15110 PRINCIPLES OF COMPUTING – LAB EXAM 1 – SPRING 2012 A

Name _____ Section _____ Andrew ID _____ Machine_____

Directions:

1. *In your home directory, create a folder named labexam.*
2. *Write a function in Ruby for each of the following problems using gedit and store these functions in the labexam 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 labexam folder into a zip file and submit it to AutoLab (http://autolab.cs.cmu.edu) by the end of lab. Do not delete the labexam folder from your home directory.*

1. (25 pts) Write a Ruby function `f1(x,y)` (in the file `f1.rb` in your `labexam` folder) that prints all of the multiples of 5 between integers `x` and `y`, inclusive, if `x` is less than or equal to `y`. Print one value per line. If `x` is greater than `y`, print INVALID with a newline. Your function should always return nil when it is done.

Sample usage:

```
>> f1(4,19)
5
10
15
=> nil
```

```
>> f1(10,25)
10
15
20
25
=> nil
```

```
>> f1(24,15)
INVALID
=> nil
```

```
>> f1(18,18)
=> nil
```

2. (25 pts) Write a function `f2(list)` (in the file `f2.rb` in your `labexam` folder) that takes a `list` of integers and returns the index of the minimum integer in the `list`. You may assume that the `list` has at least one element. If the minimum occurs more than once, return the index of the first occurrence of the minimum. **You may not use the `min` function in your solution.**

Sample usage:

```
>> f2([4,2,5,8])
=> 1
```

```
>>f2([3,5,7,9,4])
=> 0
```

```
>> f2([0,6,-4])
=> 2
```

```
>> f2([7])
=> 0
```

(OVER)

3. (25 pts) Write a function `f3(list)` (in the file `f3.rb` in your `labexam` folder) that takes a list of positive integers and prints a bar graph of X's where the number of X's on each line is given by each integer in the list. Return nil when the function is finished. You may assume the list has at least one integer. **HINT: Your function should use nested loops.**

Sample usage:

```
>> f3([3, 2, 5, 8, 4])
XXX
XX
XXXXX
XXXXXXXX
XXXX
=> nil
```

4. (25 pts) Write a function `f4(n)` (in the file `f4.rb` in your `labexam` folder) that stores the first `n` numbers in the following sequence into a new array and returns this array.

1, 2, 4, 7, 11, 16, 22, 29, …

You may assume n is a positive integer. Follow this algorithm:

1. Create a new array *sequence* of size *n*.
2. Set *sequence[0]* = 1
3. For each remaining index *i* of the *sequence* array do the following:
    a. Set each element at index *i* equal to the previous element in the array plus *i*.
4. Return the *sequence* array.

Sample usage:

```
>> f4(8)
=> [1, 2, 4, 7, 11, 16, 22, 29]

>> f4(14)
=> [1, 2, 4, 7, 11, 16, 22, 29, 37, 46, 56, 67, 79, 92]

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