1.(14 pts total) History of computation.

(a) (4 pts) Match each item in the left column with the most relevant item in the right column.

<table>
<thead>
<tr>
<th>Item</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jacquard</td>
<td>6</td>
</tr>
<tr>
<td>Babbage</td>
<td>8</td>
</tr>
<tr>
<td>Hopper</td>
<td>2</td>
</tr>
<tr>
<td>Turing</td>
<td>7</td>
</tr>
<tr>
<td>Moore</td>
<td>4</td>
</tr>
<tr>
<td>Engelbart</td>
<td>3</td>
</tr>
<tr>
<td>Pascal</td>
<td>5</td>
</tr>
<tr>
<td>Napier</td>
<td>1</td>
</tr>
</tbody>
</table>

1. bones
2. a moth
3. a mouse
4. Doubling in complexity
5. Leibniz
6. Hollerith
7. Enigma cipher
8. Lovelace

(b) (6 pts) We want to use the method of finite differences to create a table of numbers for the function \( f(x) = -7x^2 + 2x - 5 \). Compute the necessary difference functions and the initial values for \( x = 0 \) for this machine. Then fill in the table for \( x \) values from 1 to 3.

\[
\Delta f(x) = _{-14} \quad x + _{-5}
\]

\[
\Delta^2 f(x) = _{-14}
\]
(c) (2 pts) 1KB of memory is \( 2^{10} \) or \( 10^3 \) (decimal number) bytes.

(d) (2 pts) In 1982 a state of the art microprocessor had 100,000 transistors. According to Moore’s Law, when would you expect to see microprocessors with 200,000 transistors? Compute the correct year between 1982 and 1997, and explain your reasoning.

If we assume beginning of 1982 then middle of 1983, since the number of transistors double every 18 months

2. This problem focuses on expressions and data types.

(a) (6 pts) For each of the following Ruby expression, write down the value that would be output if the expression was evaluated in irb.

<table>
<thead>
<tr>
<th>x</th>
<th>( \Delta f(x) )</th>
<th>( \Delta f(x) )</th>
<th>f(x)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-14</td>
<td>-5</td>
<td>-5</td>
</tr>
<tr>
<td>1</td>
<td>-14</td>
<td>-19</td>
<td>-10</td>
</tr>
<tr>
<td>2</td>
<td>-14</td>
<td>-33</td>
<td>-29</td>
</tr>
<tr>
<td>3</td>
<td>-14</td>
<td>-47</td>
<td>-62</td>
</tr>
</tbody>
</table>

\[
\begin{array}{c|c|c|c}
 x & \Delta f(x) & \Delta f(x) & f(x) \\
0 & -14 & -5 & -5 \\
1 & -14 & -19 & -10 \\
2 & -14 & -33 & -29 \\
3 & -14 & -47 & -62 \\
\end{array}
\]

40 / 9 \( \quad 4 \) \quad 15.0 / 2 \( \quad 7.5 \)

2 * 2 ** 4 \( \quad 32 \)

15 % 2 \( \quad 1 \)

6 + 4 * 2 - 1 \( \quad 13 \)

2 != 2 \( \quad \text{false} \)

(b) (2 pts) Write a Ruby function `triangle_area` that takes two parameters `h` and `b`, respectively, for the height and base of a triangle, and returns the area of the triangle given by the formula \( A = \frac{1}{2} (\text{height} \times \text{base}) \).
def triangle_area(h, b):
    return 0.5 * h * b
end

(c) (2 pts) Write a Ruby function truncated_triangle that takes height and base parameters (h and b) as input, and computes the area of a triangle that has the tip cut off. The tip is also a triangle; its height and base are 10% of the height and base, respectively, of the larger triangle, as shown in the figure. Use the triangle_area function in your solution.

def truncated_triangle_area(h, b):
    h_tip = 0.1 * h
    b_tip = 0.1 * b
    return triangle_area(h, b) - triangle_area(h_tip, b_tip)
end

(d) (2 pts)

def mystery1(m, n):
    i = 0
    while i <= n - 1 do
        i = i + 1
        print i ** m, " "
    end
end

The Ruby function above prints a sequence of numbers. Which of the following is the output of the function expressed in terms of m and n? Circle your answer.

1 2^m 3^m ... n^m OR 1  m^2  m^3 ...  m^n
(e) (4 pts) If the print statement was taken outside of the while loop so that it occurred right after the while statement, as shown below, what would the function call mystery2(2, 3) print?

```ruby
def mystery2(m, n)
  i = 0
  while i <= (n-1) do
    i = i + 1
  end
  print i ** m, ", "
end
```

(f) (4 pts) Consider the following Ruby function:

```ruby
def mystery3(m, n)
  i = 0
  result = 0
  while i <= (n-1) do
    i = i + 1
    result = result + i ** m
  end
  return result
end
```

What would the value of the variable x be after executing the following assignment statement below?

```ruby
x = mystery3(2,4)
```

(3) (20 pts total) This question focuses on the array data type and iterators.

(a) (6 pts) Assume the following list definition in Ruby using an array.

```ruby
cars = ["Honda", "Toyota", "Kia", "Chrysler", "Mercedes"]
```

What would be displayed in `irb` for each of the following Ruby expressions?
cars.length       ___________5___________
cars.first       ___________“Honda”___________
cars[1]        ___________“Toyota”___________
cars.include?(“Mazda”)     _____________false_________
cars.include?(“KIA”)      ______________false________
cars.each { |x| print x + “*” if x  >  “Kia”}
                        _____________Toyota*Mercedes*________

(b) (4 pts) Assume the following list definition in Ruby using an array.
a= [1, 2, [3, 4, 5], 6]
What would be displayed in irb for each of the following Ruby expressions?
a.length    ________4______________
a.first                 _________1_____________
a[2]     _________[3,4,5]_____________
a + a     _______[1,2,[3,4,5],6,1,2,[3,4,5],6]_____________________

(c) (10 pts) Assume the following list definition in Ruby using an array.
a = [2, 4, 6, 7, 8]
What would be returned in irb for if the following Ruby expressions are executed in the given order?
a.collect{ |x| x / 2 }    _______[1,2,3,3,4]_____________________
a                _______[2,4,6,7,8]_____________________


4. (20 pts total) This question focuses on looping.

(a) (8 pts) We wish to define a Ruby function `find_oddball` that takes an “almost sorted” list as input and returns the first item that is not in descending order. The function should return nil if the list is entirely in descending order. For example, `find_oddball [24, 17, 5, 12, 1]` should return 12, since 12 is greater than the preceding item, 5. Complete the following iterative function `find_oddball`.

```ruby
def find_oddball (list)
    index = ___0____
    while index < __list.length-1_____
        if _____list[index]_________ < _____list[index+1]_________
            return ____ list[index+1]________
        end
        index = __index + 1__________
    end
    return ___nil______
end
```

(b) (8 pts) Consider the following recursive algorithm for returning the first item in a list that is not in descending order, else nil. Complete the recursive definition of `find_oddball`.

```ruby
def find_oddball (list)
    return find_oddball(list[1..-1]) unless list[0] > list[1]
    return list[0] if list.length == 1
    min = find_oddball(list[1..-1])
    return list[0] if min == nil or list[0] > min
    min
end
```
1. If the list has fewer than two elements, return nil.
2. If the first element in the list is less than the second element, return the first element.
3. Otherwise return the result of a recursive call on the tail of the list (i.e., everything beyond the first element.)

def find_oddball (list)
    if list.length < 2 then
        return nil
    elsif list[0] < list[1] then
        return list[0]
    else
        return find_oddball(list[1..list.length-1])
    end

(c) (2 pts) Give an example of a six element list that would be a worst case input for find_oddball.

_______Any six element list with all elements in non-ascending order
_____________________________

(d) (2 pts) What is the big O worst case complexity of find_oddball?

_______O(n)____________

5. (20 pts) This question deals with searching and sorting.

(a) (2 pts)
What is the big O complexity of linear search? _____O(n)___________

What is the big O complexity of merge sort? _____O(n log n)___________

(b) (6 pts) Fill in the table below to show how binary search would locate the value “g” in the list ["a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k"]. Use the binary search algorithm taught in the book and covered in lecture. Note: this table may contain extra rows.
(c) (6 pts) For each sorting algorithm described below, give its correct name:

- For each position $i$ in the list, find the index of the smallest item at or to the right of position $i$, and swap $\text{list}[i]$ with $\text{list}[\text{index of smallest}]$.

\[\text{selection}\]

- Organize the inputs into $N$ groups of size 1. Systematically combine adjacent groups to form $N/2$ sorted groups, each of size 2. Repeat the process, combining adjacent groups of size 2 to form $N/4$ sorted groups of size 4. Keep going until you have one sorted group of size $N$.

\[\text{merge}\]

- For each input item, find its proper position in the result list and add it at that position.

\[\text{insertion}\]

(d) (6 pts) Suppose we want to know if all the elements of a list are the same. For example, `all_same([1, 1, 1, 99, 1, 1])` returns false, but `all_same(["f","f","f","f"])` should return true. Here are two solutions. Fill in the missing elements.

```python
def no_mismatch1(list)
    list.each { |x| return ___false___ if ___list[x]___ != list[0] }
```
def no_mismatch2 (list)
    sorted_list = list.sort
    if sorted_list[0] == sorted_list[ ___list.length-1____________ ]
        return __true____
    else
        return ___false____
    end
end

What is the big O complexity of no_mismatch1? ______O(n)__________

What is the big O complexity of no_mismatch2? ______O(n lg n) or O(n^2) _depending on what is the complexity of sort _________

6. (6 pts) This question is based on your readings from the book Blown to Bits.

In parking garages, people with big parking bills sometimes attempt to get a second ticket at the end of their trip so that they would have to pay a smaller fee at the time of exit. What technology is used in many garages to prevent this? Give a one sentence answer.

Every car entering the garage has its license plate photographed at the same time the ticket is being taken.