the list must contain either all numeric values or all character values. Data set variables are either numeric or character.

6. Correct answer: d

When specifying a label, enclose it in quotation marks and limit the label to 32,767 characters.

7. Correct answer: d

MISS and MISSING are invalid keywords, and LOW does not include missing numeric values. The keyword OTHER can be used in the VALUE statement to label missing values as well as any values that are not specifically included in a range.

8. Correct answer: b

By placing the FORMAT statement in a DATA step, you permanently associate the defined format with variables.

9. Correct answer: b

To associate a user-defined format with a variable, place a period at the end of the format name when it is used in the FORMAT statement.

10. Correct answer: d

Adding the keyword FMTLIB to the PROC FORMAT statement displays a list of all the formats in your catalog, along with descriptions of their values.

Chapter 13: SAS Date, Time, and Datetime Values

1. Correct answer: c

A SAS date value is the number of days from January 1, 1960, to the given date.

2. Correct answer: d

In addition to tracking time intervals, SAS date and time values can be used in calculations like other numeric values. This lets you calculate values that involve dates much more easily than in other programming languages.

3. Correct answer: b

SAS automatically makes adjustments for leap years.

4. Correct answer: d

The SAS informat MMDDYY w . reads dates such as 10222001, 10/22/01, or 10-22-01. In this case, the field width is eight.

5. Correct answer: b

The minimum acceptable field width for the TIME w . informat is five. If you specify a w value less than five, you receive an error message in the SAS log.

6. Correct answer: d

In the time value of a date and time expression, you must use delimiters to separate the values for hour, minutes, and seconds.

7. Correct answer: b

To find the number of days spanned by two dates, subtract the first day from the last day and add one. Because SAS date values are numeric values, they can easily be used in calculations.

Chapter 14: Using Functions to Manipulate Data

1. Correct answer: b

When this DATA step is executed, SAS automatically converts the character values of PayRate to numeric values so that the calculation can occur. Whenever data is automatically converted, a message is written to the SAS log stating that the conversion has occurred.

2. Correct answer: b

You explicitly convert character values to numeric values by using the INPUT function. Be sure to select an informat that can read the form of the values.

3. Correct answer: d

You explicitly convert numeric values to character values by using the PUT function. Be sure to select a format that can read the form of the values.

4. Correct answer: a

The SCAN function is used to extract words from a character value when you know the order of the words, when their position varies, and when the words are marked by some delimiter. In this case, you do not need to specify delimiters, because the blank and the comma are default delimiters.

5. Correct answer: d

The SUBSTR function is best used when you know the exact position of the substring to extract from the character value. You specify the position to start from and the number of characters to extract.

6. Correct answer: c

The SUBSTR function replaces variable values if it is placed on the left side of an assignment statement. When placed on the right side (as in Question 5), the function extracts a substring.

7. Correct answer: b

The TRIM function removes trailing blanks from character values. In this case, extra blanks must be removed from the values of FirstName. Although answer c also works, the extra TRIM function for the variable LastName is unnecessary. Because of the LENGTH statement, all values of FullName are padded to 40 characters.

8. Correct answer: d

Use the INDEX function in a subsetting IF statement, enclosing the character string in quotation marks. Only those observations in which the function locates the string and returns a value greater than 0 are written to the data set.

Chapter 15: Producing Descriptive Statistics

1. Correct answer: c

By default, the MEANS procedure produces the n, mean, minimum, maximum, and standard deviation.

2. Correct answer: d

To specify the variables that PROC MEANS analyzes, add a VAR statement and list the variable names.

3. Correct answer: a

Unlike Age, Height, or Weight, the values of IDnum are unlikely to yield any useful statistics.

4. Correct answer: a

Unlike CLASS processing, BY-group processing requires that your data already be indexed or sorted in the order of the BY variables. You might need to run the SORT procedure before using PROC MEANS with a BY group.

5. Correct answer: b

A CLASS statement produces a single large table, whereas BY-group processing creates a series of small tables. The order of the variables in the CLASS statement determines their order in the output table.

6. Correct answer: a

You can use PROC MEANS to create the table. The MEANS procedure provides data summarization tools to compute descriptive statistics for the variables Age, Height, and Weight for each Sex group.

7. Correct answer: c

By default, PROC FREQ creates a table for all variables in a data set.

8. Correct answer: c

Both continuous values and unique values can result in lengthy, meaningless tables. Frequency distributions work best with categorical values.

9. Correct answer: d

An asterisk is used to join the variables in a two-way TABLES statement. The first variable forms the table rows. The second variable forms the table columns.

10. Correct answer: d

An asterisk is used to join the variables in crosstabulation tables. The only results shown in this table are cell percentages. The NOFREQ option suppresses cell frequencies, the NOROW option suppresses row percentages, and the NOCOL option suppresses column percentages.

Chapter 16: Creating Output

1. Correct answer: d

You can generate any number of output types as long as you open the ODS destination for each type of output you want to create.

2. Correct answer: a

HTML output is created by default in the SAS windowing environment for the Windows operating environment and UNIX, so these statements create HTML and PDF output.

3. Correct answer: a

By default, in the SAS windowing environment for the Windows operating environment and UNIX, SAS programs produce HTML output. If you want only RTF output, it is a good idea to close the HTML destination before creating RTF output, as an open destination uses system resources.

4. Correct answer: c

When multiple procedures are run while HTML output is open, procedure output is appended to the same body file.

5. Correct answer: a

The CONTENTS= option creates a table of contents containing links to the body file, **D:\Output\body.html**.

6. Correct answer: b

The table of contents contains a numbered heading for each procedure that creates output.

7. Correct answer: c

The FRAME= option creates an HTML file that integrates the table of contents and the body file.

8. Correct answer: b

Specifying the URL= suboption in the file specification provides a URL that ODS uses in the links that it creates. Specifying a simple (one name) URL creates a relative link address to the file.

9. Correct answer: c

You can change the appearance of HTML output by using the STYLE= option in the ODS HTML statement. The style name does not need quotation marks.

10. Correct answer: d

You use the PATH= option to specify the location for HTML files to be stored. When you use the PATH= option, you do not need to specify the full path name for the body, contents, or frame files.

Appendix 2

Programming Scenario Solutions

Scenario 1	394
Code Solution	394
Test Your Code Solution	395
Scenario 2	395
Code Solution	395
Test Your Code Solution	396
Scenario 3	396
Code Solution	396
Test Your Code Solution	397
Scenario 4	398
Code Solution	398
Test Your Code Solution	401
Scenario 5	401
Code Solution	401
Test Your Code Solution	402
Scenario 6	402
Code Solution	402
Test Your Code Solution	404
Scenario 7	404
Code Solution	404
Test Your Code Solution	405
Scenario 8	405
Code Solution	405
Test Your Code Solution	407
Scenario 9	407
Code Solution	407
Test Your Code Solution	408
Scenario 10	408
Code Solution	408
Test Your Code Solution	409

Scenario 1

Code Solution

The solution listed below is one example of a program that could be used to accomplish each task within each scenario. Your code can be different, so long as it results in the same answers.

```
proc sort data=cert.patients out=work.patients; /* #1 */
  by id;
run;
proc sort data=cert.measure out=work.measure;
  by id;
run;
data work.merge; /* #2 */
  merge work.patients work.measure; /* #3 */
  by id; /* #4 */
  if age<50; /* #5 */
run;
proc sort data=work.merge out=work.sortpatients; /* #6 */
  by descending Age;
run;
proc print data=work.sortpatients; /* #7 */
run;
```

- 1 Sort Cert.Patients and Cert.Measure by ID. You specify the DATA= option to specify the data set to sort. The OUT= option specifies an output data set. The required BY statement specifies the variable or variables to use in sorting the data.
- 2 The DATA step creates a new temporary data set named Work.Merge.
- 3 The MERGE statement combines observations from Work.Patients and Work.Measure into a single observation in a new data set, Work.Merge, according to the values of a common variable.
- 4 The BY statement identifies the variable that the MERGE statement uses to combine observations. During match-merging, SAS sequentially checks each observation of each data set to see whether the BY values match and then writes the combined observation to the new data set.
- 5 The IF statement specifies that only patients under the age of 50 are read into Work.Merge.
- 6 Sort Work.Merge by Age in descending order. You specify the DATA= option to specify the data set to sort. The OUT= option specifies an output data set. The required BY statement specifies the variable or variables to use in sorting the data. The DESCENDING option precedes the variable name.
- 7 The PROC PRINT step enables you to view the contents of the sorted data set, Work.Sortpatients.

Output A2.1 PROC PRINT Output of Work.Sortpatients

Obs	ID	Sex	Age	Height	Weight
1	1129	F	48	61	137
2	5438	F	42	62	168
3	8045	M	40	72	200
4	8125	M	39	70	176
5	9012	F	39	63	157
6	2304	F	16	61	102

Test Your Code Solution

1. Correct Answer: 16
2. Correct Answer: 200

If your answers are not correct, verify that you have sorted your data in descending order and that you used the PRINT procedure to print Work.Sortpatients.

Scenario 2

Code Solution

The solution listed below is one example of a program that could be used to accomplish each task within each scenario. Your code can be different, so long as it results in the same answers.

```

data work.stress1;                                /* #1 */
  set cert.stress;                                /* #2 */
  where RestHR <=70;                              /* #3 */
  TotalTime=(timemin*60)+timesec;                /* #4 */
  if TotalTime<600 then delete;                  /* #5 */
run;
proc print data=work.stress1;                      /* #6 */
run;

```

- 1 The DATA step creates a new, temporary data set named, Work.Stress1.
- 2 The SET statement specifies the SAS data set that you want to read from. To create Work.Stress1, you read from Cert.Stress.
- 3 The WHERE statement selects only the observations where the values of RestHR are greater than or equal to 70.
- 4 The assignment statement creates the TotalTime variable by multiplying the value of TimeMin by 60 and adding the value of TimeSec. The values of TotalTime are assigned to each observation.
- 5 The IF-THEN and DELETE statements subset the data by omitting observations that have a TotalTime variable value less than 600.

- 6 The PROC PRINT step enables you to view the contents of the new data set, Work.Stress1.

Output A2.2 PROC PRINT Output of Work.Stress1

Obs	ID	Name	RestHR	MaxHR	RecHR	TimeMin	TimeSec	Tolerance	TotalTime
1	2462	Almers, C	68	171	133	10	5	I	605
2	2552	Reberson, P	69	158	139	15	41	D	941
3	2555	King, E	70	167	122	13	13	I	793
4	2571	Nunnelly, A	65	181	141	15	2	I	902
5	2586	Derber, B	68	176	119	17	35	N	1055
6	2588	Ivan, H	70	182	126	15	41	N	941

Test Your Code Solution

1. Correct Answer: 6
2. Correct Answer: 1055

If your answers are not correct, verify that you omitted the observations from the Work.Stress1 data set.

Scenario 3

Code Solution

The solution listed below is one example of a program that could be used to accomplish each task within each scenario. Your code can be different, so long as it results in the same answers.

```

data work.staffreports;           /* #1 */
  set cert.staff;                 /* #2 */
  where WageCategory ne 'H';      /* #3 */
  format DOB mmddyy10.;          /* #4 */
  Raise=WageRate*0.03;            /* #5 */
run;
proc print data=work.staffreports; /* #6 */
  sum Raise;                      /* #7 */
run;

```

- 1 The DATA step creates a new data set named Work.Staffreports.
- 2 The SET statement specifies the SAS data set that you want to read from. To create Work.Staffreports, you read from Cert.Staff.
- 3 The WHERE statement selects only the observations for the values of WageCategory that do not equal H.
- 4 The FORMAT statement formats the DOB variable in the mmddyy10. format.

- 5 The assignment statement creates the Raise variable. The values for Raise are assigned for each observation by multiplying the value of WageRate by 3%.
- 6 The PROC PRINT step enables you to view the contents of the new data set, Work.Staffreports.
- 7 The SUM statement generates a grand total for the Raise variable.

Output A2.3 PROC PRINT Output of Work.Staffreports

Obs	ID	Name	DOB	WageCategory	WageRate	Bonus	Raise
1	1351	Farr, Sue	03/05/1947	S	3392.50	1187.38	101.78
2	161	Cox, Kay B	12/31/1945	S	5093.75	1782.81	152.81
3	212	Moore, Ron	05/22/1953	S	1813.30	634.65	54.40
4	2512	Ruth, G H	04/13/1952	S	1572.50	550.37	47.18
5	282	Shaw, Rick	07/17/1951	S	2192.25	767.29	65.77
6	3782	Bond, Jim S	12/04/1948	S	2247.50	786.63	67.43
7	381	Smith, Anna	06/09/1950	S	2082.75	728.96	62.48
8	3922	Dow, Tony	10/04/1947	S	2960.00	1036.00	88.80
9	412	Star, Burt	02/19/1956	S	2300.00	805.00	69.00
10	442	Lewis, Ed D	03/04/1950	S	3420.00	1197.00	102.60
11	452	Fox, Jim E	11/09/1945	S	3902.35	1365.82	117.07
12	4551	Wong, Kim P	06/12/1942	S	3442.50	1204.88	103.28
13	472	Hall, Joe B	07/17/1961	S	2262.50	791.88	67.88
14	482	Chin, Mike	12/02/1952	S	2938.00	1028.30	88.14
15	5002	Welch, W B	09/21/1957	S	5910.75	2068.76	177.32
16	5112	Delgado, Ed	08/25/1948	S	4045.85	1416.05	121.38
17	511	Vega, Julie	10/01/1957	S	4480.50	1568.18	134.42
18	5132	Overby, Phil	06/08/1951	S	6855.90	2399.57	205.68
19	5151	Coxe, Susan	01/19/1932	S	3163.00	1107.05	94.89
20	1351	Farr, Sue	03/05/1947	S	3392.50	1187.38	101.78
							2024.05

Test Your Code Solution

1. Correct Answer: 07/17/1951
2. Correct Answer: 177.32
3. Correct Answer: 2024.05

If your answers are not correct, verify that you have observations from the Work.Staffreports data set.

Scenario 4

Code Solution

The solution listed below is one example of a program that could be used to accomplish each task within the scenario. Your code can be different, as long as it results in the same answers.

```
proc sort data=cert.laguardia out=work.laguardia;      /* #1 */
  by dest;
run;
title 'Laguardia Flights';                             /* #2 */
ods pdf file='LGA Airport' style=FestivalPrinter;      /* #3 */
proc print data=work.laguardia;                         /* #4 */
  by dest;                                              /* #5 */
run;
ods pdf close;                                          /* #6 */
```

- 1 When using the SORT procedure, the DATA= option specifies the input data set, and the OUT= option specifies the output data set. The required BY statement specifies the sorting variables.
- 2 The TITLE statement specifies title lines for SAS output. In this example, the TITLE statement titles the output Laguardia Flights.
- 3 The ODS PDF statement opens the PDF destination, which produces a PDF output. The PDF file is named LGA Airport, and FestivalPrinter is used as the style template with a .pdf extension.
- 4 The PROC PRINT statement prints the observations of Work.Laguardia using all of the variables. See [Output A2.5 on page 400](#).
- 5 The BY statement in the PRINT procedure produces a separate section in the report for each BY group. As there are four destinations in Work.Laguardia, four separate sections are produced.
- 6 The ODS PDF CLOSE statement closes the PDF destination.

Output A2.4 Partial Results: PROC PRINT Output of Work.Laguardia

Obs	Flight	Date	Depart	Orig	Dest	Boarded	Transferred	Deplaned	Revenue
1	387	04MAR12	11:40	LGA	CPH	81	21	103	196540
2	387	05MAR12	11:40	LGA	CPH	142	8	152	134561
3	387	07MAR12	11:40	LGA	CPH	131	5	142	135632
4	387	08MAR12	11:40	LGA	CPH	150	9	162	128564
5	387	09MAR12	11:40	LGA	CPH	128	14	145	134523

...more observations...

42	271	05MAR12	13:17	LGA	PAR	177	22	203	128972
43	271	07MAR12	13:17	LGA	PAR	155	21	180	153423
44	271	08MAR12	13:17	LGA	PAR	152	20	176	133345
45	271	09MAR12	13:17	LGA	PAR	159	18	182	126543
46	271	10MAR12	13:17	LGA	PAR	182	9	198	134976

Output A2.5 Partial Results: PROC PRINT Output of Work.Laguardia**Laguardia Flights**

Dest=CPH

Obs	Flight	Date	Depart	Orig	Boarded	Transferred	Deplaned	Revenue
1	387	04MAR12	11:40	LGA	81	21	103	196540
2	387	05MAR12	11:40	LGA	142	8	152	134561
3	387	07MAR12	11:40	LGA	131	5	142	135632
4	387	08MAR12	11:40	LGA	150	9	162	128564
5	387	09MAR12	11:40	LGA	128	14	145	134523
6	387	10MAR12	11:40	LGA	154	18	177	109885

...more observations...

Dest=PAR

Obs	Flight	Date	Depart	Orig	Boarded	Transferred	Deplaned	Revenue
34	271	04MAR12	11:40	LGA	146	8	163	156804
35	271	05MAR12	12:19	LGA	177	15	227	190098
36	271	07MAR12	9:31	LGA	155	18	172	166470
37	271	08MAR12	12:19	LGA	152	7	187	163248
38	271	09MAR12	13:17	LGA	159	15	191	170766
39	271	10MAR12	11:40	LGA	182	9	153	195468
40	271	03MAR12	13:17	LGA	147	29	183	123456
41	271	04MAR12	13:17	LGA	146	13	163	125632
42	271	05MAR12	13:17	LGA	177	22	203	128972
43	271	07MAR12	13:17	LGA	155	21	180	153423
44	271	08MAR12	13:17	LGA	152	20	176	133345
45	271	09MAR12	13:17	LGA	159	18	182	126543
46	271	10MAR12	13:17	LGA	182	9	198	134976

Output A2.6 Partial Output: PDF Output: LGA Airport

Bookmarks

Print

Dest=CPH

 Data Set
 WORK.LAGUARDIA

Dest=FRA

 Data Set
 WORK.LAGUARDIA

Dest=LON

 Data Set
 WORK.LAGUARDIA

Dest=PAR

 Data Set
 WORK.LAGUARDIA

Laguardia Flights

Friday, December 7,

Dest=CPH

Obs	Flight	Date	Depart	Orig	Boarded	Transferred	Deplaned	Revenue
1	387	04MAR12	11:40	LGA	81	21	103	196540
2	387	05MAR12	11:40	LGA	142	8	152	134561
3	387	07MAR12	11:40	LGA	131	5	142	135632
4	387	08MAR12	11:40	LGA	150	9	162	128564
5	387	09MAR12	11:40	LGA	128	14	145	134523
6	387	10MAR12	11:40	LGA	154	18	177	109885

Dest=FRA

Obs	Flight	Date	Depart	Orig	Boarded	Transferred	Deplaned	Revenue
7	622	03MAR12	12:19	LGA	180	16	200	187636
8	622	04MAR12	12:19	LGA	137	14	155	165456
9	622	05MAR12	12:19	LGA	185	11	199	125436
10	622	07MAR12	12:19	LGA	210	22	237	107865
11	622	08MAR12	12:19	LGA	176	5	187	178543
12	622	09MAR12	12:19	LGA	173	11	189	100987
13	622	10MAR12	12:19	LGA	129	12	147	134459

Dest=LON

Obs	Flight	Date	Depart	Orig	Boarded	Transferred	Deplaned	Revenue
14	219	04MAR12	9:31	LGA	232	18	250	189065
15	219	05MAR12	9:31	LGA	160	4	167	197456
16	219	06MAR12	9:31	LGA	163	14	183	162343
17	219	07MAR12	9:31	LGA	241	9	250	134520
18	219	08MAR12	9:31	LGA	183	11	197	106753
19	219	09MAR12	9:31	LGA	211	18	235	122766
20	219	10MAR12	9:31	LGA	167	7	181	198744

Test Your Code Solution

1. Correct Answer: 189KB – 199KB. You might get a slightly different answer, depending on your hardware, operating system, and software version. On the actual exam, all candidates work from the same cloned virtual machine, so the results will be consistent for grading.
2. Correct Answer: 129
3. Correct Answer: PAR

If your answers are not correct, verify that you used a BY statement in your PROC PRINT statement.

Scenario 5

Code Solution

The highlighted portions below illustrate the areas where corrections are required in order to make this program run and generate results.

```

data work.aprilbills (drop=Total EquipCost);
  set cert.aprbills;
  if Days>7 then Discount=(RoomCharge)*.20;
  else Discount=0;
  TotalDue=Total-Discount;
  format DateIn DateOut date9.;
  format RoomRate RoomCharge Discount TotalDue dollar10.2;
run;
proc print data=work.aprilbills;
run;

```

Output A2.7 PROC PRINT Output of AprilBills

Obs	LastName	DateIn	DateOut	RoomRate	Days	RoomCharge	Discount	TotalDue
1	Akron	05APR2009	09APR2009	\$175.00	5	\$875.00	\$0.00	\$1,173.45
2	Brown	12APR2009	01MAY2009	\$125.00	20	\$2,500.00	\$500.00	\$2,326.78
3	Carnes	27APR2009	29APR2009	\$125.00	3	\$375.00	\$0.00	\$549.24
4	Denison	11APR2009	12APR2009	\$175.00	2	\$350.00	\$0.00	\$437.41
5	Fields	15APR2009	22APR2009	\$175.00	8	\$1,400.00	\$280.00	\$1,498.96
6	Jamison	16APR2009	23APR2009	\$125.00	8	\$1,000.00	\$200.00	\$1,146.28

Test Your Code Solution

1. Correct Answer: \$437.41
2. Correct Answer: \$280.00

Scenario 6**Code Solution**

The solution listed below is one example of a program that could be used to accomplish each task within the scenario. Your code can be different, as long as it results in the same answers.

```

libname certdata XLSX 'C:\Users\certdata\heart.xlsx';
data work.heart;
  set certdata.heart(drop=AgeAtDeath DeathCause);
  where Status='Alive';
  if AgeCHDDdiag=. then delete;
  length Smoking_Status $17;
  if 0<=Smoking<6 then Smoking_Status='Non-Smoker (0-5)';
  else if 6<=Smoking<=15 then Smoking_Status='Moderate (6-15)';
  else if 16<=Smoking<=25 then Smoking_Status='Heavy (16-25)';
  else if Smoking>25 then Smoking_Status='Very Heavy (> 25)';
  else Smoking_Status='Error';
run;
proc freq data=work.heart;
  tables AgeCHDDdiag*Smoking_Status/norow nocol nopercnt;

```

run;

- 1 The SAS/ACCESS LIBNAME statement creates the libref certdata, which points to the Excel workbook heart.xlsx.
- 2 The DATA step creates a new temporary data set named Work.Heart.
- 3 The SET statement indicates which worksheet in the Excel file to read. The SET statement specifies the libref (the reference to the Excel file) and the worksheet name as the input data. The DROP= data set option excludes the variables AgeAtDeath and DeathCause from being written to the data set. The DROP statement could also have been used.
- 4 The WHERE statement selects the observations where the value of the Status variable is Alive.
- 5 The IF statement causes the DATA step to continue processing only those observations that meet the condition of the expression specified in the IF statement. In the example, if the value of AgeCHDdiag is missing, then those observations are removed from the data set.
- 6 The LENGTH statement specifies that the length of Smoking_Status is set to 17 and is a character variable. This is used to avoid truncating values.
- 7 The IF/ELSE IF statements create values for the Smoking_Status variable by subsetting smoking values.
- 8 The ELSE statement gives an alternative action if all the other IF-THEN/ELSE statements are not executed.
- 9 The FREQ procedure creates a two-way frequency for Work.Heart.
- 10 The TABLES statement requests a two-way frequency table for the variables AgeCHDdiag and Smoking_Status. The options norow, nocol, and noperc suppress row percentages, column percentages, and cell percentages.

Output A2.8 Partial Results: PROC FREQ Results

Frequency	Table of AgeCHDdiag by Smoking_Status						
	AgeCHDdiag(AgeCHDdiag)	Smoking_Status					Total
		Error	Heavy (16-25)	Moderate (6-15)	Non-Smoker (0-5)	Very Heavy (> 25)	
	32	0	0	0	1	0	1
	33	0	1	0	1	0	2
	36	0	1	0	0	0	1
	37	0	0	0	2	0	2
	38	0	1	0	1	0	2
...more observations...							
	84	0	0	2	2	0	4
	85	0	1	0	2	0	3
	86	0	0	0	3	0	3
	87	0	0	0	1	0	1
	88	0	0	0	2	0	2
	Total	3	102	56	350	44	555

Test Your Code Solution

1. Correct Answer: 102
2. Correct Answer: 2
3. Correct Answer: 3

Scenario 7**Code Solution**

The solution listed below is one example of a program that could be used to accomplish each task within the scenario. Your code can be different, as long as it results in the same answers.

```
data work.scenario7;          /* #1 */
  set cert.temp18;           /* #2 */
  format Day date9.;         /* #3 */
  Month=month(day);          /* #4 */
run;
proc freq data=work.scenario7; /* #5 */
  tables HighTemp;           /* #6 */
run;
proc means data=work.scenario7; /* #7 */
  class month;               /* #8 */
  var AvgLowTemp AvgHighTemp; /* #9 */
run;
```

- 1 The DATA step creates a new temporary data set named Work.Scenario7.
- 2 The SET statement is used to read observations from one or more SAS data sets.
- 3 The FORMAT statement formats the Day variable in the date9. format.
- 4 The MONTH function returns the numeric value of the month within the Day variable.
- 5 The FREQ procedure creates one-way, two-way, and *n*-way tables. It also describes data by reporting the distribution of variable values.
- 6 The TABLES statement requests one-way to *n*-way frequency and crosstabulation tables and statistics. In this case, that is a one-way frequency with the default statistics for the variable HighTemp.
- 7 The MEANS procedure is used to compute descriptive statistics for the variables stated in the VAR statement.
- 8 The CLASS statement provides separate calculations for each value of the Month variable.
- 9 The VAR statement identifies the two variables, AvgLowTemp and AvgHighTemp, as the analysis variables, and also controls their order in the output.

Output A2.9 PROC FREQ Results of High Temp

HighTemp	Frequency	Percent	Cumulative Frequency	Cumulative Percent
21	2	2.22	2	2.22
23	1	1.11	3	3.33
26	2	2.22	5	5.56
27	1	1.11	6	6.67
28	1	1.11	7	7.78

...more observations...

68	2	2.22	85	94.44
74	1	1.11	86	95.56
77	1	1.11	87	96.67
78	2	2.22	89	98.89
82	1	1.11	90	100.00

Output A2.10 PROC MEANS Results

Month	N Obs	Variable	N	Mean	Std Dev	Minimum	Maximum
1	31	AvgLowTemp	31	28.6129032	0.4951376	28.0000000	29.0000000
		AvgHighTemp	31	43.1612903	1.7145801	35.0000000	47.0000000
2	28	AvgLowTemp	28	31.7500000	1.4813657	30.0000000	34.0000000
		AvgHighTemp	28	48.2857143	2.0880106	45.0000000	52.0000000
3	31	AvgLowTemp	31	37.6129032	2.5778210	34.0000000	42.0000000
		AvgHighTemp	31	55.8387097	3.1738176	51.0000000	61.0000000

Test Your Code Solution

1. Correct Answer: 2
2. Correct Answer: 64
3. Correct Answer: 29
4. Correct Answer: 3.17

Scenario 8**Code Solution**

The solution listed below is one example of a program that could be used to accomplish each task within the scenario. Your code can be different, as long as it results in the same answers.

```
data work.scenario8;                                /*#1*/
```

```

set cert.addresses;                                /* #2 */
Zipcode=substr(State,3,5);                          /* #3 */
State=substr(State,1,2);                            /* #4 */
run;
proc print data=work.scenario8;                     /* #5 */
  where zipcode='85069';
run;
proc freq data=work.scenario8 order=freq;           /* #6 */
  tables State;                                     /* #7 */
run;

```

- 1 The DATA step creates a temporary data set named Work.Scenario8.
- 2 The SET statement reads observations from one or more SAS data sets.
- 3 The assignment statement creates a new variable, Zipcode, which uses the SUBSTR function to extract the last 5 characters of the values in the variable State, starting at and including character 3.
- 4 The assignment statement replaces the value of State and uses the SUBSTR function to extract 2 characters of the values in the variable State, starting at and including character 1.
- 5 The PRINT procedure enables you to view the contents of the new data set, Work.Scenario8 where Zipcode is equal to 85069.
- 6 The FREQ procedure creates one-way, two-way, and n -way tables. It also describes data by reporting the distribution of variable values. The ORDER=FREQ option orders the table by descending frequency.
- 7 The TABLES statement requests a one-way frequency with the default statistics for the variable State.

Output A2.11 PROC PRINT Results

Obs	Street	City	State	Tel	Zipcode
45	1861 Clarksburg Road	Harquala Valley	AZ	928-372-871	85069

Output A2.12 PROC FREQ Results

State	Frequency	Percent	Cumulative Frequency	Cumulative Percent
FL	4	6.56	4	6.56
NC	4	6.56	8	13.11
CA	3	4.92	11	18.03
NY	3	4.92	14	22.95
PA	3	4.92	17	27.87

...more observations...

NJ	1	1.64	57	93.44
NM	1	1.64	58	95.08
UT	1	1.64	59	96.72
WA	1	1.64	60	98.36
WI	1	1.64	61	100.00

Test Your Code Solution

1. Correct Answer: 3
2. Correct Answer: 45
3. Correct Answer: 2

Scenario 9

Code Solution

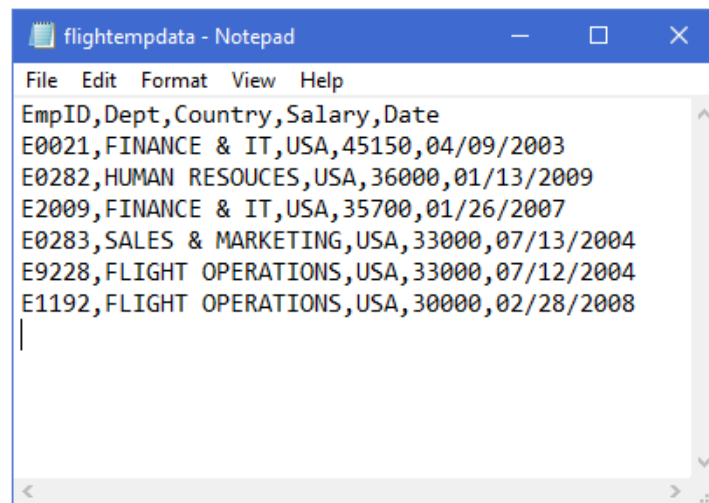
The solution listed below is one example of a program that could be used to accomplish each task within the scenario. Your code can be different, as long as it results in the same answers.

```

%let Location=USA;                                /* #1 */
data work.flightempdata;                          /* #2 */
    set cert.empdata cert.empdatu cert.empdatu2;  /* #3 */
    where Country="&Location" and Salary >= 30000; /* #4 */
run;
proc sort data=work.flightempdata;                /* #5 */
    by descending Salary;
run;
proc export data=work.flightempdata                /* #6 */
    outfile="C:\cert\flightempdata.csv"
    dbms=csv
    replace;
run;

```

- 1 The %LET statement creates a macro variable named Location that stores the character variable value of USA.
- 2 The DATA step creates a new temporary data set named Work.Flighttempdata.
- 3 The SET statement reads and concatenates the observations from the Cert.Empdata, Cert.Empdatu, and Cert.Empdatu2 data sets in that order.
- 4 The WHERE statement selects observations from the SAS data sets Cert.Empdata, Cert.Empdatu, and Cert.Empdatu2 that have a value for Country that is equal to the value of the macro variable &location. The statement also selects observations that have a value of Salary greater than or equal to \$30,000.
- 5 The PROC SORT step sorts the SAS data set Work.Flighttempdata by the values of the variable Salary in descending order.
- 6 PROC EXPORT exports the SAS data set Work.Flighttempdata to a comma-separated value file. The DATA= option identifies the input SAS data set, and the OUTFILE= option specifies the complete path and filename for the delimited external file. The DBMS = option specifies the type of data to export (in this case CSV), and the REPLACE option overwrites an existing file.

Output A2.13 PROC EXPORT Result: Flighttempdata.csv**Test Your Code Solution**

1. Correct Answer: \$33,000
2. Correct Answer: 290 – 300 bytes (any number within this range is an acceptable and correct answer)

Scenario 10**Code Solution**


The highlighted portions below illustrate the areas where corrections are required in order to make this program run and generate results.

```
data work.mycars;
```



```
set sashelp.cars;  
AvgMPG=mean(mpg_city, mpg_highway);  
run;  
title 'Cars With Average MPG Over 40';  
proc print data=work.mycars;  
var make model type avgmpg;  
where AvgMPG>40;  
run;  
title 'Average MPG by Car Type';  
proc means data=work.mycars mean min max maxdec=1;  
var avgmpg;  
class type;  
run;  
title;
```

Test Your Code Solution

1. Correct Answer: ~~262~~ 
2. Correct Answer: 4

Index

Special Characters

`_ERROR_` automatic variable
 DATA step iterations and 117
 functionality 112
 initializing variables 115
`_N_` automatic variable
 DATA step processing and 116
 functionality 112
`$w.` format 227

A

AND operator
 examples 153
 in SAS expressions 143
 in WHERE statement, PRINT
 procedure 82
 APPEND procedure
 functionality 177
 appending data sets 177
 arguments in functions 255
 arithmetic operators in SAS expressions
 142
 assignment statements 144
 conditional processing 152
 date constants 145
 examples 144, 145
 positioning SUBSTR function 282
 SAS expressions in 142
 asterisk (*) 324
 attributes
 See variable attributes

B

BEST12. format 261
 BESTw. format 312
 BY clause, DO statement 214
 BY group 129
 BY statement
 DESCENDING option 187
 group processing with 134, 314
 match-merging data sets 177, 184
 PRINT procedure 89, 91

 SORT procedure 87, 187
 syntax 314
 BY value 129
 BY variable 129
 BY-group processing
 BY group 129
 BY value 129
 BY variable 129
 DATA step 129
 determining FIRST. LAST. variables
 132
 FIRST.variable 131
 LAST.variable 131
 sorting observations 130

C

calculations
 dates and times in 247
 case sensitivity
 format values 232
 CATX function
 functionality 288
 syntax 288
 CEIL function 300
 character strings 66
 PUT statement 66
 searching 289, 291
 specifying delimiters 278
 testing programs 66
 character variables
 removing trailing blanks 286, 287
 replacing 282
 searching for strings 289
 character-to-numeric conversions 256,
 257
 CLASS statement, MEANS procedure
 313
 cleaning data 121
 CLM statistic 309
 columns
 See variables
 combining data sets 177
 by appending 177

- by concatenating 177, 182
 - by match-merging 177, 184
 - by one-to-one reading 177, 178
 - excluding unmatched observations 196
 - methods for 177
 - renaming variables 194
 - COMMA9. informat 260
 - COMMA9.2 format 228
 - COMMAw.d format 227
 - common errors
 - missing RUN statement 60
 - missing semicolon (;) 61
 - unbalanced quotation mark 61
 - compilation phase (DATA step)
 - data set variables 112
 - descriptor portion of data sets 113
 - diagnosing errors in 120
 - match-merge processing 188
 - program data vector 112
 - syntax checking 112
 - concatenating data sets 177, 182, 183
 - concatenation operator (||) 261
 - conditional processing
 - assigning variable values 152
 - assignment statements 152
 - DO groups 207
 - DO loops 218
 - providing alternative actions 154
 - PUT statement and 67
 - testing programs 67, 155
 - CONTAINS operator 82
 - CONTENTS procedure
 - reading Microsoft Excel data 48
 - viewing library contents 28
 - converting data
 - See* data conversion
 - CROSSLIST option, TABLES statement (FREQ) 327
 - CSS statistic
 - MEANS procedure 309
 - CV statistic
 - MEANS procedure 309
- D**
- data cleaning 121
 - data conversion 256
 - character-to-numeric 257
 - lowercase 294
 - numeric-to-character 256
 - numeric-to-character conversion 261
 - uppercase 293
 - data sets
 - See also* combining data sets
 - See also* match-merging data sets
 - BY-group processing 134
 - data portion 20
 - descriptor portion 17, 113
 - dropping and keeping variables 156
 - iteratively processing data 217
 - manipulating data 146
 - missing values 21
 - naming 44
 - naming conventions 14
 - observations (rows) 20
 - reading 45
 - specifying observations via system
 - options 85
 - summarized using MEANS procedure 316
 - testing programs 66
 - variable attributes 18, 112
 - variables and 21
 - DATA step
 - BY-group processing 129
 - checking processing 42
 - compilation phase 188
 - creating/modifying variables 142
 - debugging 120
 - execution phase 189
 - functionality 109
 - iterations of 117
 - manipulating data 146
 - naming data sets 44
 - reading Microsoft Excel data 48, 51
 - submitting 42
 - syntax 44
 - writing 44, 51
 - data validation 121
 - DATDIF function 263, 275
 - DATE function 263, 271
 - DATE7. format 228
 - DATE9. format 228, 265, 266
 - dates and times
 - in calculations 247
 - informat support 243
 - manipulating with functions 241
 - DATETIMEw. format 243
 - DATETIMEw. informat 246
 - DATEw. format
 - examples 227, 243
 - DATEw. informat
 - functionality 245
 - syntax 245
 - DAY function
 - functionality 254
 - manipulating date values 264
 - syntax 264
 - typical use 263
 - debugging
 - cleaning data 121

- diagnosing errors in compilation phase 120
 - diagnosing errors in execution phase 121
 - validating data 121
 - decimal places, limiting 312
 - DELETE statement
 - example 155
 - in IF-THEN statement 155
 - delimiters
 - for SCAN function 278
 - specifying multiple 279
 - DESCENDING option
 - BY statement, SORT procedure 87
 - descriptive statistics 307
 - creating summarized data sets 316
 - creating tables in list format 326
 - group processing with BY statement 314
 - group processing with CLASS statement 313
 - limiting decimal places 312
 - procedure syntax 308
 - producing frequency tables 317, 322, 324
 - selecting statistics 309
 - specifying variables in FREQ procedure 322
 - specifying variables in MEANS procedure 313
 - DO groups
 - indenting 215
 - iteratively processing data 214
 - nesting 215
 - DO loops
 - conditionally executing 218
 - constructing 210
 - decrementing 213
 - functionality 212
 - index variables 213, 216
 - nesting 215
 - specifying series of items 214
 - DO statement
 - See also* iterative DO statements
 - BY clause 214
 - grouping statements 207
 - DO UNTIL statement 218
 - DO WHILE statement
 - functionality 218, 219
 - syntax 219
 - Document destination 336
 - dollar sign (\$)
 - in format names 232
 - name literals and 49, 52
 - DOLLAR10.2 format 228
 - DOLLAR8.2 format 228
 - DOLLAR9.2 format 228
 - DOLLARw.d format 227
 - DROP statement 157
 - DROP= data set option
 - determining when to specify 46
 - selecting variables 156
- E**
- END statement
 - grouping statements 207
 - end-of-file marker 119
 - engines
 - See* SAS engines
 - error handling 61
 - correcting common errors 60
 - error types 59
 - IF-THEN/ELSE statement 67
 - in DATA step compilation phase 120
 - in DATA step execution phase 121
 - interpreting messages 59
 - invalid option 71
 - resubmitting revised programs 61
 - semicolon errors 68
 - unbalanced quotation marks 69, 70
 - validating or cleaning data 121
 - Excel data
 - See* Microsoft Excel data
 - execution phase (DATA step)
 - diagnosing errors in 121
 - end of processing actions 116, 119
 - end-of-file marker 119
 - initializing variables 115
 - input data 115
 - iterations of DATA step 117
 - match-merge processing 189
 - expressions 145
 - See also* SAS expressions
 - external files
 - reading entire files 44
- F**
- FILENAME statement
 - creating data sets from external files 34
 - naming data sets 44
 - syntax 34
 - filerefs
 - associating with external files 34
 - fully qualified filenames in 34
 - files
 - See* SAS files
 - FIND function
 - examples 292
 - functionality 291
 - syntax 291

FIRST.variable
 examples 132
 FIRSTOBS= system option 83
 FLOOR function 300
 FMTLIB keyword 235
 FOOTNOTE statement
 canceling 97
 examples 96
 modifying 97
 quotation marks in 69
 specifying in list reports 95
 FORMAT procedure
 FMTLIB keyword 235
 functionality 229, 230
 invoking 230
 LIBRARY= option 230
 syntax 230
 VALUE statement 231, 232, 234
 FORMAT statement
 assigning formats to variables 234, 250
 formatting dates 265, 266
 functionality 225
 syntax 225
 formats 19
 assigning permanent 250
 assigning to variables 234, 250
 decimal places 228
 defining unique 231
 examples 228
 field widths 227
 for variables 225
 functionality 225
 permanently assigned 103
 specifying 227
 storing 230
 storing permanently 230
 writing numeric values 312
 forward slash (/)
 specifying multiple delimiters 279
 FRAME= option
 syntax 342
 FREQ procedure
 See also TABLES statement, FREQ
 procedure
 detecting invalid data 121
 producing frequency tables 317, 322,
 324
 specifying variables 322
 suppressing table information 329
 syntax 121, 318
 frequency tables
 creating in list format 326
 n-way 317, 324
 one-way 317, 322
 suppressing information 329
 two-way 324

functions
 arguments 255
 arrays and 255
 character-to-numeric conversions 257
 converting data 256
 manipulating date/time values 241
 syntax 255
 target variables and 256
 variable lists 255

G

group processing
 with BY statement 134, 314
 with CLASS statement 313

H

HIGH keyword 232
 HTML destination 336
 HTML link and path options
 URL= suboption 343
 HTML output
 appearance of HTML 346
 frame files 342
 link and path options 343, 345
 ODS overview 336
 overview 338
 specify link and path 343
 table of contents 340, 342
 HTML table of contents
 CONTENTS= option 342
 hyphen (-) 279
 hypothesis testing 309

I

ID statement, PRINT procedure
 BY statement and 91
 VAR statement and 79
 IF-THEN statement
 assigning values conditionally 152
 cleaning data 123
 DELETE statement in 155
 DO groups 207
 ELSE statement 154
 examples 152
 for flagging errors 67
 syntax 152
 testing programs 65
 IN= data set option 196
 indenting DO groups 215
 INDEX function
 functionality 289
 syntax 289
 index variables in DO loops 213, 216

- informats 19
 - components 243
 - reading dates and times 243
 - initializing variables 115, 148
 - input buffer 117
 - INPUT function
 - character-to-numeric conversion 256, 259
 - examples 260
 - syntax 259
 - INT function 301
 - INTCK function
 - examples 273
 - functionality 263, 273
 - syntax 273
 - INTNX function 263, 274
 - invalid data 121
 - invalid option 71
 - iterative DO statements 218
 - conditional executing 218
 - nesting DO loops 215
- K**
- KEEP statement 157
 - KEEP= data set option
 - determining when to specify 46
 - selecting variables 156
 - KURTOSIS statistic, MEANS procedure 309
- L**
- LABEL option, PRINT procedure 100
 - LABEL statement
 - assigning labels in multiple 101
 - assigning labels in single 102
 - example 101
 - functionality 100
 - syntax 100
 - labels
 - assigning descriptive 100
 - assigning for variables 19
 - assigning permanent 103
 - LAST.variable
 - examples 132
 - LCLM statistic, MEANS procedure 309
 - leading blanks, removing 288
 - LEFT function 286
 - LENGTH statement
 - examples 150
 - functionality 149, 279
 - length, variable 19, 279
 - LIBNAME statement
 - assigning librefs 27
 - defining SAS libraries 26
 - referencing files in other formats 27
 - syntax 26
 - libraries
 - See SAS libraries
 - LIBRARY= option, FORMAT procedure 230
 - librefs
 - assigning 25, 26
 - defined 13
 - lifespan of 27
 - verifying 27
 - LIST option, TABLES statement (FREQ) 326
 - list reports
 - creating 309
 - creating tables for 326
 - formatting data values 225
 - generating column totals 88
 - identifying observations 78
 - selecting observations 310
 - selecting variables 77
 - sorting data 86
 - specifying footnotes 95
 - specifying titles 95
 - logic errors
 - PUTLOG statement 63
 - logical operators
 - in SAS expressions 143
 - LOW keyword 232
 - LOWCASE function 291, 294
- M**
- macro
 - using SAS macro variables 164
 - Markup Languages Family destination 336
 - match-merging data sets 188
 - compilation phase 188
 - examples 185
 - execution phase 189
 - functionality 177, 184
 - handling missing values 191
 - handling unmatched observations 191
 - selecting data 185
 - MAX statistic
 - MEANS procedure 309, 316
 - MAXDEC= option, MEANS procedure 312
 - MDY function
 - examples 270
 - functionality 263
 - missing values 270
 - MEAN statistic
 - MEANS procedure 309, 316
 - MEANS procedure

BY statement 314
 CLASS statement 313
 creating summarized data sets 316
 descriptive statistics 309
 detecting invalid data 121, 122
 functionality 307
 hypothesis testing 309
 keywords supported 309
 limiting decimal places 312
 MAXDEC= option 312
 OUTPUT statement 316
 quantile statistics 309
 selecting statistics 309
 specifying variables 313
 syntax 122, 308
 VAR statement 122, 313
 MEDIAN statistic, MEANS procedure 309
 MERGE statement
 match-merging data sets 177, 184
 RENAME= data set option 195
 RETAIN statement and 148
 syntax 184
 Microsoft Excel data 53
 CONTENTS procedure 48
 creating worksheets 53
 DATA statement 48, 51
 name literals 51
 PRINT procedure 48, 52
 referencing workbooks 49
 RUN statement 48, 52
 SET statement 48
 steps for reading 47
 WHERE statement, DATA step 51
 writing the DATA step 51
 MIN statistic
 MEANS procedure 309, 316
 missing values
 in match-merge processing 191
 MDY function and 270
 overview 21
 MMDDYY10. format 228
 MMDDYY8. format 228
 MMDDYYw. format 227
 MMDDYYw. informat
 examples 243
 functionality 243
 syntax 243
 MODE statistic
 MEANS procedure 309
 MONTH function
 examples 265
 functionality 254
 manipulating date values 264
 syntax 264
 typical use 263

N

N statistic
 MEANS procedure 309, 316
 n-way frequency tables 317, 324
 naming conventions
 for variables 18
 SAS data sets 14
 nesting
 DO groups 215
 DO loops 215
 NMISS statistic
 MEANS procedure 309
 NOCOL option, TABLES statement (FREQ) 330
 NOCUM option, TABLES statement (FREQ) 324
 NOFREQ option, TABLES statement (FREQ) 330
 NOOBS option, PRINT procedure 78
 NOPERCENT option, TABLES statement (FREQ) 330
 NOROW option, TABLES statement (FREQ) 330
 NOT operator 153
 numeric-to-character conversion 256, 261

O

OBS= option, OPTIONS statement 44
 OBS= system option 83
 observations 20
 See also combining data sets
 combining from multiple data sets 177
 creating for DO loop iterations 213
 deleting 155
 identifying 78
 limiting when testing programs 125
 selecting in list reports 80
 selecting matching 197
 specifying via system options 83
 unmatched 191, 196
 writing explicitly 54
 ODS _ALL_ CLOSE statement 337
 ODS destinations 336
 ODS EXCEL destination
 TAGATTR= style 355
 ODS EXCEL statement
 syntax 354
 ODS HTML CLOSE statement
 syntax 337
 ODS HTML statement
 syntax 338
 table of contents syntax 340
 ODS LISTING CLOSE statement 337
 ODS PDF destinations
 open and close statements 348

- table of contents 348
- ODS PDF statement
 - statements 348
 - syntax 347
- ODS RTF
 - RTF formats 353
 - RTF graphics 353
- ODS RTF destinations
 - open and close statements 353
- ODS RTF statement
 - syntax 352
- ODS statements 336
- one-to-one reading of data sets
 - example 181
 - functionality 177, 178, 179
 - selecting data 178
- one-way frequency tables 317, 322
- operands 142
- operators
 - concatenation 261
 - defined 142
 - in SAS expressions 142
 - logical 143
- OR operator
 - examples 153
 - in SAS expressions 143
 - in WHERE statement, PRINT procedure 82
- OTHER keyword 232
- Output Delivery System (ODS)
 - advantages 336
 - EXCEL 354, 355
 - HTML support 338, 340
 - opening and closing destinations 336
 - PDF 347, 348, 351
 - RTF 352, 353
- Output destination 336
- OUTPUT statement
 - creating for DO loop iterations 213
 - functionality 54
 - syntax 54
- OUTPUT statement (MEANS) 316
- OUTPUT statement, MEANS procedure
 - functionality 316
 - syntax 316

P

- P1 statistic, MEANS procedure 309
- P10 statistic, MEANS procedure 309
- P25 statistic, MEANS procedure 309
- P5 statistic, MEANS procedure 309
- P50 statistic, MEANS procedure 309
- P75 statistic, MEANS procedure 309
- P90 statistic, MEANS procedure 309
- P95 statistic, MEANS procedure 309

- P99 statistic, MEANS procedure 309
- parentheses ()
 - for function arguments 255
 - logical comparisons in 153
- PATH= Option
 - syntax 345
- PDV
 - See [program data vector](#)
- period (.)
 - in SAS filenames 13
- PRINT procedure
 - BY statement 89, 91
 - creating list reports 76
 - ID statement 79, 91
 - LABEL option 100
 - NOOBS option 78
 - reading Microsoft Excel data 48, 52
 - SUM statement 88, 89
 - VAR statement 77, 79
 - WHERE statement 80
- printer family destination 336
- PROBT statistic, MEANS procedure 309
- PROC IMPORT
 - OBS= option 44
 - verifying data 42
- PROC step
 - missing RUN statement 67
 - reading external files 44
- PROC TRANSPOSE
 - variables 158
- program data vector 112
 - DATA step processing 112, 117, 118
 - match-merge processing 188, 189, 190, 191
- programming workspace
 - SAS libraries 11
- PROPCASE function 294
- PUT function
 - numeric-to-character conversion 256, 260, 261
 - syntax 262
- PUT statement
 - character strings 66
 - conditional processing and 67
 - data set variables 66
 - syntax 65
 - testing programs 65, 66, 155
- PUTLOG statement 63

Q

- Q1 statistic, MEANS procedure 309
- Q3 statistic, MEANS procedure 309
- QRANGE statistic, MEANS procedure 309
- QTR function

- functionality 254
- manipulating date values 264
- syntax 264
- typical use 263
- quantile statistics 309
- quotation marks
 - common errors 69
 - format names and 232
 - logical operations 153
 - numeric-to-character conversion 261
 - reading Microsoft Excel data 49

R

- RANGE statistic
 - MEANS procedure 309
- RENAME= data set option 195
- renaming variables 194
- RETAIN statement
 - initializing sum variables 148
 - syntax 148
- RIGHT function 286
- ROUND function 302
- rows
 - See* observations
- RTF destination 336
- RUN statement
 - reading Microsoft Excel data 48, 52

S

- SAS engines 28
- SAS expressions 142
 - accumulating totals 147
 - arithmetic operators in 142
 - logical operators in 143
 - specifying compound 82
 - specifying in list reports 81
- SAS files 13
 - naming conventions 14
 - referencing 13
 - referencing in other formats 27
 - storing 12
 - temporary 13
 - two-level names 13, 25, 27
- SAS formats
 - See* formats
- SAS informats
 - See* informats
- SAS libraries 11
 - creating 11
 - defining 11, 25
 - deleting 12
 - storing SAS files 12
 - viewing 28
 - viewing library contents 28
- SAS log 9, 261
 - clearing 61
 - resubmitting revised programs 62
- SAS programs
 - DATA step processing 109
 - error handling 59
 - processing 8
 - resubmitting revised 61
 - results of processing 9
 - SAS log 9
- SAS sessions
 - libref lifespan and 27
- SAS statements
 - See also* assignment statements
 - DO groups 207
 - executing repeatedly 210
- SAS/ACCESS engines 28
- SAS/ACCESS LIBNAME statement
 - naming data sets 44
 - syntax 48
- Sashelp library 11
- Sasuser library 11
- SCAN function
 - functionality 277
 - specifying delimiters 278
 - specifying variable length 279
 - SUBSTR function versus 285
 - syntax 278
- semantic errors 59
- semicolon (;)
 - common errors 68
- SET statement
 - BY statement and 313
 - concatenating data sets 177, 182
 - DATA step processing 115, 117
 - one-to-one reading 177, 178
 - reading data sets 45
 - reading Microsoft Excel data 48, 51
 - RETAIN statement and 148
 - syntax 45
- SKEWNESS statistic, MEANS procedure 309
- SORT procedure
 - BY statement 87, 187
 - examples 86
 - sorting data in list reports 86
 - syntax 86
- sorting data in list reports 86
- statistics
 - quantile 309
 - summary 316
- STD statistic
 - MEANS procedure 309, 316
- STDDEV statistic, MEANS procedure 309
- STDERR statistic

- MEANS procedure 309
- STEP statement 51
- storing
 - formats 230
 - SAS files 12
- strings
 - See [character strings](#)
- STYLE= option
 - syntax 346, 351
- subsetting data 155
- subsetting IF statement
 - examples 151
 - finding year 266
 - functionality 47
 - selecting matching observations 197
 - syntax 151
- SUBSTR function
 - functionality 261, 280
 - positioning 282
 - replacing text 282
 - SCAN function versus 285
 - syntax 280
- subtotaling variables 89, 323
- sum statement
 - accumulating totals 147
 - DO loops and 211
 - syntax 147
- SUM statement, PRINT procedure
 - creating customized layouts 91
 - generating column totals 88
 - requesting subtotals 89
 - syntax 88
- SUM statistic
 - MEANS procedure 309
- sum variables, initializing 148
- summary statistics 316
- SUMWGT statistic
 - MEANS procedure 309
- syntax errors 59
- system options
 - specifying observations 83

T

- T statistic
 - MEANS procedure 309
- TABLES statement, FREQ procedure 121
 - creating n-way tables 324
 - creating one-way tables 322
 - creating two-way tables 324
 - CROSSLIST option 327
 - examples 323
 - LIST option 326
 - NOCOL option 330
 - NOCUM option 324
 - NOFREQ option 330

- NOPERCENT option 330
- NOROW option 330
 - syntax 322, 324
- target variables
 - defined 256
 - missing values and 270
- temporary variables 196, 313
- testing programs
 - character strings 66
 - conditional processing 67, 155
 - data set variables 66
 - hypothesis testing 309
 - limiting observations 125
 - PUT statement 65
- TIME function 263
- TIMEw. format 243
- TIMEw. informat 246
- TITLE statement
 - canceling 97
 - examples 95
 - modifying 97
 - quotation marks in 69
 - specifying in list reports 95
- TODAY function 263
- trailing blanks, removing 286, 287, 288
- TRANWRD function 295
- TRIM function 287
- two-way frequency tables 324

U

- UCLM statistic, MEANS procedure 309
- UNIX environment
 - SAS library implementation 12, 26
 - storing files 12
 - unbalanced quotation marks 69, 70
- unmatched observations
 - excluding 196
 - handling 191
- UPCASE function 291, 293
- UPDATE statement 148
- USS statistic
 - MEANS procedure 309

V

- validating data 121
- VALIDMEMNAME= system option
 - naming conventions 16
- VALIDNARNAME= system option
 - naming conventions 14
- VALUE statement, FORMAT procedure
 - assigning formats to variables 234
 - functionality 231
 - HIGH keyword 232
 - LOW keyword 232

- OTHER keyword 232
- specifying value ranges 232
- syntax 231
- VAR statement
 - MEANS procedure 122, 309, 313, 316
 - PRINT procedure 77, 79
- VAR statistic
 - MEANS procedure 309
- variable attributes
 - data sets 18, 112
 - format considerations 19
- variables 21
 - accumulating totals 147
 - assigning formats 234, 250
 - assigning labels 19, 330
 - assigning values conditionally 152
 - attributes 18, 19
 - creating or modifying 142
 - creating/modifying 147
 - DO groups 207
 - format overview 19
 - functionality 21
 - generating totals 88
 - index 213, 216
 - informat overview 19
 - initializing 115, 148
 - labels for 19
 - length of 19, 279
 - macro 164
 - missing values 21
 - naming conventions 14, 16, 18
 - PROC TRANSPOSE 158
 - renaming 194
 - requesting subtotals 89, 323
 - selecting in list reports 77
 - selecting to drop and keep 156
 - specifying in FREQ procedure 322
 - specifying in MEANS procedure 313
 - specifying lengths 149
 - subsetting data 155
 - sum 148
 - target 256, 270
 - temporary 196, 313
 - testing programs 66
 - types of 19
- W**
 - w. format 227
 - w.d format 227
 - w.d informat 258
 - WEEKDATEw. format 248
 - WEEKDAY function 263, 266
 - WHERE statement, DATA step
 - automatic conversions and 258
 - reading Microsoft Excel data 51
 - WHERE statement, PRINT procedure
 - CONTAINS operator 82
 - examples 82
 - specifying compound expressions 82
 - specifying expressions 81
 - syntax 80
 - Windows environment
 - SAS library implementation 12, 26
 - storing files 12
 - unbalanced quotation marks 69
 - WORDDATEw. format 249
 - Work library 11
 - writing observations explicitly 54
- Y**
 - YEAR function
 - examples 265
 - functionality 254
 - manipulating date values 264
 - syntax 264
 - typical use 263
 - YRDIF function 263, 275
- Z**
 - z/OS environment
 - SAS library implementation 12, 26
 - unbalanced quotation marks 69, 70

Ready to take your SAS® and JMP® skills up a notch?



Be among the first to know about new books,
special events, and exclusive discounts.

support.sas.com/newbooks

Share your expertise. Write a book with SAS.

support.sas.com/publish

 sas.com/books
for additional books and resources.


THE POWER TO KNOW.®

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.
Other brand and product names are trademarks of their respective companies. © 2017 SAS Institute Inc. All rights reserved. M1588358 US.0217

