CHAPTER 10
EXTRA RESOURCES

Additional Resources

1. "Abstraction and Information Hiding" (*http://tiny.cc/abstraction/*): See how subroutines can simplify complicated code.
2. "Other Names for Subroutines" (*http://tiny.cc/othernames/*): Learn what other programming languages call subroutines.
3. "Subroutines Input and Output" (*http://tiny.cc/inputandoutput/*): See how your program can pass variables to and from your subroutines.
4. "The Triangle Area Calculator" (*http://tiny.cc/trianglearea/*): Follow this tutorial to see how subroutines help you build interactions.
5. "Nesting Subroutines" (*http://tiny.cc/nestingsubroutine/*): Learn how to nest or embed your subroutines.
6. "Using Stacks" (*http://tiny.cc/stacks/*): Learn how to use stacks to manage massive amounts of user input and data.

7. "Small Basic Recursion" (*http://tiny.cc/sbrecursion/*): Check out this first look at recursion looping in subroutines.
8. "The Dragon Game" (*http://tiny.cc/dragongame/*): Share your dragon game improvements and see what others did!

Review Questions

1. What are the benefits of using subroutines?
2. What's the syntax for defining and calling a subroutine? When should you use parentheses, and when should you not use them?
3. How can you make sure subroutines don't get too complicated?
4. How does the main program differ from the subroutines?
5. How is data flow managed between a subroutine and its caller?
6. What are input variables in subroutines?
7. What are working variables in subroutines?
8. What are output variables in subroutines?
9. When can you define and access variables in Small Basic?
10. What is recursion?

Practice Exercises

1. Mickey wants to get milk from Clarabelle's Grocery Store. He needs to know how many quarts the store has so he can then convert the answer to liters using the following program. Run the program and explain how it works.

```
TextWindow.Write("How many quarts do you have? ")
quart = TextWindow.ReadNumber()

QuartToLiter()
TextWindow.WriteLine(quart + " quarts = " + liter + " liters")

Sub QuartToLiter
  liter = 0.946 * quart
EndSub
```

2. Harvey is an old, wise rabbit who lives in a cave. He has mystical powers that allow him to answer any yes/no question. You don't believe this? Complete the subroutine in this program and see for yourself.

```
Again:
TextWindow.Write("Please ask me a yes/no question: ")
ques = TextWindow.Read()
ThinkAndAnswer()

TextWindow.WriteLine("")
Goto Again

Sub ThinkAndAnswer
  ' Print one of these answers at random:
  ' "Yes, I believe so.", "No, not really.",
  ' "Not a chance.", "Probably yes.", "I have to say no.",
  ' "There is a slight possibility.", "The signs are good.",
  ' "I must say no on that one.", "Your question is confusing.",
  ' "Chances are 50/50.", "Perhaps. It depends who's asking.",
  ' "Maybe. I'm a rabbit.", "No. Tricks are for kids.", "Sure."
EndSub
```

3. The area of an equilateral triangle (a triangle whose three sides are equal) is given by this formula:

$$Area = 0.5\sqrt{0.75}(s)^2$$

s is the length of one side of the equilateral triangle. Complete the `TriArea()` subroutine in the following program.

```
' TriArea.sb
' Finds the area of an equilateral triangle, given its side length

TextWindow.Write("Please enter the side length: ")
side = TextWindow.ReadNumber()

TriArea() ' Call the TriArea() subroutine

TextWindow.WriteLine("Area = " + area)

' This subroutine finds the area of an equilateral triangle
' Input: side - the length of the side of the triangle
' Output: area - the computed area
Sub TriArea
  area =  ' Add your code here
EndSub
```

Run the program to check your answer. Here is a sample run:

```
Please enter the side-length: 5
Area = 10.83
```

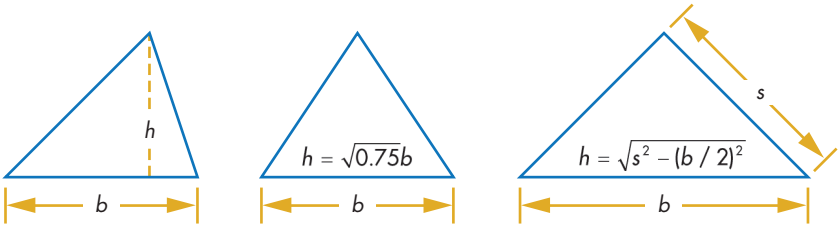
4. Write a program that calculates the area of a rectangle of length *L* and width *W* using these steps:
   a. Prompt your user to enter the length and width of the rectangle.
   b. Call a subroutine, `RectangleArea()`, to compute the area of the rectangle. Have the subroutine save the result in a variable named `area`.
   c. Display the computed area from the main program to your user.
5. Lucky the Leprechaun needs to pick his lottery numbers. Write a subroutine that picks three numbers (`num1`, `num2`, and `num3`) between 1 and 10 at random. Write another subroutine that displays these numbers in ascending order in a text window.
6. Captain Jean-Luc Skywalker is about to launch his spaceship, but he needs your help. Help him write a subroutine that displays a countdown starting from a given number `N` (ask the captain what number he wants to count down from, and then assign his number to `N`). For example, if he wants `N` to be 5, the subroutine should display: `[5 4 3 2 1 0]. Blast off!`
7. Liam lost his Infinity Stone under the stove! He's not sure if his lightsaber is long enough to help him reach the stone, because he can't measure distance in feet (he's British)! Write a subroutine named `FeetToMeter` that converts from feet to meters. Ask Liam the number of feet he needs to convert. Make the input to the subroutine a variable named `feet` and the output a variable named `meters`. Then show Liam how many meters long his lightsaber is!
8. Chester the Cheetah is a fast runner. But he's not sure if he can run faster than a car. He needs you to write a subroutine that converts feet per second (which Chester enters) to miles per hour and displays it. So, is Chester faster than a car that travels at 30 miles per hour?
9. Mickey and Liam are on the phone talking about the outside temperature, but they're confused by how cold it is in each other's cities. Mickey doesn't understand Celsius (C), and Liam doesn't understand Fahrenheit (F). Write a menu-driven program that converts temperature readings between Fahrenheit and Celsius. Show this menu to Mickey and Liam:

```
TEMPERATURE CONVERTER - Enter your choice (1 or 2):
[1] Convert from Celsius to Fahrenheit
[2] Convert from Fahrenheit to Celsius
```

   Have your program read their choices and then call a subroutine that gets the required inputs and calculates the temperature. (Hint: C = (F − 32) × 5 ÷ 9 and F = C × 9 ÷ 5 + 32.)
10. Mickey's back for more milk! This time he's getting it from Liam, so the amount is in liters. Write a subroutine named `LiterConverter` that converts liters to ounces, cups, pints, quarts, and gallons. Put Mickey's input into a variable named `liters`. Then have the subroutine compute

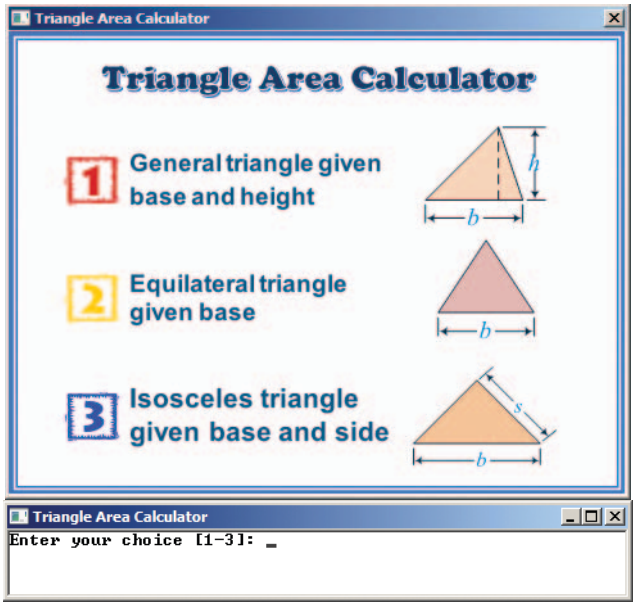
five variables named `ounces`, `cups`, `pints`, `quarts`, and `gallons` that hold the results of the conversion (and display your results). (Hint: 1 gallon = 3.785 liters; 1 gallon = 4 quarts; 1 quart = 2 pints; 1 pint = 2 cups; 1 cup = 8 ounces.) Help Mickey understand how much milk he has.

11. Open the file *TriangleArea.sb* and run it. This program can compute the area of any of the following three triangle types.



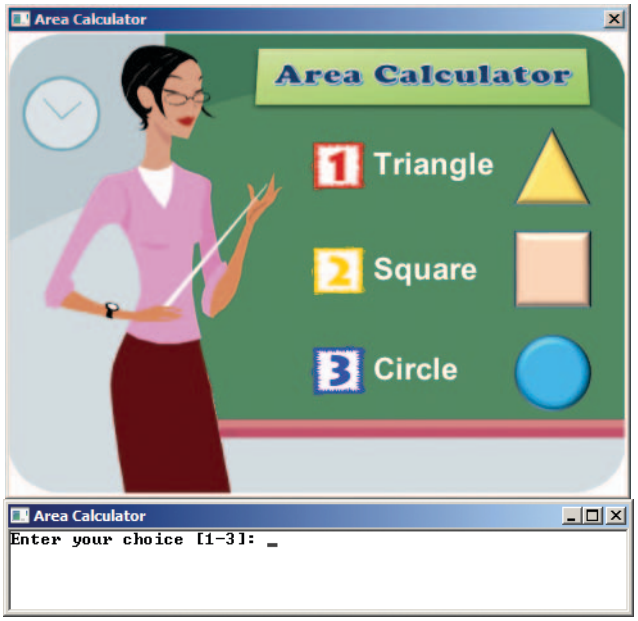
General triangle given base and height | Equilateral triangle given base | Isosceles triangle given base and side

The program starts by displaying a graphical menu (shown in the following figure) and asking the user to enter his choice. Based on the value entered, the program calls a different subroutine to collect the needed inputs, and then it computes the area. Study the program's code, and explain how it works.



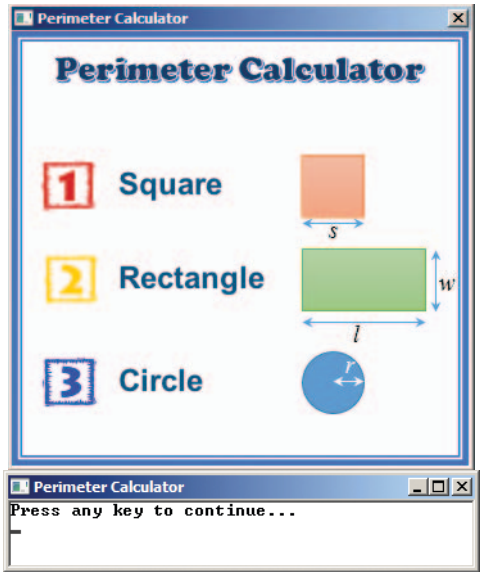
12. Open the file *AreaCalculator.sb*. This exercise extends the triangle area calculation program in the previous exercise to other geometric shapes. It adds the ability to find the area of a square—your user enters

either its side or the length of its diagonal—and the area of a circle—your user enters either its radius or its circumference (perimeter). The following interface is displayed to the user.



Study the program's structure and explain how it works.

13. Open the perimeter-calculating program, *PerimCalc_Incomplete.sb*, and run it. This incomplete application displays the following interface to your user:



Complete the program to have it read the user's choice, call the right subroutine to get the needed inputs, compute the perimeter, and then display the perimeter of the selected shape.