15110 PRINCIPLES OF COMPUTING – EXAM 3A – FALL 2012

1	8
2	12
3	14
4	14
5	14
6	20
7	18
TOTAL	100

Name ______________________________ Section ________

Directions: Answer each question neatly in the space provided.

Please read each question carefully. You have 50 minutes for

this exam. No electronic devices allowed. Good luck!

1. [8pts] This question concerns generating random numbers in Ruby. Recall that the Ruby function `rand(n)` returns a random integer between 0 and n-1, inclusive. Using the `rand` function, show how to compute the following:

1a. [2pt] A random integer between 0 and 99, including 99. rand(100)

1b. [2pt] A random integer between 5 and 10, including 10. rand(6) + 5

1c. [2pt] A random string from the array `cars` below. cars[rand(4)]

```
cars = ["Chevrolet", "Honda", "Mercedes", "Toyota"]
```

1d. [2pt] An random odd integer between 1 and 9, inclusive. 2*rand(5) + 1

2. [12pts] This question concerns writing functions that involve randomness.

2a. [2pt] Suppose that a course grade can either be a "pass" or "fail". The following function uses randomness to return a grade. The function is written such that a pass or fail grade is equally likely. Fill in the blanks. You can assume that when `rand(n)` is used, every value in the range 0 to n-1 is equally likely as a return value.

```
def lazy_teacher ()
 if rand( 2 ) == 0 then
  return "pass"
else
  return "fail"
end
```

2b. [2pt] Write a different version of the function above that returns “pass” with probability 90% and “fail” with probability 10%.

```
def generous_teacher ()
  if rand( 10 ) <= 8 then
    return "pass"
  else
    return "fail"
  end
end
```

OR Any numbers that will give the same ratio
e.g. rand(100) ≤ 89

2c. [4pt] Consider the simple functions given below. They simulate throwing a 6-sided die and a 10-sided die, respectively.

```
def roll6()
 return rand(6) + 1
end

def roll10()
  return rand(10) + 1
end
```

Suppose that we want to simulate a game by throwing a 6-sided die and a 10-sided die by using the functions above. If the result of the 6-sided die is greater than the result of the 10-sided die we return the sum of two dice. Otherwise, we return the result of the 10-sided die. Point out the flaw in the following function that attempts to implement this simulation.

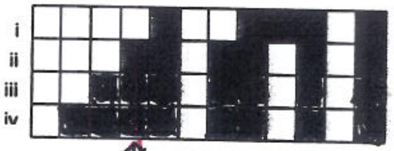
```
def faulty()
  if roll6() > roll10() then
    return roll6() + roll10()
  else
    return roll10()
  end
end
```

The function above is not a correct implementation because each time we use one of the roll functions we generate a new random number. In order to use the numbers we generate we need to store them in a variable.

2d. [4pt] Write a function of your own that eliminates the flaw in the function `faulty()`. Your function should call the functions `roll6()` and `roll10()`. Hint: Your function should also use assignments.

```
def correct( )
    x = roll6( )
    y = roll10( )
    if x > y then
      return x + y
    else
      return y
    end
end
```

3. [14pts] This problem concerns one-dimensional binary cellular automata such as the ones you experimented with in the OLI module and homework assignments. Consider an automaton whose initial state is shown in line (i) below. The next generation is shown in line (ii).



not filled in

3a. [4pts] What is the rule for this automaton? In the spaces below, fill in the dashed boxes with a 1 (for black) or a 0 (for white) to specify the rule this automaton must be using to produce line (ii) from line (i).

7	6	5	4	3	2	1	0
■■■	■■□	■□■	■□□	□■■	□■□	□□■	□□□
0	1	0	0	1	1	1	0

3b. [4pts] Using the rule numbering scheme we studied, the rule for this automaton must be a number between 0 and 255. What is the number? 78

3c. [6pts] Apply the rule to compute generations (iii) and (iv) above by coloring in selected squares.

4. [14pts] This question concerns networking and the Internet.

4a. [2pts] In packet-switching networks, two network nodes (e.g. computers) send messages by breaking the message up into small packets and sending each packet on to the network with a serial number and a destination address. A router is a device that determines the next network point to which a packet should be forwarded, on its way to its final destination.

4b. [2pts] The principle known as net neutrality advocates no restrictions by ISPs or governments on consumers' access to networks that participate in the Internet.

4c. [2pts] A domain name server is used to translate human-friendly computer hostnames into IP addresses.

4d. [2pts] A communication protocol is a collection of rules that govern the ways in which computers interact with one another.

4e. [4pts] Suppose that computers in an institution are assigned IPv4 addresses that start with a common sequence of 16 bits that uniquely identify the institution's network. Given that there are 32 bits in an IPv4 address, how many hosts can this institution have in its network using this addressing scheme?

2^{16}

4f. [2pts] A Web page is identified by a Uniform Resource Locator (URL), which has the following format:

part1://*part2*/page

Part 1 stands for a protocol such as HTTP.
Part 2 stands for a domain name.
or
host address

5. [14pts] This question concerns key establishment and encryption.

Alice and Bob want to agree on a secret key they can use to exchange encrypted messages. They use the Key Agreement Protocol of Diffie, Helman, and Merkel to construct this key. The protocol relies on the existence of a function f(x,y) that is a one-way function and also has the special property that f(f(p,q),r) equals f(f(p,r),q). Alice picks a secret value a and Bob picks a secret value b. They both know the value of g, which is an industry standard constant.

5a. [3pts] Why is it safe for Alice to publicly disclose f(g,a), and Bob to publicly disclose f(g,b)?

f is a one-way function, so cannot recover a or b

5b. [3pts] Alice can derive the secret key she needs to communicate securely with Bob by taking his publicly disclosed value for f(g,b) and her secret value a and doing what?

Compute f(f(g,b), a)

5c. [3pt] How does Bob derive the same secret key that Alice has obtained?

Compute f(f(g,a), b)

5d. [5pts] Mark each of the following as True or False, assuming f(x,y) is a one-way function.

True It is easy to compute the value of f(x,y), given x and y.

False It is easy to compute the value of y, given f(x,y) and x.

True The product of two primes is a one-way function as long as you know the product f(x,y) but neither of the primes x and y.

False The product of two primes is a one-way function even if you know f(x,y) and also x, but not y.

False A substitution cipher c=f(p,k), such as the Caesar cipher, where c is the ciphertext, k is the key, and p is the plaintext, is a one-way function if you don't know k.

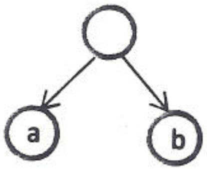
6. [20pts] This question concerns AI and recursion.

Consider a two-player game where there are two choices for each move, A or B. We can represent the game tree in Ruby using a nested array. Before the first move, the tree is just the root node:

```
[]
```

We can extend the tree to the next level by replacing each terminal node by a list of two new nodes, generated by appending a symbol :a or :b to the original node. Initially the root is the only terminal node. After the first move, we have one nonterminal node (the root) with two terminal nodes, [:a] and [:b], so the tree (also shown at right) becomes:

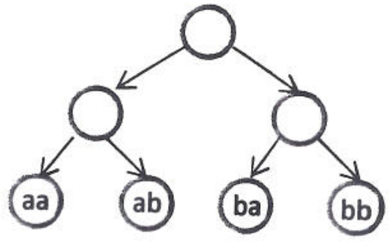
```
[ [:a], [:b] ]
```



Notice that tree[0] is the left terminal node, and tree[1] is the right one.

We can repeat the tree generation process described above to create the tree for a two move game:

```
[ [[:a,:a], [:a,:b]], [[:b,:a], [:b,:b]] ]
```



As the above list and the tree at right both show, there are four possible games consisting of two moves. The left subtree, i.e., the first nonterminal node below the root, is tree[0], and the first terminal node is tree[0][0], which is [:a,:a]. The last terminal node is tree[1][1], which is [:b,:b].

We can in turn use this tree to generate the tree for three-move games:

```
[ [[[:a,:a,:a], [:a,:a,:b]], [[:a,:b,:a],[:a,:b,:b]]],
  [[[:b,:a,:a], [:b,:a,:b]], [[:b,:b,:a],[:b,:b,:b]]] ]
```

The above tree has eight terminal nodes, corresponding to the eight possible games with three moves. The first terminal node is [:a,:a,:a] and the last is [:b,:b,:b]. Terminal nodes are lists composed of the symbols :a and :b; they do not contain any lists inside them.

6a. [6pts] Write a function `terminal?(node)` that takes as input a list representing some node in a game tree for this game, and returns true if `node` is a terminal node, and false otherwise. Your function should return true for inputs such as [:b] or [:a,:b] or [:b,:a,:a] and false for non-terminal inputs such as [[:a],[:b]]. Note that the empty game tree [ ] is also a valid terminal node. You can assume that `node` will always be either a valid terminal node or a valid non-terminal node, not some random junk.

```
def terminal?(node)
  if    not node[0].kind_of? (Array)
  then
       return true
  else
       return false
  end
end
```

6b. [7pts] Write a recursive function that takes a non-empty game tree as input and returns the first (leftmost) terminal node. Your function should call the `terminal?` function you wrote above.

```
def first_terminal(tree)
  if   terminal? (tree)
  then
       return tree
  else
       return first-terminal (tree[0])
  end
end
```

6c. [7pts] Write a recursive function that takes a game tree as input and returns the number of moves in a game. You can assume that the game tree is complete, i.e., all nodes at the same level have the same number of children, so the number of moves is equal to the level of the terminal nodes, which is the depth of the tree. Your function should call the `terminal?` function you wrote above.

```
def number_of_moves(tree)
  if  terminal? (tree)
  then
       return 0
  else
       return 1 + number-of-moves (tree[0])
  end
end
```

7. [18pts] This question concerns concurrency.

Suppose that two programs P1 and P2 have access to the same variable x in memory, and they can use actions read(x) and dec(x,c), which represent, respectively, reading from the location x and decreasing the value in location x by the value c. The program steps for P1 and P2 are given below.

```
# Program P1
P1A: a = read(x)

P1B: if a > 2 then dec(x, 3)

         else do nothing
```

```
# Program P2
P2A: b = read(x)

P2B: if b > 1 then dec(x, 2)

         else do nothing
```

7a. [4pts] Suppose that x initially contains the value 3, and P1 and P2 are executed sequentially such that the program steps are executed in the following order: P1A, P1B, P2A, P2B. What will the memory location x contain at the end of the execution?

After step P1A, the value of x is: 3

After step P1B, the value of x is: 0

After step P2A, the value of x is: 0

After step P2B, the value of x is: 0

7b. [4pts] Suppose that the programs P1 and P2 are executed by two different processors that have access to a shared memory containing the variable x. This means that the steps of the two programs may be interleaved. For example, the first 3 steps in an execution can be P1A, P2A, P2B. Now, assume that the memory location x initially contains the value 3 as in the previous question. Give a 4-step execution sequence that starts with the step P2A such that the resulting value of x is negative.

P2A, P1A, P1B, P2B

or

P2A, P1A, P2B, P1B

7c. [4pts] Now consider the following programs `P3` and `P4`, which are executed concurrently. Notice that they include calls to `P1` and `P2`, respectively. If we want to guarantee that `x` never contains a negative value after the programs complete their execution, we may want to treat the calls to `P1` and `P2` as critical sections and try to ensure that only one program is executing its critical section at a time. You can assume that `xfree_P3` and `xfree_P4` are shared variables that are initialized to true and that functions `noncritical_P3()` and `noncritical_P4()` do not involve the location `x`. Give an execution sequence in terms of the labels in the given programs that results in a deadlock.

P3A, P3B, P4A, P4B, P3C, P4C

(multiple correct answers)

```
Program P3:

while true do

 P3A:   call noncritical_P3()

 P3B:   xfree_3 = false

 P3C:   while xfree_4 == false

           do nothing

        end

 P3D: call Program P1

 P3E: xfree_3 == true

end
```

```
Program P4:

while true do

   P4A: call noncritical_P4()

   P4B: xfree_4 = false

   P4C: while xfree_3 == false

             do nothing

         end

    P4D:   call Program P2

    P4E:   xfree_4 == true

end
```

Any interleaving that sets both xfree_3 and x_free4 to false before any of P3C or P4C is executed would lead to deadlock

7d. [6pts] Pipelining is used in computers to speed up computer execution. The steps for executing an instruction are as follows:

F. Fetch instruction from memory
D. Decode the instruction
R. Read data from registers
E. Execute the instruction
W. Write the result into a register.

Suppose that each of the above steps takes 1 time unit to perform, and we want to execute 5 instructions in a pipelined fashion. Draw a diagram to illustrate the pipelined execution of the five instructions, i1 through i5. There are multiple ways to graphically depict pipelining; we used two different ways in the lecture notes. You are free to pick whatever convention you like, provided that it correctly expresses the pipelining process. For example, you might choose to have time run vertically, or you might prefer that it run horizontally. Use the grid below to make your diagram. You can label the rows and columns, in any way you need to show how your diagram makes sense. What you write in the grid squares is also up to you. And we've given you way more space than you need, so don't expect to use every row and column. This is actually an easy problem, so don't panic!

	1	2	3	4	5	6	7	8	9			
i1	F	D	R	E	W							
i2		F	D	R	E	W						
i3			F	D	R	E	W					
i4				F	D	R	E	W				
i5					F	D	R	E	W			