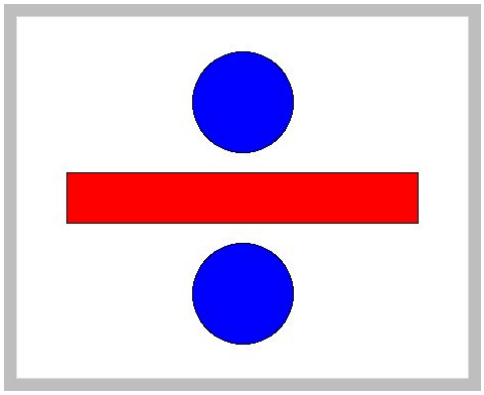
15110 PRINCIPLES OF COMPUTING – LAB EXAM 2 –FALL 2012 C

Name _____ Section _____ Andrew ID _____ Machine_____

Directions:

1. *In your home directory (as soon as you start the terminal), create a folder named labexam2.*
2. *Write a function in Ruby for each of the following problems using gedit and store these functions in the labexam2 folder. Use the Ruby Reference Sheet to assist you with syntax issues.*
3. *Test your functions by calling them within irb. Although we give you sample test runs, your function should work completely based on the given specifications and your output should match the sample usage as closely as possible for full credit. Remember that we will run your code on additional test cases that are not shown on the exam.*
4. *Once you are finished, compress the labexam2 folder into a zip file and submit it to AutoLab (http://autolab.cs.cmu.edu) by the end of lab. Do not delete the labexam2 folder from your home directory.*

1. (25 pts) Write a Ruby function `f1()` (in the file `f1.rb` in your `labexam2` folder) that draws a division symbol centered in a window of size 450 by 360. The red bar should have height 50 pixels and width 350 pixels; its top left corner is at x=50, y=155. The blue circles should have radius 50 pixels; the gap between their edge and the bar should be 25 pixels. The result should look just like the picture below, but if you're off by a pixel or two, it's okay. Refer to the Ruby Reference Sheet for help with Canvas method syntax.



2. (25 pts) Write a function `f2(m,n)` that constructs a matrix of size m rows by n columns containing sequential values from 0 up through (m*n)-1. You can assume that both m and n will be greater than zero. Use the following algorithm:

1. Create a variable *result* as an array of length m.
2. For each row i from 0 to m-1:
    a. Set result[i] to a new array of length n.
    b. For each column j in row i, set that element to (i*n)+j.
3. When you've computed all the elements, return *result*.

Sample usage:

```
>> f2(3,2)
=> [[0,1], [2,3], [4,5]]]

>> f2(2,3)
=> [[0,1,2], [3,4,5]]]
```

3. (25 pts) In the file `f3.rb` in your `labexam2` folder, write a recursive function `f3(list)` that takes a list of numbers as input and triples each one, returning a new list as its result. Do not use a for loop; your function must be purely recursive.

Sample usage:

```
>> f3([2, 3, 105, 7, 9])
=> [6, 9, 315, 21, 27]

>> f3([])
=> []
```

4. (25 pts) Write a function `f4(matrix)` (in the file `f4.rb` in your `labexam2` folder) that takes as input a matrix of numbers and returns *true* if the matrix is symmetric, i.e., if M[i][j] equals M[j][i] for all i and all j. Your function must work for any N×N matrix. Hint: go through all possible combinations of i and j and test each one to make sure the symmetry relation holds.

Sample usage:

```
>> f4([[1,2],[2,1]])
=> true

>> f4([[1,2,3],[2,4,5],[3,5,6]])
=> true

>> f4([[1,2],[3,4]])
=> false

>> f4([[1]])
=> true
```