

Scripting Languages G22.3033-002 Summer 2008 hw02

Assigned Th 5/29/2008, due Fr 6/6 at 4pm. 50 points.

<http://www.cs.nyu.edu/courses/summer08/G22.3033-002/>

Homework instructions

Homeworks are due on Fridays at 9pm. This deadline will be strictly enforced.

Email your answers to Robert Soulé robert.soule@gmail.com. Please put your solutions to VBA programming problems in a powerpoint presentation. For all other questions (including programming problems in other languages), just send a simple text file, such as what you get when using Emacs, Vi, Notepad, or the "save as text only" feature in Word.

Please make sure that your code works with the compilers and tools installed at CIWW. Specifically, please test:

JAVASCRIPT

Your JavaScript code must run with **both** Mozilla's Firefox browser **and** with Microsoft's Internet Explorer browser.

PHP Your PHP code must run with PHP 5.1.6 on the CIMS web servers, see <http://www.cims.nyu.edu/systems/userservices/webhosting/>.

PERL Your Perl code must run with Perl 5.8.8 for Linux/x86, for example on doowop1 (see <http://www.cims.nyu.edu/systems/resources/computerservers/>).

VBA Your VBA code must run with Microsoft Office 2003 for Windows on the machines in the labs CIWW 502 or CIWW 624.

Reading assignments

Read for lecture on 6/5: perlintro(1) man page

Kirrily "Skud" Robert. perlintro(1) man page. Perl 5 documentation. Available on machines that have Perl properly installed, and also available online at <http://perldoc.perl.org/perlintro.html>.

Concept questions

hw02-1 Properties

(5+5 = 10 points)

- Give an example for reading and writing each of the properties of the following VBA class. Briefly explain what it does.

```
Option Explicit
Public Celsius As Double
Public Property Let Fahrenheit(f As Double)
    Celsius = 5 * (f - 32) / 9
End Property
Public Property Get Fahrenheit() As Double
    Fahrenheit = (9 * Celsius / 5) + 32
End Property
```

- b. How would you implement a write-once property in VBA?

hw02-2 Call-backs

(5 + 5 = 10 points)

- a. Explain briefly how VBA supports call-backs.
- b. Describe another plausible choice that the VBA designers could have made for call-backs.

Programming exercises

hw02-3 Associative array (VBA)

(15 points) Consider the following abstract class `StringMap`.

```
Option Explicit

Public Function size() As Integer
End Function

Public Function Contains(key As String) As Boolean
End Function

Public Property Let Data(key As String, Value As Variant)
End Property

Public Property Get Data(key As String)
End Property
```

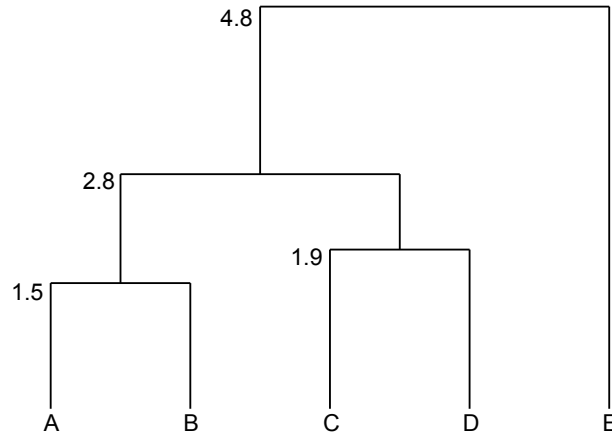
`StringMap` defines the interface for a container that maps strings to values. Write a subclass `SimpleStringMap` that implements `StringMap`. For example, consider the following test driver:

```
Sub driveStringMap()
    Dim m As StringMap
    Set m = New SimpleStringMap
    m.Data("apple") = "Apfel"
    m.Data("pear") = "Birne"
    Debug.Print m.size & " " & m.Data("apple")
    m.Data("apple") = "pomme"
    Debug.Print m.size & " " & m.Data("apple")
End Sub
```

The program should print 2 Apfel, 2 pomme. Don't worry about performance, just write a simple correct implementation of the interface. Make sure that your implementation works for an unbounded number of elements, not just two.

hw02-4 Dendrogram (VBA)

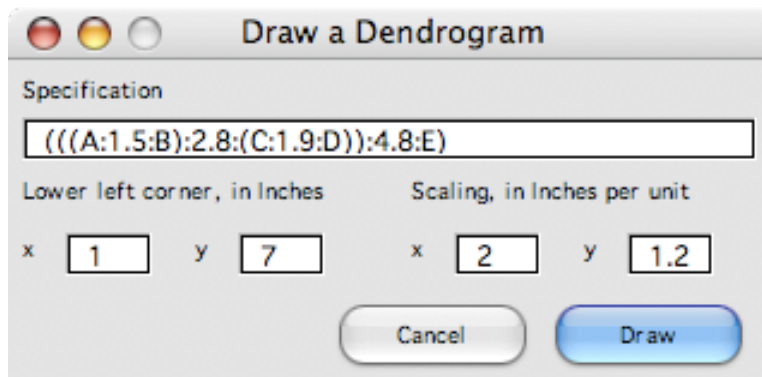
(15 points) A dendrogram is a tree diagram that arranges similar items in clusters. For example, the following dendrogram shows items A-E in a hierarchy, annotating each cluster with a similarity score.



A textual specification for the above example dendrogram is

`((A:1.5:B):2.8:(C:1.9:D)):4.8:E`

Your task is to design a dialog box for drawing a dendrogram that looks like this:



This will require you to create a user form, and to wire it up with the appropriate callbacks. To do the actual drawing, you can use the subroutine `DrawDendrogram` defined below.

```
Sub ParseDendrogram(ByVal spec As String, ByRef lspec As String, _
    ByRef y As Double, ByRef rspec As String)
    Dim c1 As Integer, c2 As Integer, depth As Integer, i As Integer
    c1 = -1: c2 = -1: depth = 0
```

```

For i = 1 To Len(spec)
  Select Case Mid(spec, i, 1)
    Case "("
      depth = depth + 1
    Case ")"
      depth = depth - 1
    Case ":"
      If 1 = depth Then If c1 < 0 Then c1 = i Else c2 = i
  End Select
Next i
lspec = Mid(spec, 2, c1 - 2)
y = CDBl(Mid(spec, c1 + 1, c2 - c1 - 1))
rspec = Mid(spec, c2 + 1, Len(spec) - c2 - 1)
End Sub

Sub DrawDendrogram(ByVal cvs As Canvas, ByVal spec As String, _
  ByRef x As Double, ByRef y As Double)
  ' The original x is the position where the left-most leaf should go.
  ' On return, x = width of subtree and y = height of subtree.
  Dim ox As Double
  ox = x
  If Mid(spec, 1, 1) = "(" Then
    Dim lspec As String, rspec As String
    ParseDendrogram spec, lspec, y, rspec
    Dim lx As Double, ly As Double, rx As Double, ry As Double
    lx = x
    DrawDendrogram cvs, lspec, lx, ly
    rx = ox + lx
    DrawDendrogram cvs, rspec, rx, ry
    x = lx + rx
    Dim x1 As Double, x2 As Double
    x1 = ox + 0.5 * (lx - 1): x2 = ox + lx + 0.5 * (rx - 1)
    With ActiveWindow.Selection.SlideRange.Shapes
      .AddLine(cvs.x(x1), cvs.y(ly), cvs.x(x1), cvs.y(y)).Select
      .AddLine(cvs.x(x1), cvs.y(y), cvs.x(x2), cvs.y(y)).Select
      .AddLine(cvs.x(x2), cvs.y(y), cvs.x(x2), cvs.y(ry)).Select
    End With
  Else
    x = 1: y = 0
    ActiveWindow.Selection.SlideRange.Shapes.AddLabel( _
      msoTextOrientationHorizontal, cvs.x(ox), cvs.y(0), 0.4, 0.4).Select
    ActiveWindow.Selection.ShapeRange.TextFrame.TextRange.Text = spec
  End If
End Sub

```

The DrawDendrogram subroutine relies on an instance of class Canvas to translate logical x/y positions into physical positions on the slide. The class module Canvas is defined as follows:

```
Public xbase As Double, xscale As Double
Public ybase As Double, yscale As Double

Public Function x(ByVal logicalX As Double)
    x = xbase + xscale * logicalX
End Function

Public Function y(ByVal logicalY As Double)
    y = ybase - yscale * logicalY
End Function
```