# Controlling DATA Step Processing

1. Add two PUTLOG statements before the RUN statement to print **"PDV Before RUN Statement"** and write all columns in the PDV to the log. Run the program.

```
putlog "PDV Before RUN statement";
putlog _all_;
run;
```

2. What is the value of **StormLength** at the end of the second iteration of the DATA step?

   **StormLength** is 9.

3. Perform the following tasks:
   a) Modify the DATA step to create three tables: monument, park, and other.
   b) Use the value of ParkType as indicated above to determine which table the row is output to.
   c) Drop ParkType from the monument and park tables.
   d) Drop Region from all three tables.
   e) Submit the program and verify the output. The note in the SAS log indicates how many rows are in each table.

```
data monument(drop=ParkType) park(drop=ParkType) other;
      set pg2.np_yearlytraffic;
      if ParkType = 'National Monument' then output monument;
     else if ParkType = 'National Park' then output park;
     else output other;
     drop Region;
run;
```

4. Create a new program.
   a) Write a DATA step that creates temporary SAS tables named camping and lodging and reads the pg2.np_2017 table.
   b) Compute a new column, CampTotal, that is the sum of CampingOther, CampingTent, CampingRV, and CampingBackcountry.
   c) Format CampTotal so that values are displayed with commas.
   d) The camping table has the following specifications:
      - includes rows if CampTotal is greater than zero
      - contains the ParkName, Month, DayVisits, and CampTotal columns
   e) The lodging table has the following specifications:
      - includes rows where LodgingOther is greater than zero
      - contains only the ParkName, Month, DayVisits, and LodgingOther columns

```
data camping(keep=ParkName Month DayVisits CampTotal)
   lodging(keep=ParkName Month DayVisits LodgingOther);
   set pg2.np_2017;
   CampTotal=sum(of Camping:);
   if CampTotal > 0 then output camping;
   if LodgingOther > 0 then output lodging;
   format CampTotal comma15.;
run;
```

5. Create a program to use SELECT groups and WHEN statements.

```
data monument(drop=ParkType) park(drop=ParkType) other;
   set pg2.np_yearlytraffic;
   select (ParkType);
      when ('National Monument') output monument;
      when ('National Park') output park;
      otherwise output other;
   end;
   drop Region;
run;
```