NAME: VISHWANTH P

REGNO: 21MIS1117

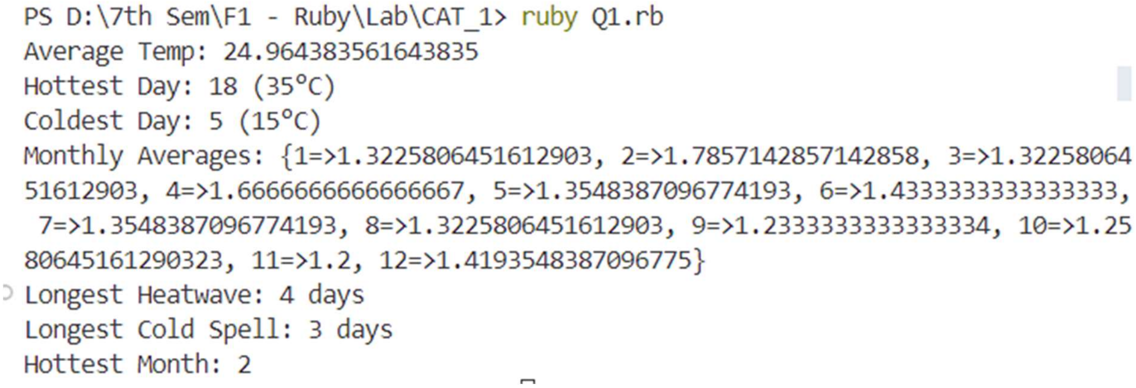
CAT – 1 Solution

Question – 1

```
def generate_temp_data
  temps = {}
  (1..365).each do |day|
    temps[day] = rand(15..35)
  end
  temps
end

def calc_avg_temp(temps)
  total_temp = temps.values.sum
  total_temp.to_f / temps.size
end

def find_temp_extremes(temps)
  hot_day, hot_temp = temps.max_by { |_day, temp| temp }
  cold_day, cold_temp = temps.min_by { |_day, temp| temp }
  { hot_day: hot_day, hot_temp: hot_temp, cold_day: cold_day, cold_temp:
cold_temp }
end

def calc_monthly_avg(temps)
  days_in_month = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
  monthly_avg = {}
  day = 1

  days_in_month.each_with_index do |days, idx|
    month_temps = temps.slice(day, days).values
    monthly_avg[idx + 1] = month_temps.sum.to_f / days
    day += days
  end

  monthly_avg
end

def find_long_heatwave(temps)
  heatwave = 0
```

```ruby
  max_wave = 0

  temps.each_value do |temp|
    if temp > 30
      heatwave += 1
    else
      max_wave = [max_wave, heatwave].max
      heatwave = 0
    end
  end
  max_wave
end

def find_long_cold_spell(temps)
  cold_spell = 0
  max_spell = 0

  temps.each_value do |temp|
    if temp < 20
      cold_spell += 1
    else
      max_spell = [max_spell, cold_spell].max
      cold_spell = 0
    end
  end
  max_spell
end

def find_hot_month(monthly_avg)
  monthly_avg.max_by { |_month, avg_temp| avg_temp }.first
end

temps = generate_temp_data
avg_temp = calc_avg_temp(temps)
extremes = find_temp_extremes(temps)
monthly_avg = calc_monthly_avg(temps)
long_heatwave = find_long_heatwave(temps)
long_cold_spell = find_long_cold_spell(temps)
hot_month = find_hot_month(monthly_avg)

puts "Average Temp: #{avg_temp}"
puts "Hottest Day: #{extremes[:hot_day]} (#{extremes[:hot_temp]}°C)"
puts "Coldest Day: #{extremes[:cold_day]} (#{extremes[:cold_temp]}°C)"
puts "Monthly Averages: #{monthly_avg}"
puts "Longest Heatwave: #{long_heatwave} days"
puts "Longest Cold Spell: #{long_cold_spell} days"
puts "Hottest Month: #{hot_month}"
```

Output



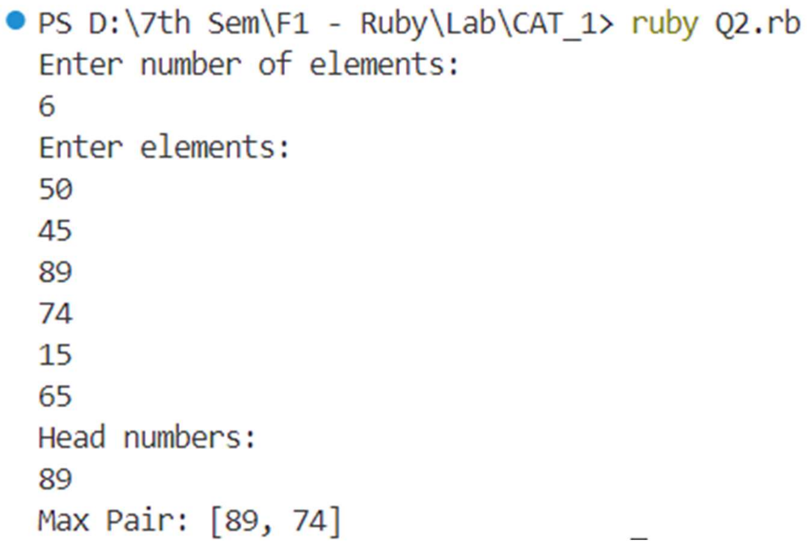
```
PS D:\7th Sem\F1 - Ruby\Lab\CAT_1> ruby Q1.rb
Average Temp: 24.964383561643835
Hottest Day: 18 (35°C)
Coldest Day: 5 (15°C)
Monthly Averages: {1=>1.3225806451612903, 2=>1.7857142857142858, 3=>1.32258064
51612903, 4=>1.6666666666666667, 5=>1.3548387096774193, 6=>1.4333333333333333,
 7=>1.3548387096774193, 8=>1.3225806451612903, 9=>1.2333333333333334, 10=>1.25
80645161290323, 11=>1.2, 12=>1.4193548387096775}
Longest Heatwave: 4 days
Longest Cold Spell: 3 days
Hottest Month: 2
```

Question – 2

```
class NumAnalyzer
  def find_heads(nums, n)
    puts "Head numbers: "
    (1...n - 1).each do |i|
      if nums[i] > nums[i - 1] && nums[i] > nums[i + 1]
        puts nums[i]
      end
    end
  end

  def find_max_pair(nums, n)
    max_pair = [nums[0], nums[1]]
    max_sum = nums[0] + nums[1]
    (0...n - 1).each do |i|
      (i + 1...n).each do |j|
        cur_sum = nums[i] + nums[j]
        if cur_sum > max_sum
          max_sum = cur_sum
          max_pair = [nums[i], nums[j]]
        end
      end
    end
    max_pair
  end
end

puts "Enter number of elements: "
n = gets.chomp.to_i

nums = []

puts "Enter elements: "
n.times do
  nums << gets.chomp.to_i
end
```

```
analyzer = NumAnalyzer.new

analyzer.find_heads(nums, n)

max_pair = analyzer.find_max_pair(nums, n)

puts "Max Pair: #{max_pair}"
```

Output



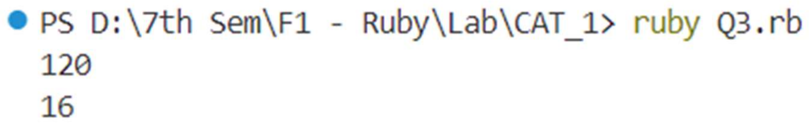
```
PS D:\7th Sem\F1 - Ruby\Lab\CAT_1> ruby Q2.rb
Enter number of elements:
6
Enter elements:
50
45
89
74
15
65
Head numbers:
89
Max Pair: [89, 74]
```

Question – 3

```
class DynamicDispatcher
  def method_missing(method_name, *args)
    if method_name.to_s.start_with?("calculate")
      operation = method_name.to_s.split("calculate")[1]
      perform_calculation(operation, args)
    else
      super
    end
  end

  def respond_to_missing?(method_name, include_private = false)
    method_name.to_s.start_with?("calculate") || super
  end

  private

  def perform_calculation(operation, args)
    case operation
    when "Factorial"
      puts factorial(args[0])
    when "Square"
      puts square(args[0])
    else
```

```
      puts "Unsupported operation: #{operation}"
    end
  end

  def factorial(n)
    return 1 if n == 0
    n * factorial(n - 1)
  end

  def square(n)
    n * n
  end
end

dispatcher = DynamicDispatcher.new

dispatcher.calculateFactorial(5)
dispatcher.calculateSquare(4)
```

Output



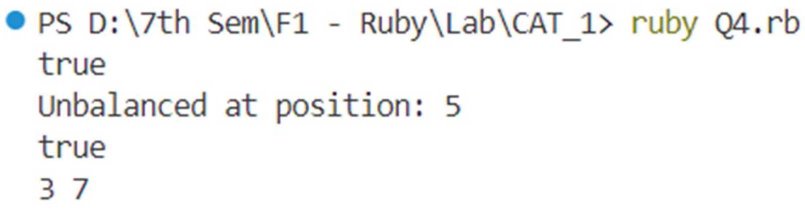
```
PS D:\7th Sem\F1 - Ruby\Lab\CAT_1> ruby Q3.rb
120
16
```

Question – 4

```
class BracketChecker
  def balanced_parentheses(str)
    pairs = { '(' => ')', '{' => '}', '[' => ']', '<' => '>' }
    stack = []

    str.each_char.with_index do |char, idx|
      if pairs.key?(char)
        stack.push([char, idx])
      elsif pairs.value?(char)
        return "Unbalanced at position: #{idx + 1}" if stack.empty? ||
pairs[stack.last[0]] != char
        stack.pop
      end
    end

    stack.empty? ? true : "Unbalanced at position: #{stack.last[1] + 1}"
  end

  def evaluate_expression(str)
    return balanced_parentheses(str) unless balanced_parentheses(str) == true
```

```ruby
    exprs = str.scan(/\([^)]+\)/)

    exprs.each do |expr|
      result = eval(expr[1..-2])
      str.sub!(expr, result.to_s)
    end

    str
  end
end

checker = BracketChecker.new

puts checker.balanced_parentheses("(11)")
puts checker.balanced_parentheses("|(|)])")
puts checker.balanced_parentheses("<(1+2) (3+4)>")

puts checker.evaluate_expression("(1+2) (3+4)")
```

Output



```
● PS D:\7th Sem\F1 - Ruby\Lab\CAT_1> ruby Q4.rb
  true
  Unbalanced at position: 5
  true
  3 7
```

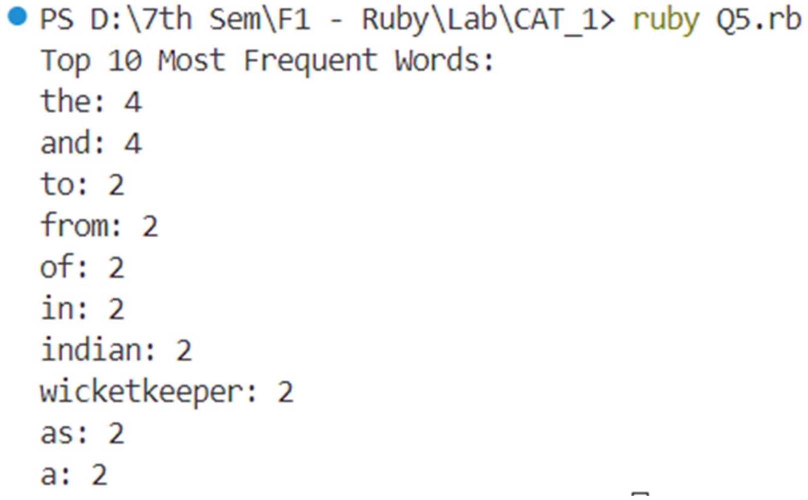
Question – 5

```ruby
require 'fileutils'

class WordCounter
  def initialize(filename)
    @filename = filename
    @word_counts = {}
  end

  def analyze_word_usage
    read_file
    count_words
    output_results
  end
  private

  def read_file
    File.open(@filename, 'r') do |file|
      @text = file.read
    end
  end

  def count_words
    words = @text.downcase.gsub(/[^a-z\s]/, '').split
    words.each do |word|
      @word_counts[word] ||= 0
      @word_counts[word] += 1
    end
  end

  def output_results
    sorted_words = @word_counts.sort_by { |_word, count| -count }
    puts "Top 10 Most Frequent Words:"
    sorted_words[0..9].each do |word, count|
      puts "#{word}: #{count}"
    end
  end
end

# Example usage
filename = 'sample.txt'
word_counter = WordCounter.new(filename)
word_counter.analyze_word_usage
```

sample.txt

Mahendra Singh Dhoni (born 7 July 1981) is an Indian professional cricketer who plays as a right-handed batter and a wicket-keeper. Widely regarded as one of the most prolific wicket-keeper batsmen and captains, he represented the Indian cricket team and was the captain of the side in limited overs formats from 2007 to 2017 and in test cricket from 2008 to 2014.

Output



```
● PS D:\7th Sem\F1 - Ruby\Lab\CAT_1> ruby Q5.rb
  Top 10 Most Frequent Words:
  the: 4
  and: 4
  to: 2
  from: 2
  of: 2
  in: 2
  indian: 2
  wicketkeeper: 2
  as: 2
  a: 2
```