CASE STUDY

thirteen

Capital Budgeting

case study

OVERVIEW

CS13.1 Application Overview and Model Development

In this application, the user can consider which of several possible future products would be most beneficial to invest in. This problem is typically modeled as an integer-programming problem in which binary variables determine which products to invest in and which not to invest in. However, in this application, we seek to solve this problem using simulation. For each run of the simulation, we calculate the expected total profit and NPV for several products with given input values. We then summarize which product can be expected to perform the best and suggest that the user invest in it.

CS13.1.1 Model Definition and Assumptions

This application has both deterministic and stochastic input. The stochastic input includes the sales volume, the product development cost, and the time horizon. Since these values are unknown, we prompt the user for the best case, worst case, and most likely values they have. A triangular random variable analyzes the values that may occur between the best case and worst case values. For this application, we want to generate a random number from the triangular distribution for a given probability of this input. We therefore model the inverse triangular distribution function; for a best case, *a*, worst case, *b*, and most likely value, *m*, and a given probability, *p*, the formula is:

$$a \le m \le b$$

$$F^{-1}(p) = a + \left(\frac{(b-a)(m-a)p}{2}\right), \text{ if } 0 \le p \le \frac{m-a}{b-a}$$

$$F^{-1}(p) = b - \left(\frac{(b-a)(b-m)(1-p)}{2}\right), \text{ if } \frac{m-a}{b-a} \le p \le 1$$

The *Triang* function procedure takes as input the best case, worst case, and most likely values and outputs a random number that is found from this triangular distribution for a given probability.

```
Function Triang(Best, Worst, MostLikely)    'triangular distribution function
  Dim p As Double
  p = Rnd()
  If p < (MostLikely - Worst) / (Best - Worst) Then
    Triang = Abs(Worst + ((Best - Worst) * (MostLikely - Worst) * p) / 2)
  Else If p > (MostLikely - Worst) / (Best - Worst) Then
    Triang = Abs(Best - ((Best - Worst) * (Best - MostLikely) * (1 - p)) / 2)
  End If
End Function
```

We use this function to create values for this stochastic input during the simulation runs. In each run, these values are updated and placed on a simulation calculation worksheet that contains several formulas. The formulas are therefore updated with these new values, and the run results are recorded.

```
  For runs = 1 To NumRuns
    For i = 1 To NumProd
      'calculate unknown values using Triang function and put in table
'PRODUCT COST
```

```
        Range("Prod" & i & "Start").Offset(7, 1).Value = Triang(ProdCost(i, 1), ProdCost(i,
          2), ProdCost(i, 3))

        'TIME HORIZON
        NumYears = Triang(Time(i, 1), Time(i, 2), Time(i, 3))
        Range(Range("Prod" & i & "NPV").Offset(-1, 0), Range("Prod" & i & "NPV") _
            .Offset(-1, NumYears)).Name = "Prod" & i & "Time"

        'SALES VOLUMES
        For year = 1 To NumYears
          Range("Prod" & i & "Start"). Offset(6, year + 1).Value = Triang (Sales(i, 1),
          Sales(i, 2), Sales(i, 3))
        Next
        Range(Range("Prod" & i & "Start").Offset(6, NumYears + 2), _
            Range("Prod" & i & "Start").Offset(6, 11)).ClearContents
                    'record NPV and profit to SimDetails
        Range("DetailsStart").Offset(runs + 1, 2 * i - 1).Value = Range("Prod" & i &
       "NPV").Value
        Range("DetailsStart").Offset(runs + 1, 2 * i - 1).Offset(0, 1).Value = _
        Application.WorksheetFunction.Sum(Range(Range("Prod" & i & "NPV").Offset(-2,
              1), _Range("Prod" & i & "NPV").Offset(-2, NumYears)))
    Next i
Next
```

The simulation calculations seek to determine the total profit and net present value (NPV) of each product given its input values. The deterministic input values include the tax rate, the discount rate, the labor cost, the variable cost, and the selling price. Using these values as well as the stochastic values calculated for each run, the application calculates the depreciation, the before tax profit, the after tax profit, the cash flow, and the NPV. The simple formulas for these calculations appear in the table below; the NPV Excel function calculates the NPV.

Range(“Prod” & i & “Start”) **Product Name**	**0**	**1… (to 10)**
Tax Rate	(from input sheet)	
Discount Rate	(from input sheet)	
Labor Cost	(from input sheet)	
Variable Cost	(from input sheet)	
Selling Price	(from input sheet)	
Sales Volume		(from *Triang* function)
Product Development Cost	(from *Triang* function)	
Depreciation		Prod. Devel. Cost / 5
Before tax profit		Labor Cost + Selling Price * Sales Vol – Variable Cost * Sales Vol – Depreciation
After tax profit		((1-Tax Rate) * Before Tax Profit
Cash flow	***Range(“Prod” & i & “Time”)…*** Prod. Devel. Cost	***Range(“Prod1Time”) …*** After Tax Profit + Depreciation
NPV	***Range(“Prod” & i & “NPV”)*** (NPVfunction using Discount Rate and Time Horizon)	

Please refer to *Practical Management Science* by Winston and Albright as well as *Introduction to Probability Models* by Winston for more details.

CS13.1.2 Input

This application requires two types of input: deterministic and stochastic. The deterministic input is:

- Tax rate
- Discount rate
- Labor cost
- Variable cost
- Selling price
- Desired NPV

The stochastic input for the application is:

- Sales volume (best, worst, and most likely)
- Product development cost (best, worst, and most likely)
- Time horizon (best, worst, and most likely)

CS13.1.3 Output

This application has two types of output: output per product and overall, or comparative, output. The output per product is:

- Depreciation
- Before tax profit
- After tax profit
- Cash flow
- Actual NPV

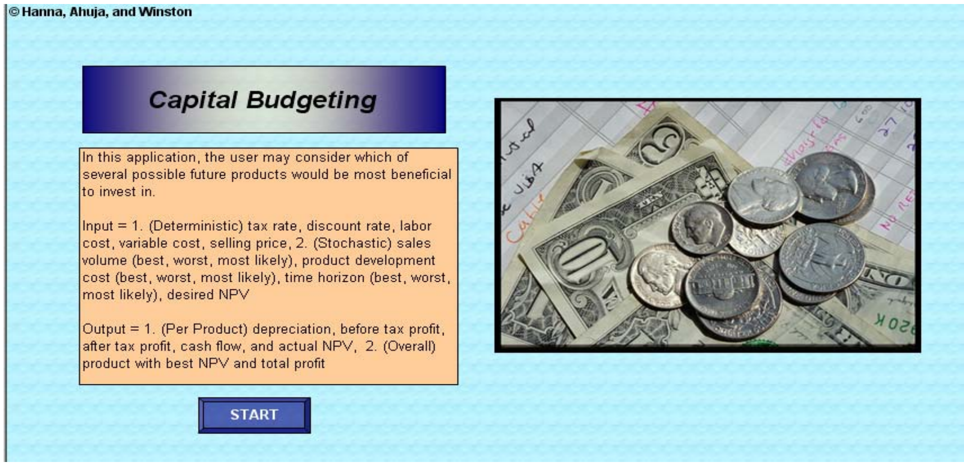
The overall output is:

- Product with best average NPV
- Product with best average total profit

CS13.2 *Worksheets*

This application requires five worksheets: the welcome sheet, the input sheet, the simulation calculations sheet, the simulation details sheet, and the results sheet. The welcome sheet contains the title, the description of the application, and the “Start” button. (See Figure CS13.1.) The “Start” button on the welcome sheet prompts the user for the input values and then displays the input sheet.

The input sheet stores the input values for each product entered by the user. (See Figure CS13.2.) These input values are recorded from a user form, which we will describe in the next section. This sheet stores the deterministic values for the labor cost, the variable cost, the selling price, the tax rate, the discount rate, and the desired NPV. The stochastic input is recorded by best, worst, and most likely values for the sales volume, production development cost, and time horizon.



© Hanna, Ahuja, and Winston

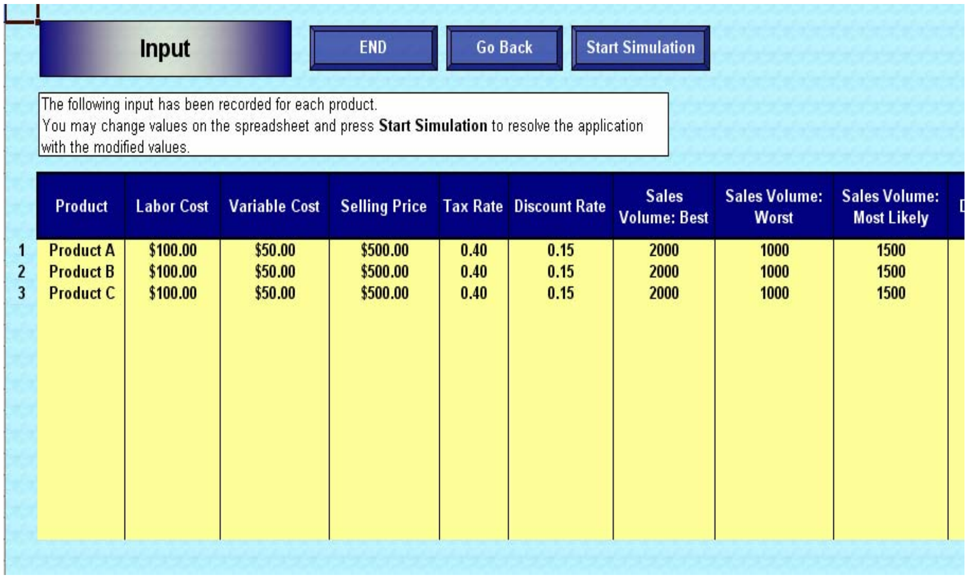
Capital Budgeting

In this application, the user may consider which of several possible future products would be most beneficial to invest in.

Input = 1. (Deterministic) tax rate, discount rate, labor cost, variable cost, selling price, 2. (Stochastic) sales volume (best, worst, most likely), product development cost (best, worst, most likely), time horizon (best, worst, most likely), desired NPV

Output = 1. (Per Product) depreciation, before tax profit, after tax profit, cash flow, and actual NPV, 2. (Overall) product with best NPV and total profit

Figure CS13.1 The welcome sheet.



Input

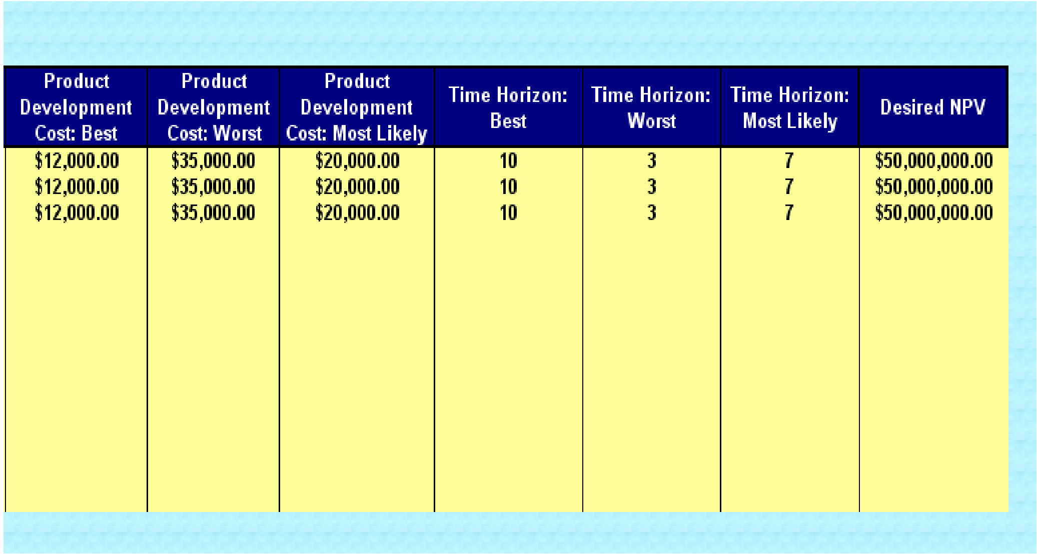
The following input has been recorded for each product.
You may change values on the spreadsheet and press **Start Simulation** to resolve the application with the modified values.

	Product	Labor Cost	Variable Cost	Selling Price	Tax Rate	Discount Rate	Sales Volume: Best	Sales Volume: Worst	Sales Volume: Most Likely	D
1	Product A	$100.00	$50.00	$500.00	0.40	0.15	2000	1000	1500	
2	Product B	$100.00	$50.00	$500.00	0.40	0.15	2000	1000	1500	
3	Product C	$100.00	$50.00	$500.00	0.40	0.15	2000	1000	1500	

Figure CS13.2 (a)

In the simulation calculations sheet, the application calculates the simulation formulas to determine each product's total profit and NPV. (See Figure CS13.3.) The deterministic input values are referenced from this sheet as input for the formulas for depreciation, before tax profit, after tax profit, cash flow, and NPV. The stochastic input values are updated for each run of the simulation using the triangular distribution function with the best, worst, and most likely parameters provided by the user. As these values are updated for each run of the simulation, the formula results are also updated and recorded on the simulation details sheet.

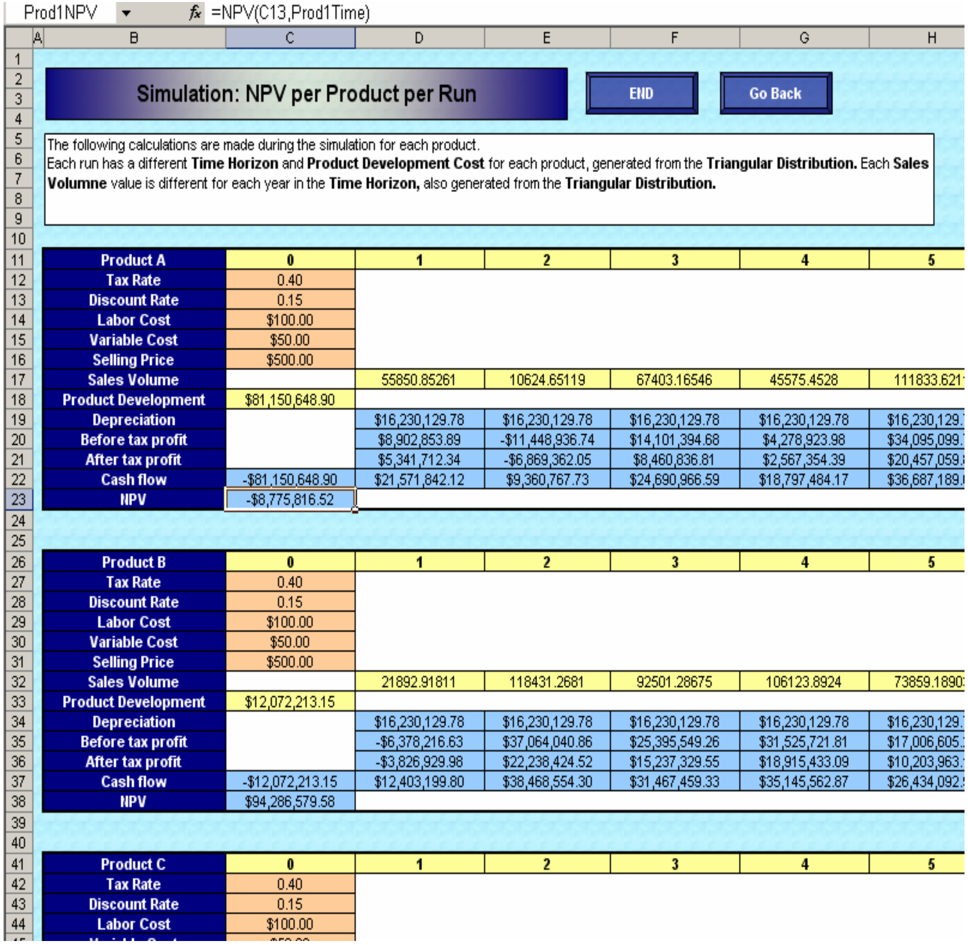
The simulation details sheet records the NPV and total profit values for each product for each run of the simulation. (See Figure CS13.4.) The application employs these values to determine each product's minimum, maximum, and average NPV and profit values, which are stored on the results sheet.



Product Development Cost: Best	Product Development Cost: Worst	Product Development Cost: Most Likely	Time Horizon: Best	Time Horizon: Worst	Time Horizon: Most Likely	Desired NPV
$12,000.00	$35,000.00	$20,000.00	10	3	7	$50,000,000.00
$12,000.00	$35,000.00	$20,000.00	10	3	7	$50,000,000.00
$12,000.00	$35,000.00	$20,000.00	10	3	7	$50,000,000.00

Figure CS13.2 (b)

Figure CS13.2 The input sheet.



Simulation: NPV per Product per Run

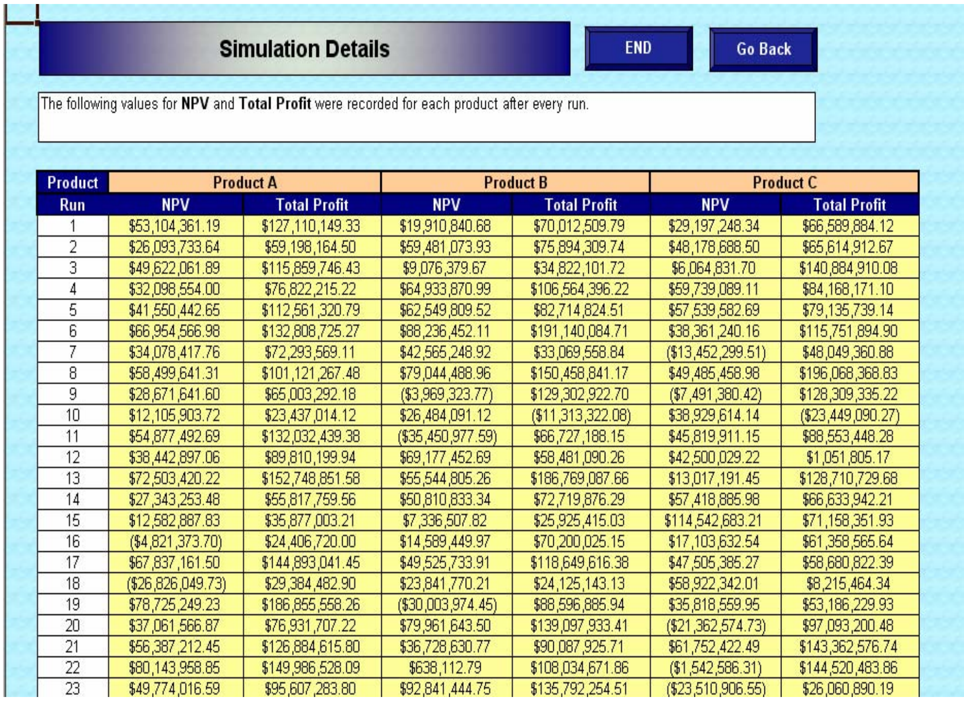
The following calculations are made during the simulation for each product.
Each run has a different **Time Horizon** and **Product Development Cost** for each product, generated from the **Triangular Distribution.** Each **Sales Volumne** value is different for each year in the **Time Horizon,** also generated from the **Triangular Distribution.**

Product A	0	1	2	3	4	5
Tax Rate	0.40					
Discount Rate	0.15					
Labor Cost	$100.00					
Variable Cost	$50.00					
Selling Price	$500.00					
Sales Volume		55850.85261	10624.65119	67403.16546	45575.4528	111833.621
Product Development	$81,150,648.90					
Depreciation		$16,230,129.78	$16,230,129.78	$16,230,129.78	$16,230,129.78	$16,230,129.
Before tax profit		$8,902,853.89	-$11,448,936.74	$14,101,394.68	$4,278,923.98	$34,095,099.
After tax profit		$5,341,712.34	-$6,869,362.05	$8,460,836.81	$2,567,354.39	$20,457,059.
Cash flow	-$81,150,648.90	$21,571,842.12	$9,360,767.73	$24,690,966.59	$18,797,484.17	$36,687,189.
NPV	-$8,775,816.52					

Product B	0	1	2	3	4	5
Tax Rate	0.40					
Discount Rate	0.15					
Labor Cost	$100.00					
Variable Cost	$50.00					
Selling Price	$500.00					
Sales Volume		21892.91811	118431.2681	92501.28675	106123.8924	73859.1890
Product Development	$12,072,213.15					
Depreciation		$16,230,129.78	$16,230,129.78	$16,230,129.78	$16,230,129.78	$16,230,129.
Before tax profit		-$6,378,216.63	$37,064,040.86	$25,395,549.26	$31,525,721.81	$17,006,605.
After tax profit		-$3,826,929.98	$22,238,424.52	$15,237,329.55	$18,915,433.09	$10,203,963.
Cash flow	-$12,072,213.15	$12,403,199.80	$38,468,554.30	$31,467,459.33	$35,145,562.87	$26,434,092.
NPV	$94,286,579.58					

Product C	0	1	2	3	4	5
Tax Rate	0.40					
Discount Rate	0.15					
Labor Cost	$100.00					

Figure CS13.3 The simulation calculations sheet.

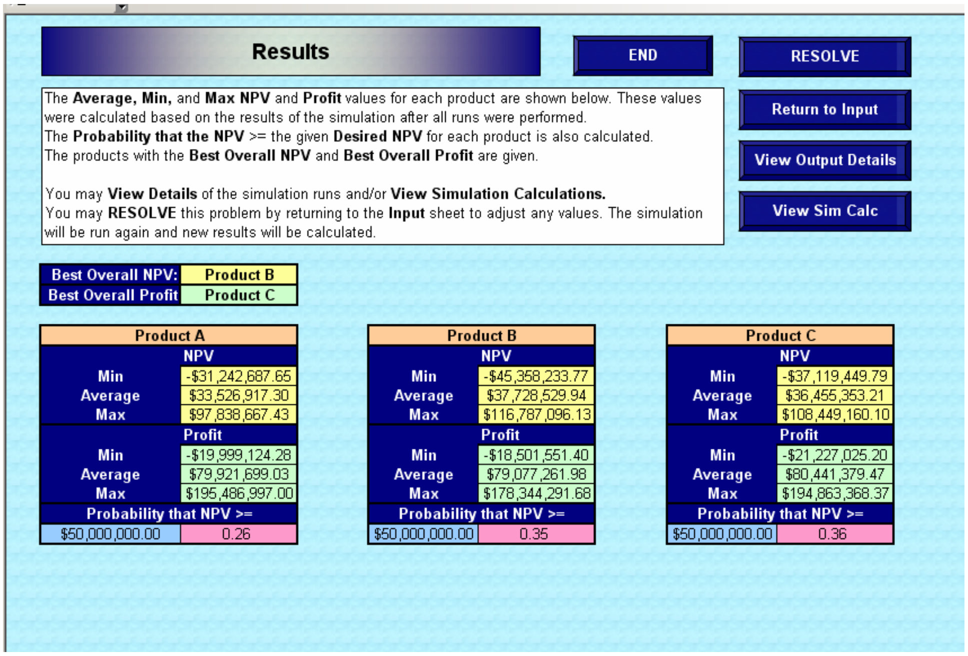


The following values for **NPV** and **Total Profit** were recorded for each product after every run.

Product	Product A		Product B		Product C	
Run	NPV	Total Profit	NPV	Total Profit	NPV	Total Profit
1	$53,104,361.19	$127,110,149.33	$19,910,840.68	$70,012,509.79	$29,197,248.34	$66,589,884.12
2	$26,093,733.64	$59,198,164.50	$59,481,073.93	$75,894,309.74	$48,178,688.50	$65,614,912.67
3	$49,622,061.89	$115,859,746.43	$9,076,379.67	$34,822,101.72	$6,064,831.70	$140,884,910.08
4	$32,098,554.00	$76,822,215.22	$64,933,870.99	$106,564,396.22	$59,739,089.11	$84,168,171.10
5	$41,550,442.65	$112,561,320.79	$62,549,809.52	$82,714,824.51	$57,539,582.69	$79,135,739.14
6	$66,954,566.98	$132,808,725.27	$88,236,452.11	$191,140,084.71	$38,361,240.16	$115,751,894.90
7	$34,078,417.76	$72,293,569.11	$42,565,248.92	$33,069,558.84	($13,452,299.51)	$48,049,360.88
8	$58,499,641.31	$101,121,267.48	$79,044,488.96	$150,458,841.17	$49,485,458.98	$196,068,368.83
9	$28,671,641.60	$65,003,292.18	($3,969,323.77)	$129,302,922.70	($7,491,380.42)	$128,309,335.22
10	$12,105,903.72	$23,437,014.12	$26,484,091.12	($11,313,322.08)	$38,929,614.14	($23,449,090.27)
11	$54,877,492.69	$132,032,439.38	($35,450,977.59)	$66,727,188.15	$45,819,911.15	$88,553,448.28
12	$38,442,897.06	$89,810,199.94	$69,177,452.69	$58,481,090.26	$42,500,029.22	$1,051,805.17
13	$72,503,420.22	$152,748,851.58	$55,544,805.26	$186,769,087.66	$13,017,191.45	$128,710,729.68
14	$27,343,253.48	$55,817,759.56	$50,810,833.34	$72,719,876.29	$57,418,885.98	$66,633,942.21
15	$12,582,887.83	$35,877,003.21	$7,336,507.82	$25,925,415.03	$114,542,683.21	$71,158,351.93
16	($4,821,373.70)	$24,406,720.00	$14,589,449.97	$70,200,025.15	$17,103,632.54	$61,358,565.64
17	$67,837,161.50	$144,893,041.45	$49,525,733.91	$118,649,616.38	$47,505,385.27	$58,680,822.39
18	($26,826,049.73)	$29,384,482.90	$23,841,770.21	$24,125,143.13	$58,922,342.01	$8,215,464.34
19	$78,725,249.23	$186,855,558.26	($30,003,974.45)	$88,596,885.94	$35,818,559.95	$53,186,229.93
20	$37,061,566.87	$76,931,707.22	$79,961,643.50	$139,097,933.41	($21,362,574.73)	$97,093,200.48
21	$56,387,212.45	$126,884,615.80	$36,728,630.77	$90,087,925.71	$61,752,422.49	$143,362,576.74
22	$80,143,958.85	$149,986,528.09	$638,112.79	$108,034,671.86	($1,542,586.31)	$144,520,483.86
23	$49,774,016.59	$95,607,283.80	$92,841,444.75	$135,792,254.51	($23,510,906.55)	$26,060,890.19

Figure CS13.4 The simulation details sheet.

The results sheet displays each product's minimum, maximum, and average NPV and profit values in a table. (See Figure CS13.5.) It also uses the simulation details sheet values to calculate the probability that the NPV will be greater than or equal to the desired NPV provided by the user. The comparative, or overall, output shows which product has the largest average NPV and profit values.



The **Average, Min,** and **Max NPV** and **Profit** values for each product are shown below. These values were calculated based on the results of the simulation after all runs were performed.
The **Probability that the NPV** >= the given **Desired NPV** for each product is also calculated.
The products with the **Best Overall NPV** and **Best Overall Profit** are given.

You may **View Details** of the simulation runs and/or **View Simulation Calculations.**
You may **RESOLVE** this problem by returning to the **Input** sheet to adjust any values. The simulation will be run again and new results will be calculated.

Best Overall NPV:	Product B
Best Overall Profit	Product C

Product A	
	NPV
Min	-$31,242,687.65
Average	$33,526,917.30
Max	$97,838,667.43
	Profit
Min	-$19,999,124.28
Average	$79,921,699.03
Max	$195,486,997.00
Probability that NPV >=	
$50,000,000.00	0.26

Product B	
	NPV
Min	-$45,358,233.77
Average	$37,728,529.94
Max	$116,787,096.13
	Profit
Min	-$18,501,551.40
Average	$79,077,261.98
Max	$178,344,291.68
Probability that NPV >=	
$50,000,000.00	0.35

Product C	
	NPV
Min	-$37,119,449.79
Average	$36,455,353.21
Max	$108,449,160.10
	Profit
Min	-$21,227,025.20
Average	$80,441,379.47
Max	$194,863,368.37
Probability that NPV >=	
$50,000,000.00	0.36

Figure CS13.5 The results sheet.

Summary

Welcome sheet Contains the application description and the “Start” button.

Input sheet Stores the deterministic and stochastic input values for each product.

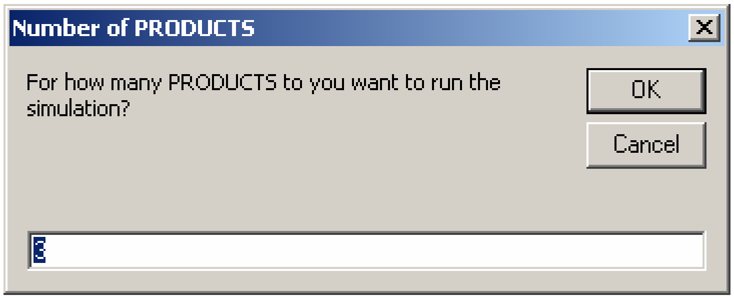
Simulation calculations sheet Calculates the simulation formulas to find the NPV and profit values for each product. Formulas and results are updated for new stochastic values for each run of the simulation.

Simulation details sheet Records the NPV and profit values calculated for each product for each run of the simulation.

Results sheet Shows the minimum, maximum, and average NPV and profit values for each product, the probability that the NPV is greater than or equal to the desired NPV for each product, and the product with the best overall NPV and profit values.

CS13.3 *User Interface*

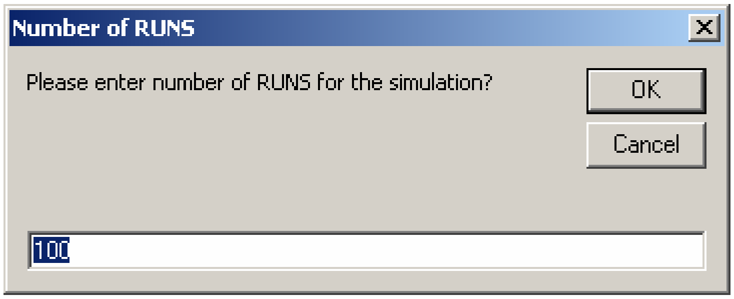
For this application’s user interface, we include navigational buttons, input boxes, and a user form. From the “Start” button on the welcome sheet, the user first views two input boxes. The first prompts the user for the number of products to compare. (See Figure CS13.6.) The second input box then prompts the user for the number of simulation runs. (See Figure CS13.7.)


Number of PRODUCTS

For how many PRODUCTS to you want to run the simulation?

3

Figure CS13.6 The input box for the number of products.

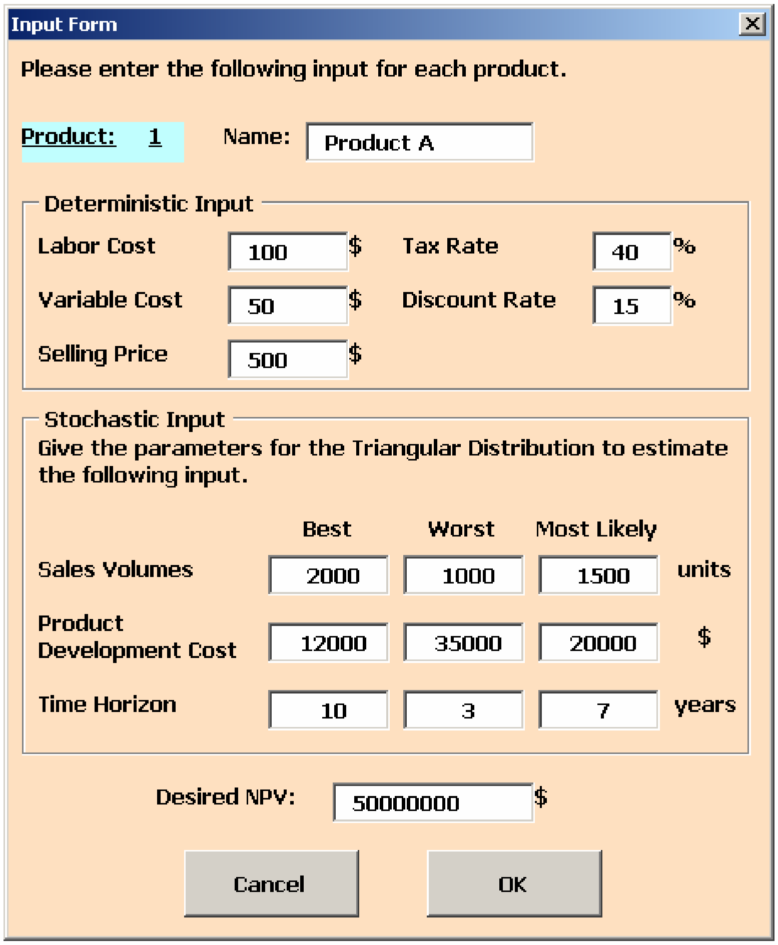

Number of RUNS

Please enter number of RUNS for the simulation?

100

Figure CS13.7 The input box for the number of runs.

Then, the user views an input form for each product he or she is comparing. (See Figure CS13.8.) This form prompts the user for the deterministic and stochastic input; each set of input values is grouped into a respective frame. Text boxes are used for all the data, and the stochastic input text boxes are arranged in a tabular format since each requires a best, worst, and most likely value.



Please enter the following input for each product.

Product: 1 Name: Product A

Deterministic Input

Labor Cost	100 $	Tax Rate	40 %
Variable Cost	50 $	Discount Rate	15 %
Selling Price	500 $		

Stochastic Input

Give the parameters for the Triangular Distribution to estimate the following input.

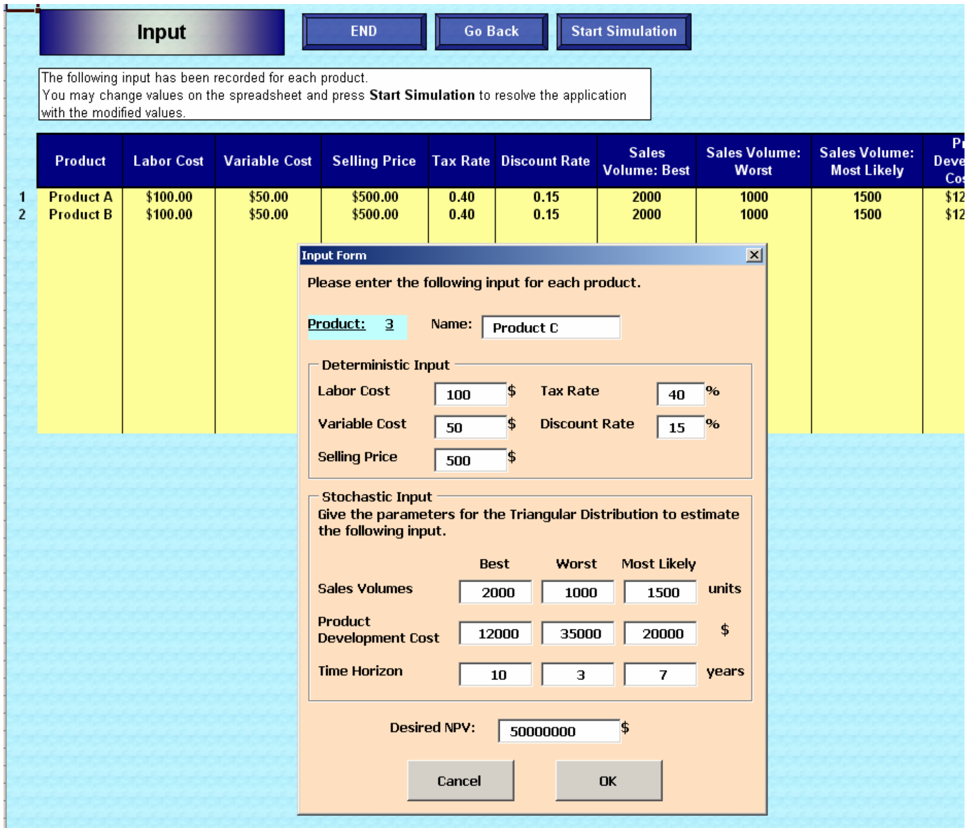
	Best	Worst	Most Likely	
Sales Volumes	2000	1000	1500	units
Product Development Cost	12000	35000	20000	$
Time Horizon	10	3	7	years

Desired NPV: 50000000 $

Figure CS13.8 The input form.

This same form is shown to the user for each product; therefore, the product number label is updated for each product shown. Default values are provided for all the input, including the product name. As the user enters the input values for each product, they are recorded to the input sheet, which appears in the background. For example, in Figure CS13.9, the input values have already been entered and recorded for the first two products on the input sheet. The input form now prompts the user for the input values of the third product.

Summary	**Input boxes**	For the number of products and the number of runs.
	Input form	For the deterministic and stochastic input values; shown for each product; the values are recorded on the input sheet.
	Navigational buttons	“Start” on the welcome sheet; “Go Back” on the input, simulation calculations, and details sheets; “Return to Input,” “View Output Details,” and “View Sim Calc” on the results sheet; and “End” on all the sheets.
	Functional buttons	“Start Simulation” on the input sheet and “Re-solve” on the results sheet.



The following input has been recorded for each product.
You may change values on the spreadsheet and press **Start Simulation** to resolve the application with the modified values.

	Product	Labor Cost	Variable Cost	Selling Price	Tax Rate	Discount Rate	Sales Volume: Best	Sales Volume: Worst	Sales Volume: Most Likely	Pr Deve Co
1	Product A	$100.00	$50.00	$500.00	0.40	0.15	2000	1000	1500	$12
2	Product B	$100.00	$50.00	$500.00	0.40	0.15	2000	1000	1500	$12

Input Form

Please enter the following input for each product.

Product: 3 Name: Product C

Deterministic Input

Labor Cost	100 $	Tax Rate	40 %
Variable Cost	50 $	Discount Rate	15 %
Selling Price	500 $		

Stochastic Input

Give the parameters for the Triangular Distribution to estimate the following input.

	Best	Worst	Most Likely	
Sales Volumes	2000	1000	1500	units
Product Development Cost	12000	35000	20000	$
Time Horizon	10	3	7	years

Desired NPV: 50000000 $

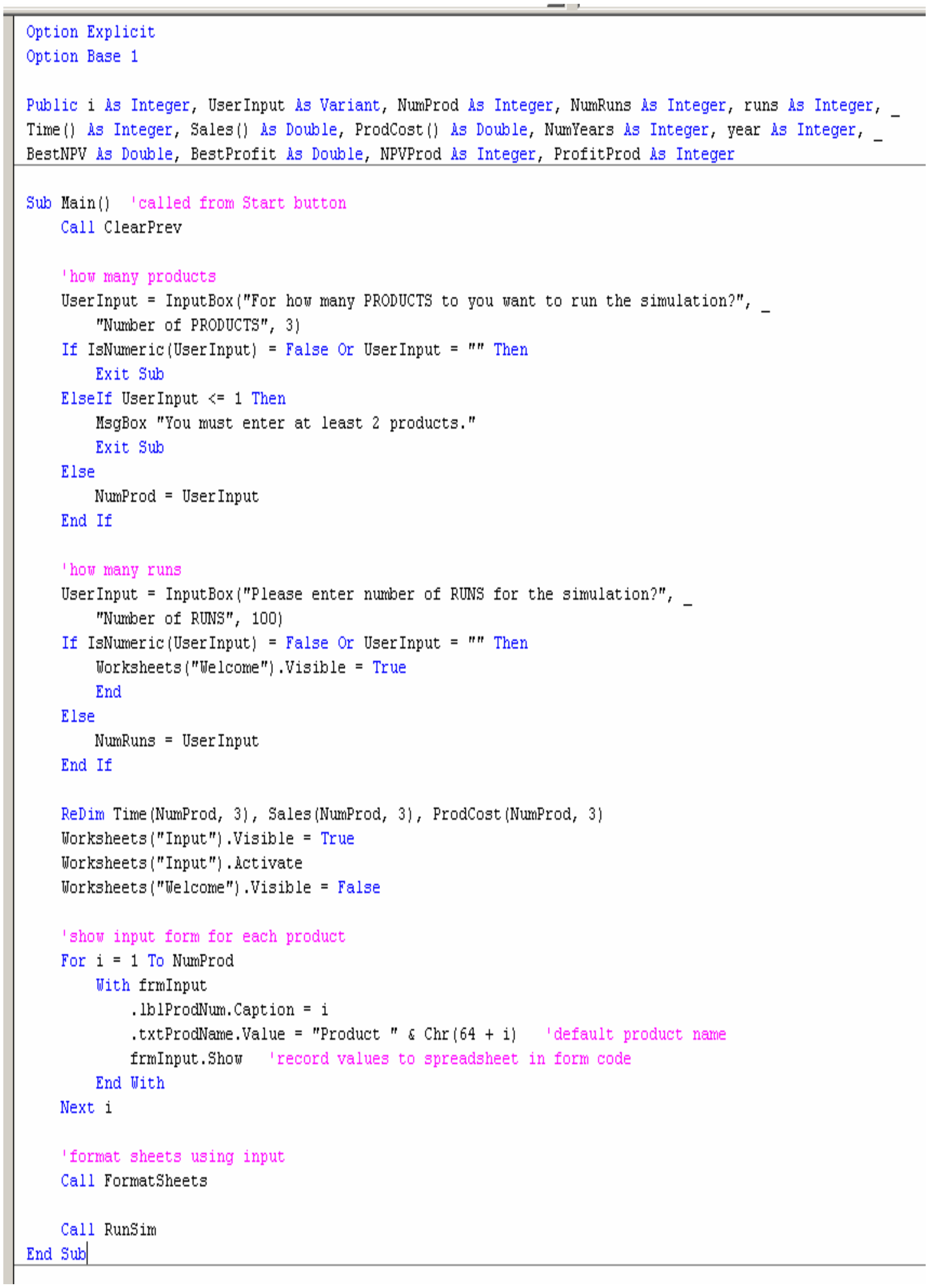
Figure CS13.9 The input form appears in front of the input sheet.

CS13.4 *Procedures*

We will now outline the procedures for this application beginning with the variable declarations and initial sub procedures. (See Figure CS13.10.) The *Main* procedure is called from the "Start" button. It calls the *ClearPrev* procedure to clear previous values and tables from all the sheets. It then prompts the user with the two input boxes for the number of products and the number of runs for the simulation. Notice that the error checking code is used for both of these input boxes. *ClearPrev* then uses the number of products to re-dimension some arrays and takes the user to the input sheet. Once the input sheet appears, the *Main* procedure performs a loop to display the input form for each product. The product number label on the form is updated in each loop along with the default value of the product name.

The input form procedures record all of the input values on the input sheet. (See Figure CS13.11.) Notice that the variable *i* used in the loop in the *Main* procedure to show the input form is used in the input form procedures to determine on which row on the input sheet values will be recorded. This variable also indexes the arrays to store the stochastic input parameters. This code also performs some error checking to ensure that the value for the time horizon worst case input is less than or equal to 10.

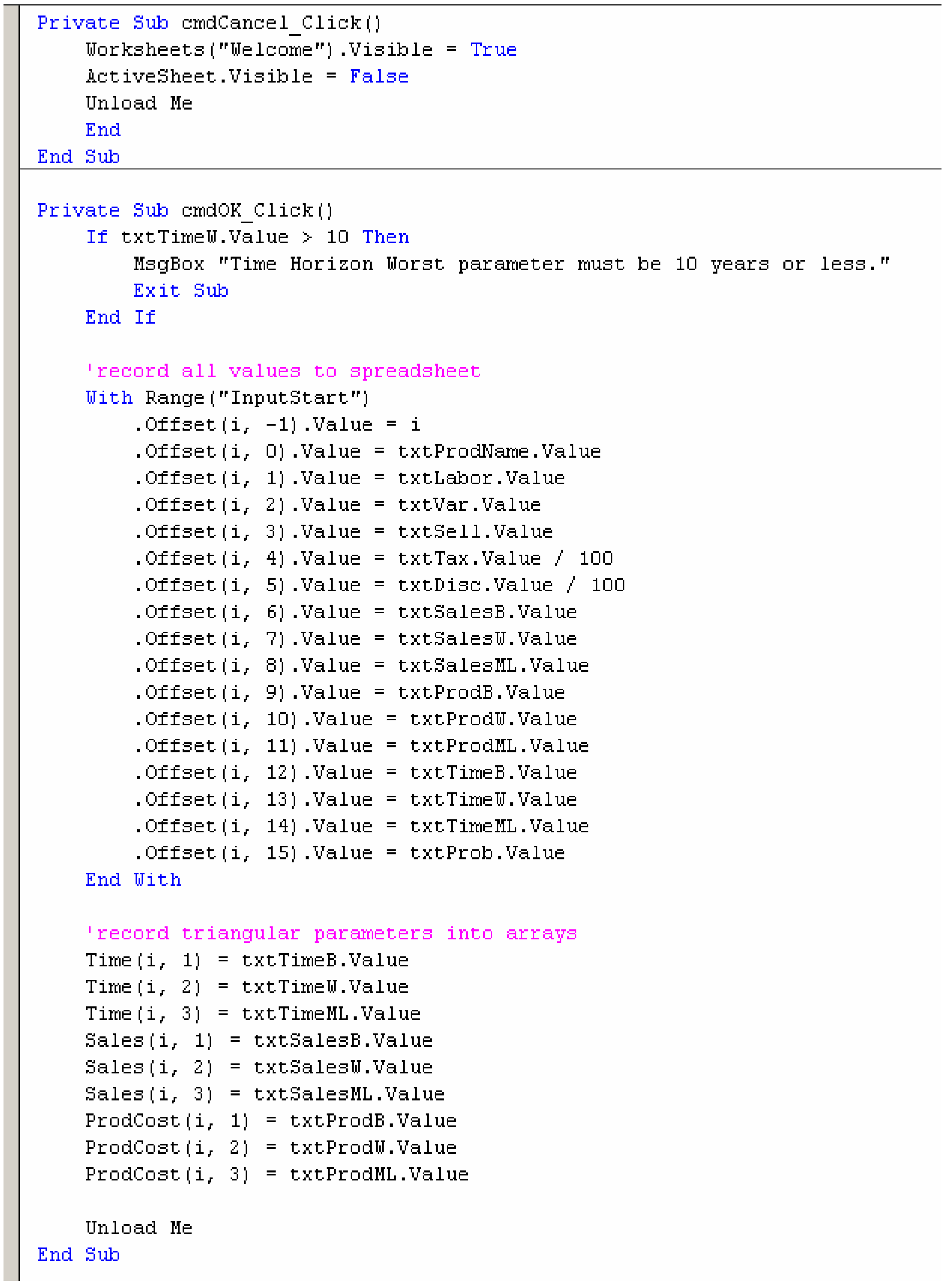
The *Main* procedure ends by calling the *FormatSheets* and *RunSim* procedures.



```
Option Explicit
Option Base 1

Public i As Integer, UserInput As Variant, NumProd As Integer, NumRuns As Integer, runs As Integer, _
Time() As Integer, Sales() As Double, ProdCost() As Double, NumYears As Integer, year As Integer, _
BestNPV As Double, BestProfit As Double, NPVProd As Integer, ProfitProd As Integer

Sub Main()  'called from Start button
    Call ClearPrev

    'how many products
    UserInput = InputBox("For how many PRODUCTS to you want to run the simulation?", _
        "Number of PRODUCTS", 3)
    If IsNumeric(UserInput) = False Or UserInput = "" Then
        Exit Sub
    ElseIf UserInput <= 1 Then
        MsgBox "You must enter at least 2 products."
        Exit Sub
    Else
        NumProd = UserInput
    End If

    'how many runs
    UserInput = InputBox("Please enter number of RUNS for the simulation?", _
        "Number of RUNS", 100)
    If IsNumeric(UserInput) = False Or UserInput = "" Then
        Worksheets("Welcome").Visible = True
        End
    Else
        NumRuns = UserInput
    End If

    ReDim Time(NumProd, 3), Sales(NumProd, 3), ProdCost(NumProd, 3)
    Worksheets("Input").Visible = True
    Worksheets("Input").Activate
    Worksheets("Welcome").Visible = False

    'show input form for each product
    For i = 1 To NumProd
        With frmInput
            .lblProdNum.Caption = i
            .txtProdName.Value = "Product " & Chr(64 + i)   'default product name
            frmInput.Show   'record values to spreadsheet in form code
        End With
    Next i

    'format sheets using input
    Call FormatSheets

    Call RunSim
End Sub
```

Figure CS13.10 The variable declarations and the *Main* procedure.

The *RunSim* procedure performs the simulation calculations. (See Figure CS13.12.) For each simulation run, each product's production development cost, time horizon, and sales volume values are updated with the triangular distribution function. The formulas on the simulation calculation sheet are then updated, and the resulting NPV and profit values are recorded on the simulation details sheet. When all of the runs are complete, the *RunSim* procedure calls the *CreateResults* procedure.

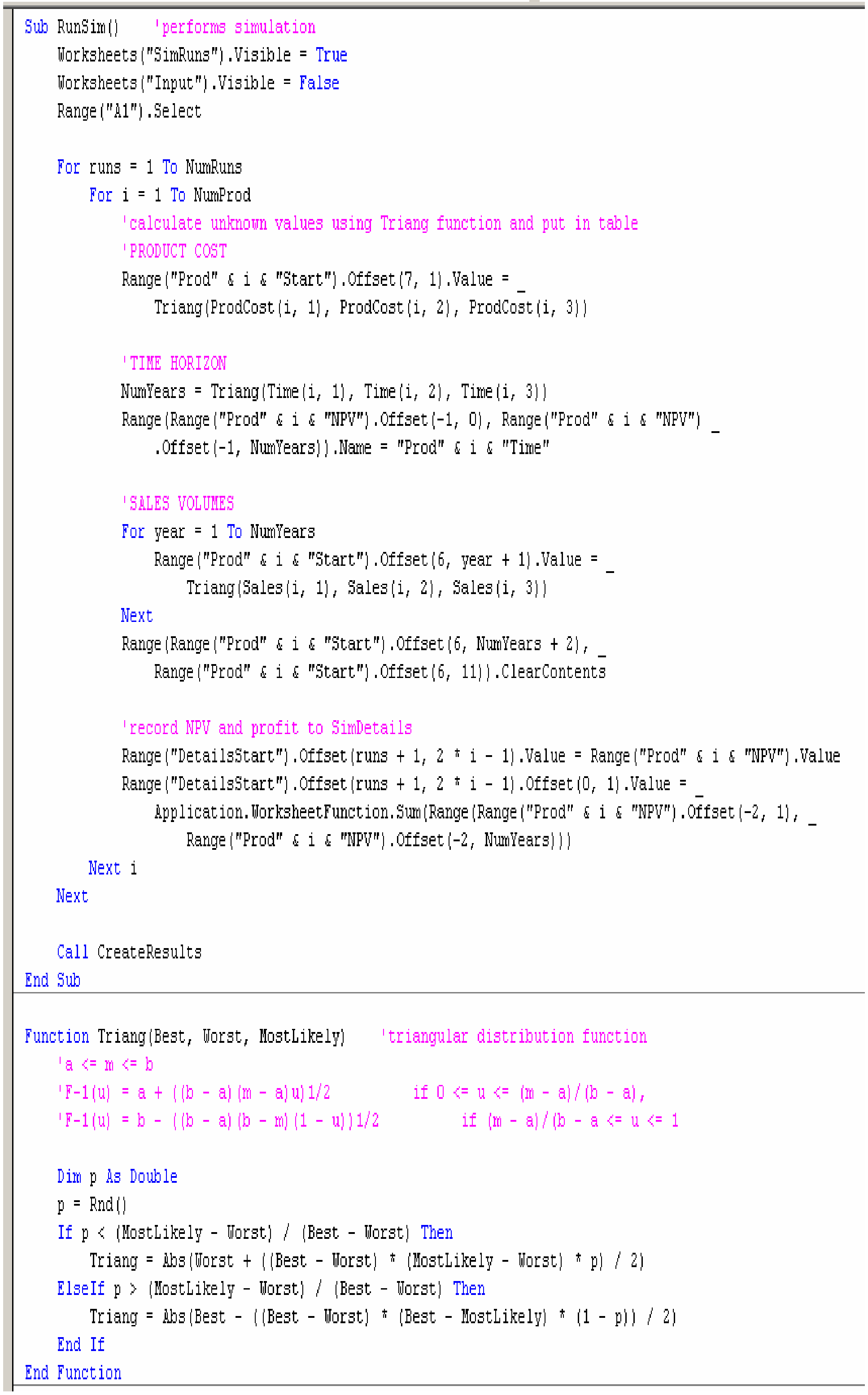


```
Private Sub cmdCancel_Click()
    Worksheets("Welcome").Visible = True
    ActiveSheet.Visible = False
    Unload Me
    End
End Sub

Private Sub cmdOK_Click()
    If txtTimeW.Value > 10 Then
        MsgBox "Time Horizon Worst parameter must be 10 years or less."
        Exit Sub
    End If

    'record all values to spreadsheet
    With Range("InputStart")
        .Offset(i, -1).Value = i
        .Offset(i, 0).Value = txtProdName.Value
        .Offset(i, 1).Value = txtLabor.Value
        .Offset(i, 2).Value = txtVar.Value
        .Offset(i, 3).Value = txtSell.Value
        .Offset(i, 4).Value = txtTax.Value / 100
        .Offset(i, 5).Value = txtDisc.Value / 100
        .Offset(i, 6).Value = txtSalesB.Value
        .Offset(i, 7).Value = txtSalesW.Value
        .Offset(i, 8).Value = txtSalesML.Value
        .Offset(i, 9).Value = txtProdB.Value
        .Offset(i, 10).Value = txtProdW.Value
        .Offset(i, 11).Value = txtProdML.Value
        .Offset(i, 12).Value = txtTimeB.Value
        .Offset(i, 13).Value = txtTimeW.Value
        .Offset(i, 14).Value = txtTimeML.Value
        .Offset(i, 15).Value = txtProb.Value
    End With

    'record triangular parameters into arrays
    Time(i, 1) = txtTimeB.Value
    Time(i, 2) = txtTimeW.Value
    Time(i, 3) = txtTimeML.Value
    Sales(i, 1) = txtSalesB.Value
    Sales(i, 2) = txtSalesW.Value
    Sales(i, 3) = txtSalesML.Value
    ProdCost(i, 1) = txtProdB.Value
    ProdCost(i, 2) = txtProdW.Value
    ProdCost(i, 3) = txtProdML.Value

    Unload Me
End Sub
```

Figure CS13.11 The procedures for the input form.

The *Triang* function procedure generates a possible value from the triangular distribution function using the best case, worst case, and most likely case values as parameters. (See Figure CS13.12.) These parameter values are passed to the *Triang* function procedure using the corresponding arrays for product development cost, time horizon, and sales volume. (Refer to Section CS16.1.1 for details about the triangular distribution function.)

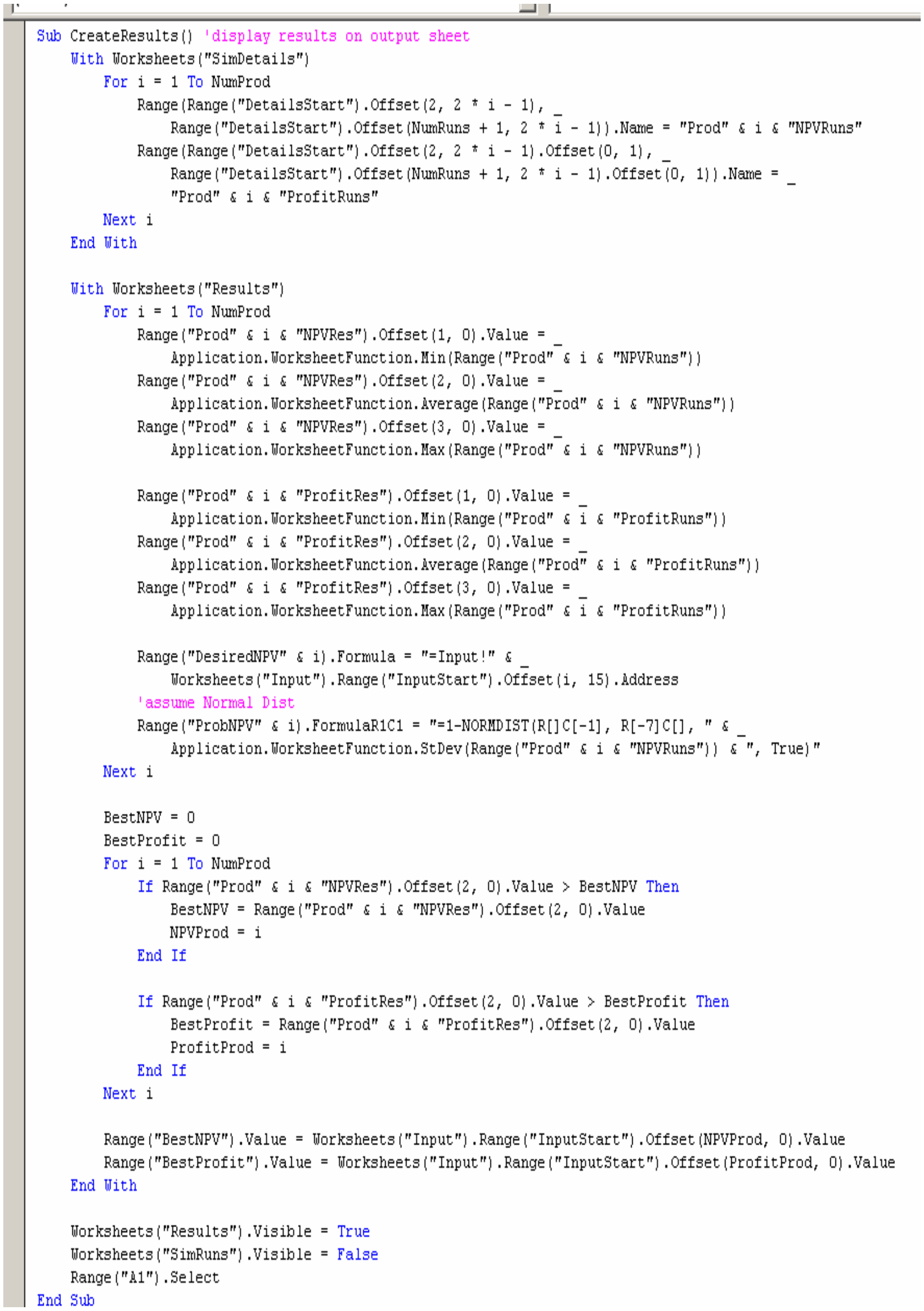
The *CreateResults* procedure displays the results on the results sheet. (See Figure CS13.13.) First, it first creates some dynamic range names on the simulation details sheet. Then, with these range names, it finds the minimum, maximum, and average NPV and profit values for each product. It next loops over all of the average NPV and profit values to find the product with the best NPV and the product with the best profit.



```
Sub RunSim()    'performs simulation
    Worksheets("SimRuns").Visible = True
    Worksheets("Input").Visible = False
    Range("A1").Select

    For runs = 1 To NumRuns
        For i = 1 To NumProd
            'calculate unknown values using Triang function and put in table
            'PRODUCT COST
            Range("Prod" & i & "Start").Offset(7, 1).Value = _
                Triang(ProdCost(i, 1), ProdCost(i, 2), ProdCost(i, 3))

            'TIME HORIZON
            NumYears = Triang(Time(i, 1), Time(i, 2), Time(i, 3))
            Range(Range("Prod" & i & "NPV").Offset(-1, 0), Range("Prod" & i & "NPV") _
                .Offset(-1, NumYears)).Name = "Prod" & i & "Time"

            'SALES VOLUMES
            For year = 1 To NumYears
                Range("Prod" & i & "Start").Offset(6, year + 1).Value = _
                    Triang(Sales(i, 1), Sales(i, 2), Sales(i, 3))
            Next
            Range(Range("Prod" & i & "Start").Offset(6, NumYears + 2), _
                Range("Prod" & i & "Start").Offset(6, 11)).ClearContents

            'record NPV and profit to SimDetails
            Range("DetailsStart").Offset(runs + 1, 2 * i - 1).Value = Range("Prod" & i & "NPV").Value
            Range("DetailsStart").Offset(runs + 1, 2 * i - 1).Offset(0, 1).Value = _
                Application.WorksheetFunction.Sum(Range(Range("Prod" & i & "NPV").Offset(-2, 1), _
                    Range("Prod" & i & "NPV").Offset(-2, NumYears)))
        Next i
    Next

    Call CreateResults
End Sub

Function Triang(Best, Worst, MostLikely)    'triangular distribution function
    'a <= m <= b
    'F-1(u) = a + ((b - a)(m - a)u)1/2          if 0 <= u <= (m - a)/(b - a),
    'F-1(u) = b - ((b - a)(b - m)(1 - u))1/2          if (m - a)/(b - a <= u <= 1

    Dim p As Double
    p = Rnd()
    If p < (MostLikely - Worst) / (Best - Worst) Then
        Triang = Abs(Worst + ((Best - Worst) * (MostLikely - Worst) * p) / 2)
    ElseIf p > (MostLikely - Worst) / (Best - Worst) Then
        Triang = Abs(Best - ((Best - Worst) * (Best - MostLikely) * (1 - p)) / 2)
    End If
End Function
```

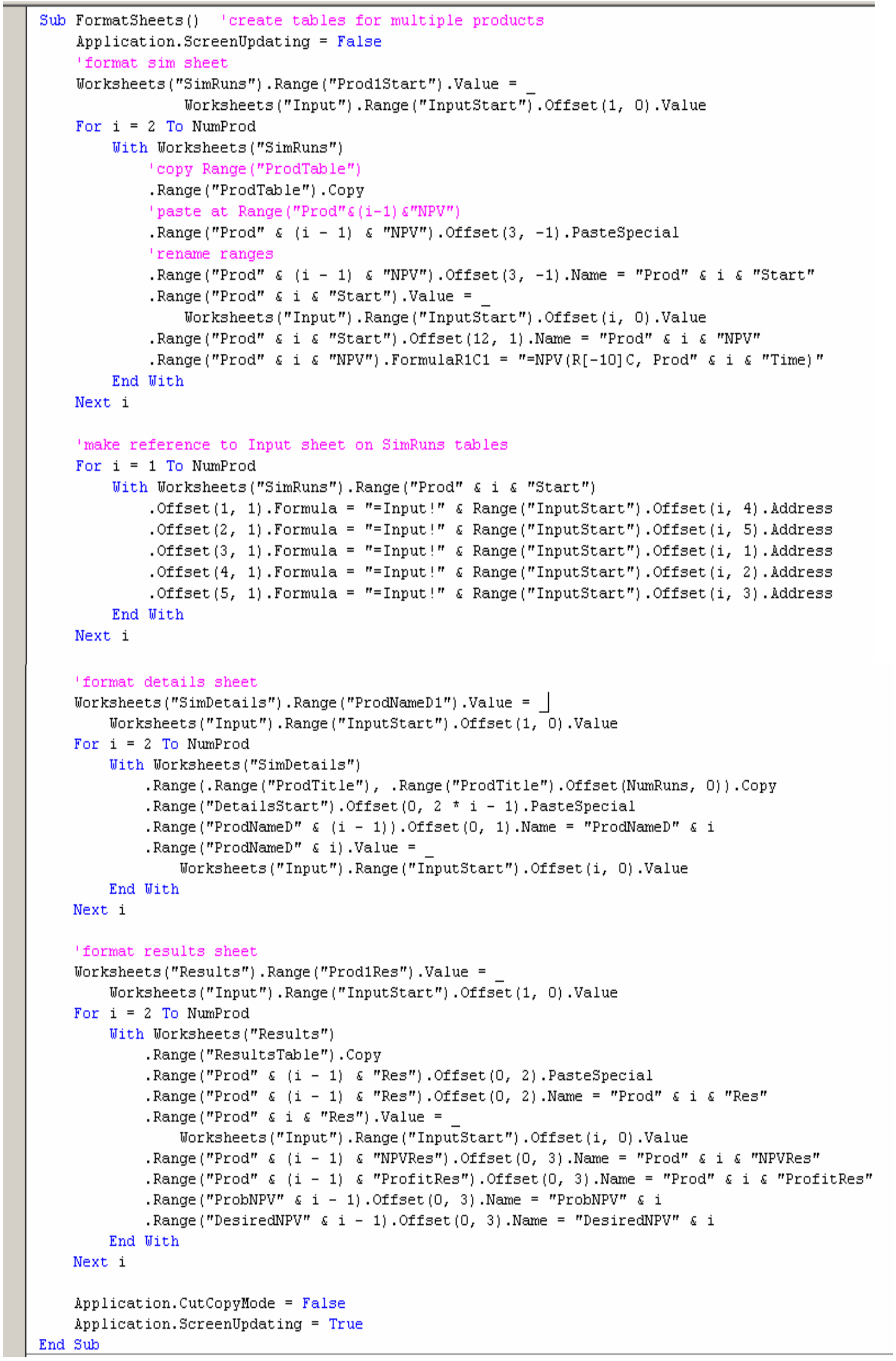
Figure CS13.12 The *RunSim* procedure and the *Triang* function procedure.

The *FormatSheets* procedure creates tables for each product on various sheets. (See Figure CS13.14.) It begins by creating tables for each product on the simulation calculations sheet. It also creates some dynamic range names on this sheet; these names are used in the formulas for the simulation calculations. *FormatSheets* then references the input values from the input sheet on these new tables on the simulation calculation sheet. It then creates columns for storing the simulation results on the simulation details sheet and also updates some of its dynamic range names. Finally, it formats the results sheet by creating the tables for each product to store the results. Notice that for each product, tables are created by copying and pasting an initial table that is never cleared from the sheet.



```
Sub CreateResults() 'display results on output sheet
    With Worksheets("SimDetails")
        For i = 1 To NumProd
            Range(Range("DetailsStart").Offset(2, 2 * i - 1), _
                Range("DetailsStart").Offset(NumRuns + 1, 2 * i - 1)).Name = "Prod" & i & "NPVRuns"
            Range(Range("DetailsStart").Offset(2, 2 * i - 1).Offset(0, 1), _
                Range("DetailsStart").Offset(NumRuns + 1, 2 * i - 1).Offset(0, 1)).Name = _
                "Prod" & i & "ProfitRuns"
        Next i
    End With

    With Worksheets("Results")
        For i = 1 To NumProd
            Range("Prod" & i & "NPVRes").Offset(1, 0).Value = _
                Application.WorksheetFunction.Min(Range("Prod" & i & "NPVRuns"))
            Range("Prod" & i & "NPVRes").Offset(2, 0).Value = _
                Application.WorksheetFunction.Average(Range("Prod" & i & "NPVRuns"))
            Range("Prod" & i & "NPVRes").Offset(3, 0).Value = _
                Application.WorksheetFunction.Max(Range("Prod" & i & "NPVRuns"))

            Range("Prod" & i & "ProfitRes").Offset(1, 0).Value = _
                Application.WorksheetFunction.Min(Range("Prod" & i & "ProfitRuns"))
            Range("Prod" & i & "ProfitRes").Offset(2, 0).Value = _
                Application.WorksheetFunction.Average(Range("Prod" & i & "ProfitRuns"))
            Range("Prod" & i & "ProfitRes").Offset(3, 0).Value = _
                Application.WorksheetFunction.Max(Range("Prod" & i & "ProfitRuns"))

            Range("DesiredNPV" & i).Formula = "=Input!" & _
                Worksheets("Input").Range("InputStart").Offset(i, 15).Address
            'assume Normal Dist
            Range("ProbNPV" & i).FormulaR1C1 = "=1-NORMDIST(R[]C[-1], R[-7]C[], " & _
                Application.WorksheetFunction.StDev(Range("Prod" & i & "NPVRuns")) & ", True)"
        Next i

        BestNPV = 0
        BestProfit = 0
        For i = 1 To NumProd
            If Range("Prod" & i & "NPVRes").Offset(2, 0).Value > BestNPV Then
                BestNPV = Range("Prod" & i & "NPVRes").Offset(2, 0).Value
                NPVProd = i
            End If

            If Range("Prod" & i & "ProfitRes").Offset(2, 0).Value > BestProfit Then
                BestProfit = Range("Prod" & i & "ProfitRes").Offset(2, 0).Value
                ProfitProd = i
            End If
        Next i

        Range("BestNPV").Value = Worksheets("Input").Range("InputStart").Offset(NPVProd, 0).Value
        Range("BestProfit").Value = Worksheets("Input").Range("InputStart").Offset(ProfitProd, 0).Value
    End With

    Worksheets("Results").Visible = True
    Worksheets("SimRuns").Visible = False
    Range("A1").Select
End Sub
```

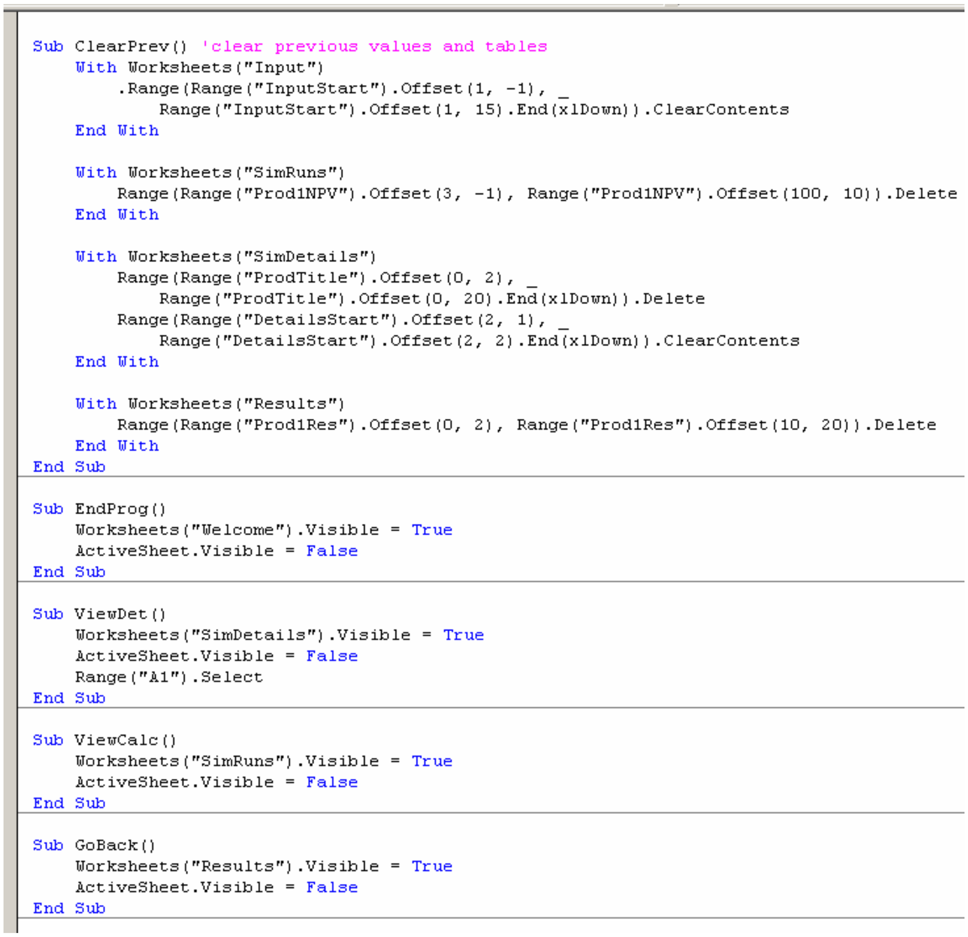
Figure CS13.13 The *CreateResults* procedure.



```
Sub FormatSheets()  'create tables for multiple products
    Application.ScreenUpdating = False
    'format sim sheet
    Worksheets("SimRuns").Range("Prod1Start").Value = _
                Worksheets("Input").Range("InputStart").Offset(1, 0).Value
    For i = 2 To NumProd
        With Worksheets("SimRuns")
            'copy Range("ProdTable")
            .Range("ProdTable").Copy
            'paste at Range("Prod"&(i-1)&"NPV")
            .Range("Prod" & (i - 1) & "NPV").Offset(3, -1).PasteSpecial
            'rename ranges
            .Range("Prod" & (i - 1) & "NPV").Offset(3, -1).Name = "Prod" & i & "Start"
            .Range("Prod" & i & "Start").Value = _
                Worksheets("Input").Range("InputStart").Offset(i, 0).Value
            .Range("Prod" & i & "Start").Offset(12, 1).Name = "Prod" & i & "NPV"
            .Range("Prod" & i & "NPV").FormulaR1C1 = "=NPV(R[-10]C, Prod" & i & "Time)"
        End With
    Next i

    'make reference to Input sheet on SimRuns tables
    For i = 1 To NumProd
        With Worksheets("SimRuns").Range("Prod" & i & "Start")
            .Offset(1, 1).Formula = "=Input!" & Range("InputStart").Offset(i, 4).Address
            .Offset(2, 1).Formula = "=Input!" & Range("InputStart").Offset(i, 5).Address
            .Offset(3, 1).Formula = "=Input!" & Range("InputStart").Offset(i, 1).Address
            .Offset(4, 1).Formula = "=Input!" & Range("InputStart").Offset(i, 2).Address
            .Offset(5, 1).Formula = "=Input!" & Range("InputStart").Offset(i, 3).Address
        End With
    Next i

    'format details sheet
    Worksheets("SimDetails").Range("ProdNameD1").Value = _
        Worksheets("Input").Range("InputStart").Offset(1, 0).Value
    For i = 2 To NumProd
        With Worksheets("SimDetails")
            .Range(.Range("ProdTitle"), .Range("ProdTitle").Offset(NumRuns, 0)).Copy
            .Range("DetailsStart").Offset(0, 2 * i - 1).PasteSpecial
            .Range("ProdNameD" & (i - 1)).Offset(0, 1).Name = "ProdNameD" & i
            .Range("ProdNameD" & i).Value = _
                Worksheets("Input").Range("InputStart").Offset(i, 0).Value
        End With
    Next i

    'format results sheet
    Worksheets("Results").Range("Prod1Res").Value = _
        Worksheets("Input").Range("InputStart").Offset(1, 0).Value
    For i = 2 To NumProd
        With Worksheets("Results")
            .Range("ResultsTable").Copy
            .Range("Prod" & (i - 1) & "Res").Offset(0, 2).PasteSpecial
            .Range("Prod" & (i - 1) & "Res").Offset(0, 2).Name = "Prod" & i & "Res"
            .Range("Prod" & i & "Res").Value = _
                Worksheets("Input").Range("InputStart").Offset(i, 0).Value
            .Range("Prod" & (i - 1) & "NPVRes").Offset(0, 3).Name = "Prod" & i & "NPVRes"
            .Range("Prod" & (i - 1) & "ProfitRes").Offset(0, 3).Name = "Prod" & i & "ProfitRes"
            .Range("ProbNPV" & i - 1).Offset(0, 3).Name = "ProbNPV" & i
            .Range("DesiredNPV" & i - 1).Offset(0, 3).Name = "DesiredNPV" & i
        End With
    Next i

    Application.CutCopyMode = False
    Application.ScreenUpdating = True
End Sub
```

Figure CS13.14 The *FormatSheets* procedure.

The *ClearPrev* and navigational procedures are illustrated in Figure CS13.15.



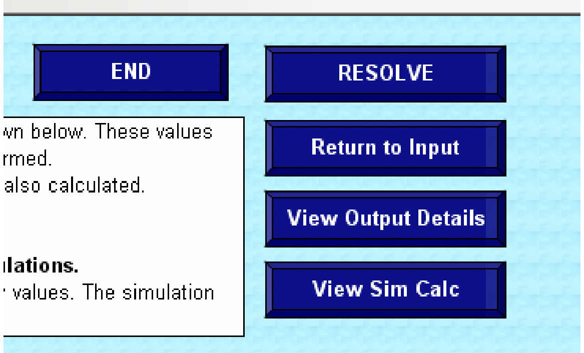
```
Sub ClearPrev() 'clear previous values and tables
    With Worksheets("Input")
        .Range(Range("InputStart").Offset(1, -1), _
            Range("InputStart").Offset(1, 15).End(xlDown)).ClearContents
    End With

    With Worksheets("SimRuns")
        Range(Range("Prod1NPV").Offset(3, -1), Range("Prod1NPV").Offset(100, 10)).Delete
    End With

    With Worksheets("SimDetails")
        Range(Range("ProdTitle").Offset(0, 2), _
            Range("ProdTitle").Offset(0, 20).End(xlDown)).Delete
        Range(Range("DetailsStart").Offset(2, 1), _
            Range("DetailsStart").Offset(2, 2).End(xlDown)).ClearContents
    End With

    With Worksheets("Results")
        Range(Range("Prod1Res").Offset(0, 2), Range("Prod1Res").Offset(10, 20)).Delete
    End With
End Sub

Sub EndProg()
    Worksheets("Welcome").Visible = True
    ActiveSheet.Visible = False
End Sub

Sub ViewDet()
    Worksheets("SimDetails").Visible = True
    ActiveSheet.Visible = False
    Range("A1").Select
End Sub

Sub ViewCalc()
    Worksheets("SimRuns").Visible = True
    ActiveSheet.Visible = False
End Sub

Sub GoBack()
    Worksheets("Results").Visible = True
    ActiveSheet.Visible = False
End Sub
```

Figure CS13.15 *ClearPrev* and the navigational procedures.

Summary	**Main**	Initializes the application and prompts the user with the input boxes and the input form.
	ClearPrev	Clears the previous values and tables from all the sheets.
	Input form procedures	Record the deterministic input values to the input sheet; record the stochastic input parameters to the input sheet and the arrays.
	FormatSheets	Creates the tables for all the products on all the sheets; creates the dynamic range names.
	RunSim	Performs the simulation calculations; updates the stochastic values by calling the *Triang* function procedure.
	Triang function procedure	Uses the triangular distribution function to generate values for the stochastic input.
	CreateResults	Displays the results for each product on the results sheet; finds the product with the best NPV and profit values.
	Navigational	For the “End,” “View Output Details,” “View Sim Calc,” and

CS13.5 *Re-solve Options*

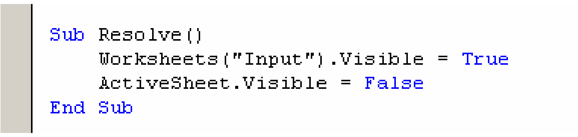
The user can re-solve this application by pressing the “Re-solve” button on the results sheet or the “Start Simulation” button on the input sheet. The “Re-solve” button (as well as the “Return to Input” button) on the results sheet calls the *Re-solve* procedure. (See Figure CS13.16.) This procedure, shown in Figure CS13.17, simply returns the user to the input sheet where he or she can change any of the input values or stochastic input parameters. Once the user updates the input values, he or she can click on the “Start Simulation” button to re-call the *RunSim* procedure and re-run the simulation. (See Figure CS13.18.)



wn below. These values
rmed.
also calculated.

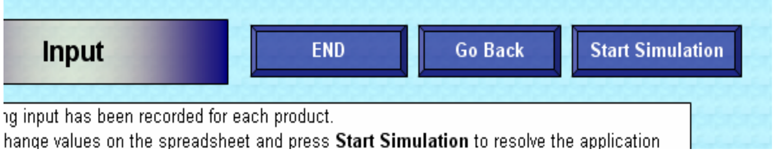
lations.
values. The simulation

Figure CS13.16 The buttons on the results sheet; both “Re-solve” and “Return to Input” lead the user to the re-solve options.



```
Sub Resolve()
    Worksheets("Input").Visible = True
    ActiveSheet.Visible = False
End Sub
```

Figure CS13.17 The *Re-solve* procedure.



ng input has been recorded for each product.
hange values on the spreadsheet and press **Start Simulation** to resolve the application

Figure CS13.18 The buttons on the input sheet; “Start Simulation” re-solves the application.

Summary		
	“Re-solve” button on the results sheet	Returns the user to the input sheet where he or she can change the input values before resolving the problem.
	“Start Simulation” button on the input sheet	Re-calls the *RunSim* procedure to re-run the simulation.

CS13.6 *Summary*

- In this application, the user can consider which of several possible future products would be most beneficial to invest in.
- This application requires five worksheets: the welcome sheet, the input sheet, the simulation calculations sheet, the simulation details sheet, and the results sheet.
- For this application's user interface, we use navigational buttons, input boxes, and a user form.
- Several procedures in this application collect the input for the model and solve the model by running a simulation.
- The user can re-solve this application by pressing the "Re-solve" button on the results sheet or the "Start Simulation" button on the input sheet.

CS13.7 *Extensions*

- Add to the results sheet the product that has the highest probability of having a NPV greater than or equal to the desired NPV.
- Add histograms to the results sheet that illustrate the frequency with which each product's profit and NPV values were achieved. (Remember to also add to the ClearPrev procedure so that each histogram is deleted for the previous products.)
- Restructure this application to solve the capital budgeting problem with an integer programming model. Which procedures change and which can be retained? Which sheets change and which can be retained?