Recitation 10

* Reciation git repo: https://github.com/21fa-cit591/recitation10.git

Topics Recap

1. Recursion
   - Concepts
     - A function that calls itself repeatedly
     - A recursive algorithm is an algorithm that breaks the problem into smaller subproblems and applies the same algorithm to solve the smaller subproblems.
     - The base case is the case where a recursive algorithm actually does some final work--grading the one exam in the previous case.
   - Key concepts
     - Base-case -- what is the simplest form of your problem?
     - Sub-problems -- how to break the problem in to smaller problem?
   - Appplication
     - Math problem like GCD
     - Tree/ Graph traversal
     - Pathfinding/ Depth-First Search
2. Design Choices
   - Don't Repeat Yourself (D.R.Y.)
     - Keep things simple
     - Example in class: validateTransaction() helper method in Account class
   - (High) Cohension
     - The degree to which all features of a public interface are related to the single concept that the class represents.
     - A class should represent a single concept
     - High cohesion -> better design -> better readability, tesitability -> easier to debug
     - Example in class PaymentSystem and Employeee
   - (Low) Coupling
     - Each class should depend on as few others as possible in order to fulfill its purposes
     - Low Coupling -> less dependency -> less change needed + less impact of changes

---

Recap Example

In RecapExample.java, there are 2 methods to be implemented. We will be going through these examples during recitation.

1. Use Recursion to Print a String in reverse order
2. Use Recursion to count the number of digits in a positive integer

---

Recitation Problem Set

Complete (either Q1-Q3 or Q4-Q5) + Q6 in ProblemSet.java.

1. Implement factorial() using Recursion
2. Print a number pattern using Recursion
3. Reverse a String using Recursion
4. Count the number of '1' in the bit representation of a int, using Recursion
5. Find all the permutation of an arraylist of numbers, using Recursion
6. Use D.R.Y. principle to improve a method's readability and testability

What to Submit?

Submit (Q6 + Q1-Q3) or (Q6 + Q4-Q5) Copy and paste your code into gradscopes.

Group Assignment

Group #	Member 1	Member 2	Member 3	Member 4
Group 1	Kim, Yunchae	Li, Yunhe	Ye, Huifang	Arguello-Gonzalez, Marcos Abraham
Group 2	Cheema, Sardar Asfandy	Gallagher, John Manus	Kung, Ling-Hsin	Wang, Kehan
Group 3	Yu, Qingyu	Sheng, Xinyue	Mammadov, Elmar	Patel, Rishi
Group 4	Zhang, Zhihui	Bales, Elijah	Richmond, Christian	Biscaro, Denise
Group 5	Cai, Jialin	Sabri, Rita	Chen, Zheyi	Bernat, Kevin Bruno

Group #	Member 1	Member 2	Member 3	Member 4
Group 6	Qiu, Xi	Pace, Benjamin Michael	Ren, Yue	Patel, Siddharth Bhagwanji
Group 7	He, Donglun	Xiao, Zijian	Kong, Rachel	Yiu, Hon-Cheung
Group 8	Chen, Xiyue	Thenappan, Bala Sundar	Graham, Alexander Richard	Shah, Rushabh
Group 9	Pizzico, Tyler R	Hu, Lucy Qian	Zhang, Yang	Nguyen, Tai D
Group 10	Sha, Yumeng	Lim, Xi Zhen	Tims, George	Kallas Jatene, Rafael
Group 11	Rigas, Andrew	Zhang, Minzheng	Choi, Jae Ho	Carnation, Kayla Rae
Group 12	Liu, Jiayun	Pinheiro, Benjamin B	Jiang, Yao	Guo, Zhaosen
Group 13	Zhang, Yihong	Lee, Jaeyoung	Xue, Mingxin	Williams, Levester Randall
Group 14	Liu, Xinyue	Zhang, Miaoyan	Wang, Yuanqi	Chou, Randy
Group 15	Qiu, Chengzhuo	Ng, Wai Chung	Hu, Yuxin	Wu, Jeng-Ru
Group 16	Wang, Liujia	Huang, Wenyi	Wang, An-Jie	He, Ziyi
Group 17	Cho, Suebin Grace	Lai, Qimei	Liu, Shufan	Cruz, Marye I
Group 18	Schnall, Aaron Hewitt	Zhang, Han	Chheda, Shagun Pritesh	Nojoomi, Radin