Garbage collection

Subject: CSW2(CSE3141) **Section: All**

Session: Feb 2024 to April 2024 **Branch: CSE&CSIT**

……

1. Write a program that initializes a lot of objects in a loop and observe how much time it takes to crash the program.
2. Create a diagram that shows how objects become eligible for garbage collection.
3. How do we make objects eligible for garbage collection? Discuss all possible ways along with their implementation.
4. What distinguishes **anonymous objects** from regular objects? Demonstrate how anonymous objects can be used to successfully call and run methods
5. Demonstrate how garbage collectors can be used to remove anonymous objects from the heap.
6. You are tasked with designing a memory management system for a Java application that heavily utilizes anonymous objects. Your goal is to implement a robust garbage collection mechanism to ensure efficient memory usage and prevent memory leaks caused by lingering anonymous objects.
7. Design a **class** contains **private data members** of integer and double type along with methods. Create two objects which call the required method to initialize/set/update the data members of the defined class. Use the **Runtime class** to calculate the **total memory** allocated and the memory occupied by the objects. Use any one technique as you know to make objects unreachable, hence eligible for garbage collection. At the end, call **Runtime class** again to recheck the **utilized/total memory**.
8. Does JVM guarantees that upon invoking **gc()** method, objects will be eligible for immediate garbage collection? Justify your answer through code.
9. How do you define generational garbage collection? Also discuss how generational garbage collection works along with proper heap map.
10. Implement a simplified version of generational garbage collection in Java. Generational garbage collection divides objects into different generations based on their age and aims to optimize memory management by applying different collection strategies to each generation.