Garbage Collection

Subject: CSW2(CSE3141)

Session: Feb 2024 to April 2024

Branch: CSE&CSIT

Section : All

Q1. Write a Java program illustrating garbage collection through an **UnreachableObject** class. This class will include a constructor initializing an object with a given name, a **show()** method creating an instance of **UnreachableObject** class and then invoking **display()**, a **display()** method creating an **UnreachableObject** instance, and a **main()** method calling **show()** followed by an explicit invocation of the garbage collector. Additionally, the program will override the **finalize()** method to print the object's name upon successful garbage collection.

Q2. Develop a Java program showcasing reference reassignment and garbage collection using the **ReassigningReference** class. This class features a constructor initializing an object with a given name. In the **main()** method, two instances of **ReassigningReference** are created. Then, the reference is reassigned. Subsequently, the garbage collector is explicitly invoked. Additionally, the program overrides the **finalize()** method to print the name of the object upon successful garbage collection.

Q3. Write a Java program illustrating nullification of references and garbage collection using the **NullifiedReference** class. This class comprises a constructor initializing an object with a provided name. Within the **main()** method, an instance of **NullifiedReference** is created and assigned, followed by a nullification of the object reference. Subsequently, the garbage collector is explicitly invoked. Furthermore, the program overrides the **finalize()** method to print the name of the object upon successful garbage collection.

Q4. Create a Java program demonstrating anonymous objects and garbage collection with the **AnonymousObject** class. It includes a constructor initializing an object with a name. In **main()**, an anonymous object is created, and the garbage collector is invoked. The **finalize()** method prints the object's name upon successful garbage collection.

Q5. Develop a Java class containing private data members of integer and double types, along with methods for initializing, setting, and updating these data members. Create two objects of this class, each calling the necessary methods to set or update the data members. Utilize the Runtime class to calculate the total memory allocated and the memory occupied by the objects. Employ any technique to make objects unreachable, hence eligible for garbage collection. Finally, recheck the utilized and total memory using the Runtime class.

Q6. Write a memory-intensive program which creates a lot of objects. Try G1 collector on this program. Print timestamp and heap size. Use the following commands to print the heap size and free space.

Command to print total memory of heap:

Runtime.getRuntime().totalMemory();

Command to print free memory of heap:

Runtime.getRuntime().freeMemory();

Q7. Create a Java program for university student enrollment. Use a Student class for course management and student information. Implement efficient garbage collection for memory management. Utilize Runtime class to monitor memory usage. Override finalize() method to print a message on successful garbage collection.