Homework 1: due by 11:59pm on Wednesday, February 1, 2023 (Total: 100 points)

Mack Joyner

Commit all work to the github classroom hw1 repo at https://classroom.github.com/a/HPg2ZvTr that we created for you. In case of problems committing your files, please contact the teaching staff at comp322-staff@mailman.rice.edu before the deadline to get help resolving your issues.

Your solution to the written assignment should be submitted as a PDF file named `hw1_written.pdf` in your github classroom hw1 top level directory. This is important — you will be penalized 5 points if you place the file in some other folder or with some other name. The PDF file can be created however you choose. If you scan handwritten text, make sure that the writing is clearly legible in the scanned copy. Your solution to the programming assignment should be submitted in the appropriate location in the hw1 directory.

The slip day policy for COMP 322 is similar to that of COMP 321. All students will be given 3 slip days to use throughout the semester. When you use a slip day, you will receive up to 24 additional hours to complete the assignment. You may use these slip days in any way you see fit (3 days on one assignment, 1 day each on 3 assignments, etc.). If you plan to use a slip day, you need to say so in an github committed README.md file before the deadline. You should specifically mention how many slip days you plan to use. The README.md file should be placed in the top level directory (e.g. hw1). Other than slip days, no extensions will be given unless there are exceptional circumstances (such as severe sickness, not because you have too much other work). Such extensions must be requested and approved by the instructor (via e-mail, phone, or in person) before the due date for the assignment. Last minute requests are likely to be denied.

If you see ambiguity or inconsistency in a question, please seek clarification on Piazza (remember not to share homework solutions in public posts) or from the teaching staff. If it is not resolved through those channels, you should state the ambiguity/inconsistency that you see, as well as any assumptions that you make to resolve it.

Honor Code Policy: All submitted homework is expected to be the result of your individual effort. You are free to discuss course material and approaches to problems with your other classmates, the teaching assistants and the instructors, but you should never misrepresent someone else's work as your own. If you use any material from external sources, you must provide proper attribution.

1 Written Assignment (15 points total)

As mentioned earlier, your solution to the written assignment should be submitted as a PDF file named ***hw1_written.pdf*** *in the hw1 directory.* Failure to do this will result in a loss of points.

1.1 Recursive Fibonacci (15 points total)

Consider the Java code shown below in Listing 1 to compute the Fibonacci function using recursion. This is a very intuitive recursive algorithm: fib(n) = fib(n-1) + fib(n-2). Unfortunately, it is not very efficient, as fib(k) will get called multiple times when computing fib(n), for all $k < n$.

1.1.1 Recursive Fibonacci Complexity (5 points)

What is the formula (exact answer) for the *total work* performed by a call to fib(n)? Assume that a single call to fib() (without any recursive calls inside) has a total WORK of 1. Include an explanation of the analysis, and state what expression you get for WORK(n) as a function of n. The exact answer should not be recursive (in terms of WORK).

```
1 public class RecursiveFib {
2   public static int fib(int n) {
3     if (n <= 0) return 0;
4     else if (n == 1) return 1;
5     else return fib(n - 1) + fib(n - 2);
6   }
7 }
```

Listing 1: Recursive Fibonacci

1.1.2 Memoized Fibonacci Complexity (10 points)

Now, you are given a different Java code in (Listing 2) that uses memoization (our Lazy class from lectures) to avoid the unecessary work when calling fib.

- (5 points) What is the big-O formula for the WORK as a function of n performed by a call to $fib(n)$ (where $n < MaxMemo$) for the very first time? In other words, there were no calls to fib at all, then a call to $fib(n)$ happens. How much work will it perform? Again, assume that a single call to fib() (without any recursive calls inside) has a total WORK of 1.
- (5 points) After a long series of calls to $fib(k_1)$, $fib(k_2)$,. . . $fib(k_m)$, with k_1, k_2, . . . k_m being random values between 0 and $MaxMemo$, what will be the big-O for the expected WORK performed by a call to $fib(n)$ (where $n < MaxMemo$)?

```
1  public class MemoizationFib {
2    private static final int MaxMemo = 1000000; // max memoized results
3    private static final Lazy<Integer>[] memoized = // array of memos
4      IntStream.range(0, MaxMemo)// Stream (0, 1, 2, ... MaxMemo-1)
5     .mapToObj(e->Lazy.of(()->fib(e)))// Stream of memos to compute
6                                      // (fib(0), fib(1), ... fib(MaxMemo-1))
7     .toArray(Lazy[]::new); // convert to array
8
9    public static int fib(int n) {
10     if (n <= 0) return 0;
11     else if (n == 1) return 1;
12     else if (n >= MaxMemo) return fib(n - 1) + fib(n - 2);
13     else return memoized[n-1].get() + memoized[n-2].get();
14   }
15 }
```

Listing 2: Memoized Fibonacci

2 Programming Assignment (85 points)

2.1 Functional Programming: Trees

2.1.1 Provided

In this part of the homework we provide you with an implementation of generic functional trees, similar to the *GList* generic functional lists used in lectures and Lab 2. This implementation can be found in *provided/trees*. Write your solutions to problems 1, 2, and 4 in the TreeSolutions file, so that you can check your solutions using the TreeSolutionsTest test suite. The solution to problem 3 is provided. You will need to add your lab 2 solution for tree *map* and *filter*.

Functional Trees

1. Write a recursive sum function to calculate the sum of the values in all of the nodes in a `Tree<Integer>`. You are not allowed to use any higher order functions, mutaton or loops.
2. Calculate the same sum using the higher order `GList` functions *map*, *fold*, and *filter*. These functions can be found in the *GList* implementation.
3. Complete the implementation of the higher order *fold* function for trees. This is already implemented!
4. Use the implementation of fold above to calculate the sum of the tree's nodes.

2.2 Java Streams: Sales Data

2.2.1 Provided

In this homework, we have provided you with a database of *Products*, *Orders*, and *Customers*. **Customers can place multiple orders and each order can contain a number of products with varying discounts**. All the information is already loaded for you in the code we have provided, and is inside 3 Java collections: *CustomerRepo*, *OrderRepo*, and *ProductRepo*. We used the Spring Framework (https://spring.io/projects/spring-framework) to load all the data into these collections. You are welcome to read up on the Spring Framework if you are interested, but you are not required to learn anything about it in order to complete this homework. All you need to know is how to extract information from these collections, which you can do by calling the `findAll()` method on the repositories, which returns an `Iterable` object. You can create a stream from that iterable object by calling `stream()` method on it. For example:

```
productRepo.findAll().stream()
```

This will create a Java Stream of all the products, which you can further process to compute the answers required in this homework. Similarly, you can create Java Streams from the *orderRepo*, and *customerRepo*.

2.2.2 Problems

For each of the following problems, utilize the provided data repositories and Java Streams API to create both sequential and a parallel functional solution to each. In doing this, you will hopefully be able to see differences in execution time between parallel and sequential executions of the same code. Write your solutions to problems 1-8 in the StreamSolutions, so that you can check your solutions using the StreamSolutionsTest test suite.

Stream Operations

1. Calculate the companies maximum possible revenue from all online sales during the month of February. In other words, calculate revenue assuming that all buyers paid full price for their products.
2. Get the order IDs of the 5 most recently placed orders.
3. Count the number of distinct customers making purchases.
4. Calculate the total discount for all orders placed in March 2021. For this problem, total discount is the total difference between the discount price and the full price for all items purchased during this time period. For example, if 3 items were purchased: item A for $5, B for $3, and C for $15, and the original prices were $7, $5, and $15 respectively, the solution would be $4: (7-5)+(5-3)+(15-15). **Hint**: the provided Discount object will be helpful for calculating the discount on an item.

Data Map Creation

5. Create a mapping between customers IDs and the total dollar amount they spent on products, assuming they purchased all products at full price.
6. Create a mapping between product categories and the average cost of an item in that category.
7. Create a mapping between products IDs in the tech category (category = "Tech") and the IDs of the customers which ordered them.
8. An advertising firm would like to target ads for product sales to customers without membership tiers (tier = 0) based on the customer's sale utilization. The company defines sale utilization as the total percentage of purchases a customer makes at a discounted rate. For example, if a customer purchased three items, two of which were on sale, this customer would have a sales utilization rate of $\frac{2}{3} = 66.6\%$. Create a mapping between the IDs of customers without membership tiers and their sales utilization rate. **Hint**: the provided Discount object will be helpful for calculating whether or not an object is discounted.

2.3 Submission

Your submission should include the following in the hw_1 directory:

1. (60 points) Complete solutions for the 12 programming problems given above.
2. (10 points) 10 points will be allocated for coding style and documentation. We have provided the basic checkstyle rules as part of your Maven project. At a minimum, all code should include basic documentation for each method in each class. You are also welcome to add additional unit tests to test corner cases and ensure the correctness of your implementation.
3. (15 points) A report file formatted as a PDF file named `hw1_report.pdf` in the hw1 top level directory. The report should contain the following:
   (a) A description of the machine on which you were running your code
   (b) A table comparing the execution time of your solutions when using Java Streams vs. parallel Java Streams
   (c) A discussion of the results you obtained. Is the performance of the parallel Java Streams when compared to the Java Streams what you expected on the machine that you were running? If not, why do you think that is the case?

3 Submitting Your Assignment

You will need to add, commit, and push all work to the github classroom hw1 repository at https://classroom.github.com/a/HPg2ZvTr. Please open a browser, navigate to the url, and verify that you successfully committed your homework. Don't forget to modify the README.md file if you plan to use slip days.