|  |
| --- |
| # ======================================================================================== |
| # Assignment 2 |
| # ======================================================================================== |
|  |
| # ======================================================================================== |
| # Problem 1 - `to\_sentence` |
|  |
| # implement method `to\_sentence` |
|  |
| # creates an english string from array |
|  |
|  |
| **def to\_sentence(ary)**  **return nil if ary.nil?**  **return ary[0] if ary.length == 1**  **return ary[0..-2].join(', ') + " and " + ary[-1] if ary.length > 1**  **end** |
|  |
|  |
| # Your method should generate the following results: |
| to\_sentence [] #=> "" |
| to\_sentence ["john"] #=> "john" |
| to\_sentence ["john", "paul"] #=> "john and paul" |
| to\_sentence [1, "paul", 3, "ringo"] #=> "1, paul, 3 and ringo" |
|  |
|  |
| # ======================================================================================== |
| # Problem 2 - `mean, median` |
|  |
| # implement methods "mean", "median" on Array of numbers |
| **def mean(ary)**  **ary.inject(0) {|sum,x| sum += x} / ary.size**  **end** |
|  |
| **def median(ary)**  **m\_pos = ary.size / 2**  **return ary.size % 2 == 1 ? ary[m\_pos] : mean(ary[m\_pos-1..m\_pos])**  **end** |
|  |
|  |
|  |
|  |
|  |
| # Your method should generate the following results: |
| mean [1, 2, 3] #=> 2 |
| mean [1, 1, 4] #=> 2 |
|  |
| median [1, 2, 3] #=> 2 |
| median [1, 1, 4] #=> 1 |
|  |
|  |
| # ======================================================================================== |
| # Problem 3 - `pluck` |
|  |
| # implement method `pluck` on array of hashes |
|  |
| **def pluck(records)**  **records =[**  **{ :name=>"John", :instrument=>"guitar" },**  **{ :name=>"Paul", :instrument=>"bass" },**  **{ :name=>"George", :instrument=> "guitar" },**  **{ :name=>"Ringo", :instrument=>"drums" }**  **]**  **end**   1. **records.map{|h| h[:name]}** 2. **records.map{|h| h[:instrument]}** |
|  |
|  |
| # Your method should generate the following results: |
| records = [ |
| {name: "John", instrument: "guitar"}, |
| {name: "Paul", instrument: "bass" }, |
| {name: "George", instrument: "guitar"}, |
| {name: "Ringo", instrument: "drums" } |
| ] |
| pluck records, :name #=> ["John", "Paul", "George", "Ringo"] |
| pluck records, :instrument #=> ["guitar", "bass", "guitar", "drums"] |
|  |
|  |
| # ======================================================================================== |
| # Problem 4 - monthly bank statement |
|  |
| # given a CSV file with bank transactions for a single account (see assignment02-input.csv) |
| # generate an HTML file with a monthly statement |
|  |
| # assume starting balance is $0.00 |
|  |
| # the monthly statement should include the following sections: |
| # - withdrawals |
| # - deposits |
| # - daily balance |
| # - summary: |
| # - starting balance, total deposits, total withdrawals, ending balance  **def create\_table(date, payee, peposits, withdrawals, dbalance)**  **{date: "Date", payee: "Payee", deposit: "Deposits", withdrawals: "Withdrawals", dbalance: "Daily Balance"}**  **end**  **File.open("assignment02-input.csv") do |input|**  **records = input.readlines.map do |line|**  **fields = line.split ","**  **table = create\_table fields[0], fields[1], fields[2], fields[3], fields[4]**  **end**  **end**  **def starting\_balance(s)**  **s = 0**  **end**  **def total\_deposits(t)**  **t = deposits.reduce {|deposits, acc| deposits + acc}**  **end**  **def total\_withdrawals(w)**  **w = withdrawals.reduce {|withdrawels, acc| withdrawels + acc}**  **end**  **def daily\_balance(b)**  **b = (s+t)-w**  **end**  **def render\_html(title, records)**  **<<HTML**  **<!doctype html>**  **<html>**  **def render\_head(title)**  **<<HEAD**  **<head>**  **<title> December Bank Statement </title>**  **</head>**  **HEAD**  **end**  **def render\_body(title, records)**  **<<BODY**  **<body>**  **<h1>Transaction Records</h1>**  **def render\_records(records)**  **<<RECORDs**  **<table>**  **def render\_record(r)**  **<<RECORD**  **<tr>**  **<td>#{r[:date]}</td>**  **<td>#{r[:payee]}</td>**  **<td>#{r[:deposits]}</td>**  **<td>#{r[:withdrawals]}</td>**  **<td>#{r[:dbalance]}</td>**  **</tr>**  **RECORD**  **end**  **RECORDS**  **end**  **<h2>SUMMARY</h2>**  **<table>**  **def render\_record(s)**  **<<RECORD**  **<tr>**  **<tr><th>Starting Balance</th><th>Total Deposits</th><th>Total Withdrawals</th><th>Ending Balance}</th></tr>**  **<tr><td>#[s]</td><td>#[t]</td><td>#r[w]</td><td>#r[b]</td>**  **</tr>**  **RECORD**  **end**  **</body>**  **BODY**  **end**  **</html>**  **HTML**  **end** |