DO Loop Processing

• Statements within a DO loop execute for a specific number of iterations or until a specific condition stops the loop.

DO Loop Processing

DATA statement
SAS statements
DO statement
iterated SAS statements
END statement
SAS statements
RUN statement

Repetitive Coding

You can use DO loops to
• perform repetitive calculations
• generate data
• eliminate redundant code
• execute SAS code conditionally.

Repetitive Coding

• Compare the interest for yearly versus quarterly compounding on a $50,000 investment made for one year at 7.5 percent interest.

• How much money will a person accrue in each situation?

Repetitive Coding

data compound;
  Amount=50000;
  Rate=.075;
  Yearly=Amount*Rate;
  Q=0;
  Q=Q+((Q+Amount)*Rate/4);
  Q=Q+((Q+Amount)*Rate/4);
  Q=Q+((Q+Amount)*Rate/4);
  Q=Q+((Q+Amount)*Rate/4);
run;

Repetitive Coding

proc print data=compound noobs;
run;

PROC PRINT Output

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>0.075</td>
<td>3750</td>
<td>3856.79</td>
</tr>
</tbody>
</table>

What if you wanted to determine the quarterly compounded interest after a period of 20 years (80 quarters)?
DO Loop Processing

```sas
data compound;
  Amount=50000;
  Rate=.075;
  Yearly=Amount*Rate;
  Q=0;
  do Qtr=1 to 80;
    Q=Q+((Q+Amount)*Rate/4);
  end;
run;
```

Repetitive Coding

```sas
proc print data=compound noobs;
run;
```

PROC PRINT Output

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>0.075</td>
<td>3750</td>
<td>170993.62</td>
<td>81</td>
</tr>
</tbody>
</table>

What if you wanted to determine the quarterly compounded interest after a period of 20 years (80 quarters)?

The Iterative DO Statement

The iterative DO statement executes statements between DO and END statements repetitively, based on the value of an index variable.

```sas
DO index-variable=specifications;
 <additional SAS statements>
END;
```

The Iterative DO Statement : Range

```sas
DO index-variable=start TO stop <BY increment>;
```

• The values of `start`, `stop`, and `increment`
  – must be numbers or expressions that yield numbers
  – are established before executing the loop
  – `increment` is optional, default value is 1
• Any changes to the values of `stop` or `increment` made within the DO loop do not affect the number of iterations.

The Iterative DO Statement

```sas
do i=1 to 12;
 do j=2 to 10 by 2;
 do k=14 to 2 by -2;
 do m=3.6 to 3.8 by .05;
```

The Iterative DO Statement

```sas
DO index-variable=item-1 <...,item-n>;
```

• `item-1` through `item-n` can be either all numeric or all character constants, or they can be variables.
• The DO loop is executed once for each value in the list.
The Iterative DO Statement
How many times will each DO loop execute?

- `do Month='JAN','FEB','MAR';` 3 times.
- `do Fib=1,2,3,5,8,13,21;` 7 times.
- `do i=Var1,Var2,Var3;` 3 times.
- `do j=BeginDate to Today() by 7;` Unknown. The number of iterations depends on the values of BeginDate and Today().
- `do k=Test1-Test50;` 1 time. A single value of k is determined by subtracting Test50 from Test1.

Performing Repetitive Calculations
Compare the interest for yearly versus quarterly compounding on a $50,000 investment made for one year at 7.5 percent interest.

```sas
data compound;
    Amount=50000;
    Rate=.075;
    Yearly=Amount*Rate;
    Q=0;
    do Qtr=1 to 4;
        Q=Q+(Q+Amount)*Rate/4;
    end;
run;
```

---

Execute

```
data compound;
    Amount=50000;
    Rate=.075;
    Yearly=Amount*Rate;
    Q=0;
    do Qtr=1 to 4;
        Q=Q+(Q+Amount)*Rate/4;
    end;
run;
```

INDEX VARIABLE

```
PDV

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
```

Initialize PDV to missing.

```
data compound;
    Amount=50000;
    Rate=.075;
    Yearly=Amount*Rate;
    Q=0;
    do Qtr=1 to 4;
        Q=Q+(Q+Amount)*Rate/4;
    end;
run;
```

```
data compound;
    Amount=50000;
    Rate=.075;
    Yearly=Amount*Rate;
    Q=0;
    do Qtr=1 to 4;
        Q=Q+(Q+Amount)*Rate/4;
    end;
run;
```

```
<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
```

```
data compound;
    Amount=50000;
    Rate=.075;
    Yearly=Amount*Rate;
    Q=0;
    do Qtr=1 to 4;
        Q=Q+(Q+Amount)*Rate/4;
    end;
run;
```

```
data compound;
    Amount=50000;
    Rate=.075;
    Yearly=Amount*Rate;
    Q=0;
    do Qtr=1 to 4;
        Q=Q+(Q+Amount)*Rate/4;
    end;
run;
```

```
<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
```
data compound;
Amount=50000;
Rate=.075;
Yearly=Amount*Rate;
Q=0;
do Qtr=1 to 4;
Q=Q+((Q+Amount)*Rate/4);
end;
run;

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>937.50</td>
<td>1</td>
</tr>
</tbody>
</table>

Is Qtr out of range?

PDV

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>937.50</td>
<td>2</td>
</tr>
</tbody>
</table>

935.70 + ((935.70+50000) * .075/4)

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>1892.58</td>
<td>3</td>
</tr>
</tbody>
</table>

Is Qtr out of range?

PDV

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>1892.58</td>
<td>3</td>
</tr>
</tbody>
</table>
```plaintext
data compound;
  Amount=50000;
  Rate=.075;
  Yearly=Amount*Rate;
  Q=0;
  do Qtr=1 to 4;
    Q=Q+((Q+Amount)*Rate/4);
  end;
run;
```

### Table

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>0.075</td>
<td>3750</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1892.58</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PDV**

Is Qtr out of range?

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>0.075</td>
<td>3750</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>2865.56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PDV**

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>0.075</td>
<td>3750</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>3856.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PDV**

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>0.075</td>
<td>3750</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>2865.56</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**PDV**

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>0.075</td>
<td>3750</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>3856.79</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
data compound;
Amount=50000;
Rate=.075;
Yearly=Amount*Rate;
Q=0;
do Qtr=1 to 4;
Q=Q+((Q+Amount)*Rate/4);
end;
run;

PDV

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>3856.79</td>
<td>5</td>
</tr>
</tbody>
</table>

Performing Repetitive Calculations

Proc print data=compound noobs;
Run;

PROC PRINT Output

<table>
<thead>
<tr>
<th>Amount</th>
<th>Rate</th>
<th>Yearly</th>
<th>Q</th>
<th>Qtr</th>
</tr>
</thead>
<tbody>
<tr>
<td>50000</td>
<td>.075</td>
<td>3750</td>
<td>3856.79</td>
<td>5</td>
</tr>
</tbody>
</table>

Using SAS Arrays

Array Processing

You can use arrays to simplify programs
- perform repetitive calculations
- create many variables with the same attributes
- compare variables

Performing Repetitive Calculations

Records that contain 5 ICD-9 diagnostic codes. You want to be able to assign the string “unk” to the unknown values

- Partial Listing of loop

<table>
<thead>
<tr>
<th>DX1</th>
<th>DX2</th>
<th>DX3</th>
<th>DX4</th>
<th>DX5</th>
</tr>
</thead>
<tbody>
<tr>
<td>9674</td>
<td>E8523</td>
<td>30390</td>
<td>3099</td>
<td></td>
</tr>
<tr>
<td>9779</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>V300</td>
<td>7651</td>
<td>9608</td>
<td>E856</td>
<td></td>
</tr>
<tr>
<td>9695</td>
<td>E9503</td>
<td>311</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Performing Repetitive Calculations

```sas
DATA example; set mylib.loop;
n=0; *counter;
if dx1=' ' then do;
   n=n+1; *increase counter;
   dx1='unk'; *value changes;
   dy1=0; *new variable;
end;
else dy1=1;
Etc for dx2 dx3 dx4 and dx5
Run;
```

What Is a SAS Array?

A SAS array
– is a temporary grouping of SAS variables that are arranged in a particular order
– is identified by an array name
– exists only for the duration of the current DATA step
– is not a variable.

The ARRAY Statement

The ARRAY statement defines the elements in an array. These elements will be processed as a group. You refer to elements of the array by the array name and subscript. Use { or (

```sas
array Alldx{5} dx1 dx2 dx3 dx4 dx5;
```

What Is a SAS Array?

• Each value in an array is
  – called an element
  – identified by a subscript that represents the position of the element in the array

• When you use an array reference, the corresponding value is substituted for the reference.

Example: ALLDX(1) refers to DX1
The ARRAY Statement
The ARRAY statement
– must contain all numeric or all character elements
– must be used to define an array before the array name can be referenced
– creates variables if they do not already exist in the PDV
– is a compile-time statement.

Defining an Array
array Alldx{5} dx1 dx2 dx3 dx4 dx5;
Write an ARRAY statement that defines the five diagnosis variables as elements of an array.

Processing an Array
Array processing often occurs within DO loops. An iterative DO loop that processes an array has the following form:

DO index-variable=1 TO number-of-elements-in-array;
  additional SAS statements
  using array-name{index-variable}...
END;

To execute the loop as many times as there are elements in the array, specify that the values of index-variable range from 1 to number-of-elements-in-array.

Example : Array and Loop
DATA example; set mylib.loop;
array alldx (5)$dx1 dx2 dx3 dx4 dx5;
do i=1 to 5;
  if alldx(i)=' ' then alldx(i)='unk';
end;

Performing Repetitive Calculations
PROC PRINT DATA=mylib.loop2 noobs;
var n dx1 dx2 dx3 dx4 dx5;
format dx1-dx5 $5.;

•Partial PROC PRINT Output
<table>
<thead>
<tr>
<th></th>
<th>DX1</th>
<th>DX2</th>
<th>DX3</th>
<th>DX4</th>
<th>DX5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>9674</td>
<td>EB523</td>
<td>30390</td>
<td>3099</td>
<td>unk</td>
</tr>
<tr>
<td>2</td>
<td>9779</td>
<td>unk</td>
<td>unk</td>
<td>unk</td>
<td>unk</td>
</tr>
<tr>
<td>3</td>
<td>V300</td>
<td>7551</td>
<td>9608</td>
<td>EB56</td>
<td>unk</td>
</tr>
<tr>
<td>4</td>
<td>9695</td>
<td>9503</td>
<td>311</td>
<td>unk</td>
<td>unk</td>
</tr>
<tr>
<td>5</td>
<td>5070</td>
<td>981</td>
<td>EB621</td>
<td>unk</td>
<td>unk</td>
</tr>
<tr>
<td>6</td>
<td>96509</td>
<td>9500</td>
<td>unk</td>
<td>unk</td>
<td>unk</td>
</tr>
</tbody>
</table>
Example: Loop.SAS

```sas
DATA mylib.loop2; set mylib.loop;
array alldx (5)$dx1 dx2 dx3 dx4 dx5;
array alldy (5) dy1 dy2 dy3 dy4 dy5;
n=0;
do i=1 to 5;
   if alldx(i)='' then do;
      n=n+1;
      alldx(i)='unk';
   end;
   else alldy(i)=1;
end;
```

Performing Repetitive Calculations

```sas
PROC PRINT DATA=mylib.loop2 noobs;
var n dy1-dy5 i ;
format dy1-dy5 2. i n 1. ;
```

<table>
<thead>
<tr>
<th>n</th>
<th>dy1</th>
<th>dy2</th>
<th>dy3</th>
<th>dy4</th>
<th>dy5</th>
<th>i</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>.</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>.</td>
<td>.</td>
<td>6</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>.</td>
<td>.</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>1</td>
<td>.</td>
<td>.</td>
<td>.</td>
<td>6</td>
</tr>
</tbody>
</table>