**# Java-Practice**

**1. Basics of Java**

* Write a program to print "Welcome to Java Programming".
* Write a program to add, subtract, multiply, and divide two numbers.
* Write a program to calculate the area of a rectangle, circle, and triangle.
* Write a program to find the square and cube of a number.
* Write a program to convert temperature from Celsius to Fahrenheit and vice versa.
* Write a program to find the greatest of three numbers.
* Write a program to print numbers from 1 to 100.
* Write a program to check if a number is odd or even.
* Write a program to check if a number is positive, negative, or zero.
* Write a program to calculate the compound interest.
* Write a program to find the ASCII value of a character.
* Write a program to demonstrate type casting (implicit and explicit).
* Write a program to calculate the perimeter of a square and rectangle.
* Write a program to swap two numbers using a third variable.
* Write a program to swap two numbers without using a third variable.
* Write a program to calculate the average of three numbers.
* Write a program to check whether a year is a leap year.
* Write a program to print multiplication tables from 1 to 10.
* Write a program to reverse a number.
* Write a program to calculate the factorial of a number using recursion.
* Write a program to find the sum of all digits in a number.
* Write a program to find the greatest common divisor (GCD) of two numbers.
* Write a program to find the least common multiple (LCM) of two numbers.
* Write a program to check if a number is a palindrome.
* Write a program to print the Fibonacci series up to n terms.

**2. Control Statements**

* Write a program to find the largest of three numbers using if-else.
* Write a program to check if a number is divisible by both 5 and 11.
* Write a program to implement a calculator using switch-case.
* Write a program to check if a character is a vowel or consonant using if-else.
* Write a program to print all even numbers from 1 to 50.
* Write a program to print all odd numbers between 1 and 50.
* Write a program to find the sum of natural numbers up to n.
* Write a program to find the factorial of a number using loops.
* Write a program to reverse a given number using a loop.
* Write a program to find the sum of digits in a number using a loop.
* Write a program to print the multiplication table for a given number.
* Write a program to display the prime numbers between 1 and 100.
* Write a program to print the first n terms of the Fibonacci series.
* Write a program to display numbers divisible by both 3 and 5 between 1 and 100.
* Write a program to find the GCD of two numbers using a loop.
* Write a program to check if a number is a perfect number.
* Write a program to print the reverse of a string using a loop.
* Write a program to count the number of digits in an integer.
* Write a program to generate a pyramid pattern using nested loops.
* Write a program to print Floyd’s triangle.
* Write a program to implement the star pattern:

markdown

Copy

Edit

\*

\*\*

\*\*\*

\*\*\*\*

* Write a program to calculate the power of a number using a loop.
* Write a program to find the largest and smallest digits in a number.
* Write a program to find all Armstrong numbers between 1 and 500.
* Write a program to display the first 10 terms of a geometric progression.

**3. Arrays**

* Write a program to store and print 10 integers using an array.
* Write a program to calculate the sum of elements in an array.
* Write a program to find the largest and smallest numbers in an array.
* Write a program to reverse an array.
* Write a program to count occurrences of a number in an array.
* Write a program to sort an array in ascending order.
* Write a program to sort an array in descending order.
* Write a program to merge two arrays.
* Write a program to remove duplicate elements from an array.
* Write a program to find the second largest element in an array.
* Write a program to find the maximum product of two integers in an array.
* Write a program to find the intersection of two arrays.
* Write a program to rotate an array to the left by a given number of positions.
* Write a program to rotate an array to the right by a given number of positions.
* Write a program to find the union of two arrays.
* Write a program to check if an array is sorted in ascending order.
* Write a program to find all pairs in an array with a given sum.
* Write a program to implement binary search on an array.
* Write a program to find the missing number in an array of size n.
* Write a program to find the majority element in an array.
* Write a program to rearrange an array such that even and odd numbers alternate.
* Write a program to check if two arrays are equal.
* Write a program to find the difference between the maximum and minimum elements in an array.
* Write a program to split an array into two halves.
* Write a program to move all zeros in an array to the end.

**4. Strings**

* Write a program to find the length of a string without using the length() method.
* Write a program to reverse a string without using built-in methods.
* Write a program to check if a string is a palindrome.
* Write a program to count vowels, consonants, and spaces in a string.
* Write a program to check if two strings are anagrams.
* Write a program to find all substrings of a string.
* Write a program to count the frequency of each character in a string.
* Write a program to remove all white spaces from a string.
* Write a program to capitalize the first letter of every word in a string.
* Write a program to replace all vowels in a string with a specific character.
* Write a program to reverse each word in a sentence.
* Write a program to find the longest word in a sentence.
* Write a program to convert a string to uppercase without using toUpperCase().
* Write a program to find the number of words in a string.
* Write a program to find all permutations of a string.
* Write a program to check if a string contains only digits.
* Write a program to remove duplicate characters from a string.
* Write a program to sort characters in a string alphabetically.
* Write a program to extract all numbers from a string.
* Write a program to check if two strings are rotations of each other.
* Write a program to find the first non-repeating character in a string.
* Write a program to find the longest palindrome in a string.
* Write a program to implement a basic Caesar cipher for encryption.
* Write a program to count the number of times a substring appears in a string.
* Write a program to reverse a string recursively.

**5. Object-Oriented Programming (OOP)**

* Write a program to create a class Person with attributes name and age. Create objects and display their attributes.
* Write a program to demonstrate the use of constructors.
* Write a program to create a class Rectangle with methods to calculate area and perimeter.
* Write a program to implement method overloading with different parameter types.
* Write a program to demonstrate inheritance with a base class Animal and derived classes Dog and Cat.
* Write a program to override a method from the parent class.
* Write a program to demonstrate polymorphism using method overriding.
* Write a program to implement an abstract class Shape with subclasses Circle and Rectangle.
* Write a program to demonstrate the use of interfaces.
* Write a program to create a simple banking system using encapsulation.
* Write a program to demonstrate the this keyword with constructors and methods.
* Write a program to demonstrate the super keyword in inheritance.
* Write a program to create a class Employee and implement a copy constructor.
* Write a program to create a static method in a class and call it without creating an object.
* Write a program to demonstrate the final keyword with variables, methods, and classes.
* Write a program to demonstrate the concept of multiple inheritance using interfaces.
* Write a program to calculate the area of different shapes using dynamic method dispatch (polymorphism).
* Write a program to implement composition (Has-A relationship).
* Write a program to implement aggregation in a class.
* Write a program to demonstrate the singleton design pattern.
* Write a program to create a Student class and maintain a list of students with getter and setter methods.
* Write a program to demonstrate how instanceof works.
* Write a program to implement the builder design pattern.
* Write a program to implement the factory design pattern.
* Write a program to create an immutable class.

**6. Exception Handling**

* Write a program to handle division by zero using try-catch.
* Write a program to catch multiple exceptions in a single try-catch block.
* Write a program to demonstrate the use of finally in exception handling.
* Write a program to throw a user-defined exception.
* Write a program to create a custom exception class and handle it.
* Write a program to demonstrate the use of the throw keyword.
* Write a program to demonstrate the throws keyword.
* Write a program to catch and display NumberFormatException.
* Write a program to handle an ArrayIndexOutOfBoundsException.
* Write a program to demonstrate NullPointerException handling.
* Write a program to handle InputMismatchException.
* Write a program to demonstrate nested try-catch blocks.
* Write a program to demonstrate the concept of checked exceptions.
* Write a program to demonstrate the concept of unchecked exceptions.
* Write a program to catch exceptions using the Exception superclass.
* Write a program to re-throw an exception after catching it.
* Write a program to demonstrate the finally block when an exception is not thrown.
* Write a program to create chained exceptions.
* Write a program to read a file and handle file not found exceptions.
* Write a program to handle exceptions using try-with-resources.
* Write a program to demonstrate how exceptions can be propagated in Java.
* Write a program to validate user input using exceptions.
* Write a program to demonstrate ArithmeticException.
* Write a program to implement a retry mechanism using exceptions.
* Write a program to log exceptions to a file.

**7. Multithreading**

* Write a program to create a thread by extending the Thread class.
* Write a program to create a thread by implementing the Runnable interface.
* Write a program to demonstrate thread priority.
* Write a program to create multiple threads and execute them simultaneously.
* Write a program to demonstrate the sleep() method.
* Write a program to demonstrate the join() method.
* Write a program to demonstrate synchronization using the synchronized keyword.
* Write a program to solve the producer-consumer problem using threads.
* Write a program to demonstrate inter-thread communication using wait(), notify(), and notifyAll().
* Write a program to demonstrate the Thread states (New, Runnable, Blocked, Waiting, Timed Waiting, Terminated).
* Write a program to implement a thread pool using the ExecutorService.
* Write a program to use the Callable interface and Future.
* Write a program to demonstrate the ReentrantLock class.
* Write a program to demonstrate a deadlock scenario.
* Write a program to prevent deadlocks.
* Write a program to demonstrate a race condition and solve it using synchronization.
* Write a program to demonstrate thread-safe collections like ConcurrentHashMap.
* Write a program to demonstrate ThreadLocal.
* Write a program to create a daemon thread.
* Write a program to terminate a thread gracefully.
* Write a program to implement parallel processing using multiple threads.
* Write a program to use the Semaphore class.
* Write a program to simulate a countdown using CountDownLatch.
* Write a program to demonstrate the use of CyclicBarrier.
* Write a program to demonstrate the use of the ScheduledExecutorService.

**8. Data Structures**

* Write a program to implement a stack using an array.
* Write a program to implement a queue using an array.
* Write a program to implement a linked list.
* Write a program to reverse a linked list.
* Write a program to implement a doubly linked list.
* Write a program to implement a circular linked list.
* Write a program to implement a stack using a linked list.
* Write a program to implement a queue using a linked list.
* Write a program to implement a binary search tree (BST).
* Write a program to traverse a BST using in-order, pre-order, and post-order traversal.
* Write a program to implement a priority queue.
* Write a program to implement a hash table.
* Write a program to implement graph traversal using BFS.
* Write a program to implement graph traversal using DFS.
* Write a program to detect a cycle in a directed graph.
* Write a program to find the shortest path using Dijkstra's algorithm.
* Write a program to solve the N-Queens problem.
* Write a program to implement a min-heap.
* Write a program to implement a max-heap.
* Write a program to implement a trie and search for a word.
* Write a program to implement the bubble sort algorithm.
* Write a program to implement the merge sort algorithm.
* Write a program to implement the quick sort algorithm.
* Write a program to implement the insertion sort algorithm.
* Write a program to implement the selection sort algorithm.

**9. File Handling**

* Write a program to create a new file.
* Write a program to write text to a file.
* Write a program to read text from a file.
* Write a program to append data to an existing file.
* Write a program to delete a file.
* Write a program to count the number of lines, words, and characters in a file.
* Write a program to copy contents from one file to another.
* Write a program to rename a file.
* Write a program to check if a file exists.
* Write a program to read and write binary data to a file.
* Write a program to list all files in a directory.
* Write a program to find the largest file in a directory.
* Write a program to read data from a CSV file.
* Write a program to write data to a CSV file.
* Write a program to serialize an object to a file.
* Write a program to deserialize an object from a file.
* Write a program to encrypt and decrypt text in a file.
* Write a program to compress a file.
* Write a program to extract data from a compressed file.
* Write a program to handle file not found exceptions.
* Write a program to read a file line by line using a BufferedReader.
* Write a program to read a file character by character using a FileReader.
* Write a program to demonstrate try-with-resources for file handling.
* Write a program to calculate the checksum of a file.
* Write a program to create a log file and write logs to it.

**10. Generics**

* Write a generic class Box that can store and retrieve any type of object.
* Write a program to demonstrate a generic method that accepts two arguments of the same type and swaps their values.
* Write a program to implement a generic stack class.
* Write a program to create a generic Pair class with two type parameters.
* Write a program to demonstrate bounded type parameters for generics (e.g., <T extends Number>).
* Write a program to demonstrate the wildcard (?) in generics.
* Write a program to use the extends wildcard to restrict types in generics.
* Write a program to use the super wildcard in generics.
* Write a program to implement a generic method to find the maximum of three numbers.
* Write a program to create a generic interface.
* Write a program to demonstrate the use of multiple bounds in generics (e.g., <T extends ClassA & InterfaceB>).
* Write a program to use generics with collections (e.g., ArrayList<T>).
* Write a program to demonstrate raw types in generics and why they should be avoided.
* Write a program to implement a generic queue.
* Write a program to use a bounded generic type to restrict input to Comparable types only.
* Write a program to demonstrate the use of Generic Arrays.
* Write a program to demonstrate type erasure in generics.
* Write a program to use reflection to analyze a generic class.
* Write a program to use generics with multiple type parameters (e.g., <T, U>).
* Write a program to create a generic linked list.
* Write a program to implement a generic binary search tree (BST).
* Write a program to sort a list of generic objects using a Comparator.
* Write a program to demonstrate how to create and use generic methods in a utility class.
* Write a program to demonstrate covariance and contravariance with generics.
* Write a program to demonstrate the difference between bounded and unbounded wildcards.

**11. Reflection**

* Write a program to get the name of a class at runtime using reflection.
* Write a program to list all methods of a class using reflection.
* Write a program to list all constructors of a class using reflection.
* Write a program to get and set the value of a private field using reflection.
* Write a program to call a private method of a class using reflection.
* Write a program to demonstrate the use of getDeclaredMethods() and getMethods().
* Write a program to find the superclass of a class using reflection.
* Write a program to dynamically create an object using reflection.
* Write a program to check if a class implements an interface using reflection.
* Write a program to invoke a method with parameters using reflection.
* Write a program to get all annotations of a class using reflection.
* Write a program to demonstrate the use of Field, Method, and Constructor classes.
* Write a program to analyze the modifiers (public, private, static, etc.) of a class using reflection.
* Write a program to get the type of a field in a class using reflection.
* Write a program to list all declared fields in a class using reflection.
* Write a program to determine whether a given method is static using reflection.
* Write a program to load a class dynamically using Class.forName().
* Write a program to implement a custom annotation and access it using reflection.
* Write a program to measure the execution time of a method using reflection.
* Write a program to demonstrate the use of Proxy classes with reflection.
* Write a program to implement dependency injection using reflection.
* Write a program to demonstrate how to access generic type parameters at runtime using reflection.
* Write a program to create a class loader using reflection.
* Write a program to find all interfaces implemented by a class using reflection.
* Write a program to create a dynamic proxy for an interface using reflection.

**12. Java Streams**

* Write a program to filter all even numbers from a list using streams.
* Write a program to find the maximum and minimum values in a list using streams.
* Write a program to map a list of strings to their lengths using streams.
* Write a program to collect all unique elements from a list using streams.
* Write a program to count the number of elements greater than 10 in a list using streams.
* Write a program to sort a list of integers in ascending and descending order using streams.
* Write a program to find the sum and average of a list of integers using streams.
* Write a program to demonstrate the use of the reduce() function.
* Write a program to generate an infinite stream of even numbers.
* Write a program to find the first non-repeating character in a string using streams.
* Write a program to group a list of words by their first letter using Collectors.groupingBy().
* Write a program to partition a list of numbers into even and odd using Collectors.partitioningBy().
* Write a program to flatten a list of lists using streams.
* Write a program to remove duplicates from a list using streams.
* Write a program to count the frequency of each word in a list using streams.
* Write a program to check if all elements in a list are positive using streams.
* Write a program to filter a list of employees based on their salary using streams.
* Write a program to find the second highest number in a list using streams.
* Write a program to concatenate all strings in a list using streams.
* Write a program to demonstrate the use of skip() and limit() in streams.
* Write a program to generate the Fibonacci sequence using streams.
* Write a program to collect a list of integers into a set using streams.
* Write a program to calculate the total salary of employees grouped by department using streams.
* Write a program to convert a list of strings to uppercase using streams.
* Write a program to demonstrate the difference between parallelStream() and stream().

**13. Spring Boot**

* Write a Spring Boot application to print "Hello, Spring Boot!".
* Create a REST API to manage a list of students (CRUD operations).
* Create a Spring Boot application to handle GET, POST, PUT, and DELETE requests.
* Write a program to integrate a Spring Boot application with a MySQL database.
* Create a REST API with pagination and sorting.
* Write a program to implement global exception handling using @ControllerAdvice.
* Write a program to secure a REST API using Spring Security.
* Write a program to implement JWT-based authentication in a Spring Boot application.
* Create a REST API to upload and download files.
* Write a program to schedule a task to run at fixed intervals using Spring Scheduler.
* Create a Spring Boot application to send emails using JavaMailSender.
* Write a program to demonstrate dependency injection in Spring Boot.
* Create a Spring Boot application to handle validation using @Valid and @Constraint.
* Write a program to create custom annotations in Spring Boot.
* Write a program to implement caching using Spring Cache.
* Create a REST API to integrate with a third-party API (e.g., consuming a weather API).
* Write a program to use @Transactional for managing database transactions.
* Write a program to demonstrate the use of ApplicationListener in Spring Boot.
* Write a program to implement a custom filter in Spring Boot.
* Create a Spring Boot application to connect to Redis.
* Write a program to demonstrate the use of the @Aspect annotation in Spring Boot.
* Create a Spring Boot application to upload files to Amazon S3.
* Write a program to monitor the performance of a Spring Boot application using Actuator.
* Write a program to configure multiple data sources in a Spring Boot application.
* Create a Spring Boot application to send and receive messages using RabbitMQ.

**14. Hibernate**

* Write a program to configure Hibernate with a MySQL database.
* Write a program to demonstrate the SessionFactory and Session in Hibernate.
* Create a Hibernate application to perform CRUD operations on an entity.
* Write a program to demonstrate one-to-one mapping in Hibernate.
* Write a program to demonstrate one-to-many mapping in Hibernate.
* Write a program to demonstrate many-to-many mapping in Hibernate.
* Write a program to implement inheritance mapping in Hibernate.
* Write a program to demonstrate the use of HQL (Hibernate Query Language).
* Write a program to implement criteria queries in Hibernate.
* Write a program to perform batch processing in Hibernate.
* Write a program to demonstrate lazy and eager loading in Hibernate.
* Write a program to demonstrate cascading in Hibernate.
* Write a program to implement a native SQL query in Hibernate.
* Write a program to demonstrate @Embeddable and @Embedded annotations in Hibernate.
* Write a program to implement optimistic locking in Hibernate.
* Write a program to implement pessimistic locking in Hibernate.
* Write a program to handle @Temporal annotation for date and time in Hibernate.
* Write a program to map a composite key in Hibernate.
* Write a program to integrate Hibernate with Spring Boot.
* Write a program to cache entities using Hibernate's second-level cache.
* Write a program to audit database changes using Hibernate Envers.
* Write a program to validate entities using Hibernate Validator.
* Write a program to handle transactions programmatically in Hibernate.
* Write a program to use the @Query annotation with custom JPQL.
* Write a program to use Hibernate for dynamic table creation.

**15. Java Design Patterns**

* Implement a Singleton pattern in Java and demonstrate thread safety.
* Write a program to implement a Factory design pattern for creating different shapes.
* Write a program to implement an Abstract Factory design pattern.
* Demonstrate the Builder design pattern with a House class.
* Implement the Prototype design pattern and clone an object.
* Write a program to implement the Adapter design pattern.
* Demonstrate the Bridge design pattern with an example.
* Implement the Composite design pattern for a file system structure.
* Write a program to demonstrate the Decorator design pattern.
* Write a program to demonstrate the Facade design pattern.
* Implement the Flyweight design pattern for an object pool.
* Demonstrate the Proxy design pattern with a virtual proxy example.
* Write a program to implement the Chain of Responsibility pattern.
* Implement the Command pattern for a remote control system.
* Write a program to demonstrate the Interpreter design pattern.
* Implement the Iterator pattern to traverse a collection.
* Write a program to demonstrate the Mediator pattern for chat rooms.
* Implement the Memento pattern to save and restore object states.
* Write a program to demonstrate the Observer pattern for event handling.
* Implement the State design pattern to represent different states of a vending machine.
* Write a program to demonstrate the Strategy pattern for sorting algorithms.
* Implement the Template Method pattern with an abstract class.
* Write a program to demonstrate the Visitor design pattern for an object structure.
* Implement the Dependency Injection pattern using constructors and setters.
* Demonstrate the use of the MVC (Model-View-Controller) pattern for a basic application.

**16. Java Functional Programming (Lambdas and Streams)**

* Write a program to filter and print all odd numbers from a list using a lambda expression.
* Write a program to sort a list of strings by their lengths using lambdas.
* Create a custom functional interface and use it with a lambda expression.
* Write a program to implement a basic calculator using lambdas and functional interfaces.
* Use Consumer to print all elements of a list.
* Use Function to convert a list of strings to their lengths.
* Use Predicate to filter out negative numbers from a list.
* Write a program to demonstrate BiConsumer with key-value pairs.
* Write a program to combine two Predicate conditions using and() and or().
* Use Supplier to generate random numbers.
* Write a program to group a list of names by their first letter using streams.
* Write a program to perform a reduction operation to find the sum of integers in a list.
* Use flatMap to flatten a list of lists into a single list.
* Write a program to collect the elements of a stream into a map.
* Write a program to count the occurrences of each word in a string using streams.
* Write a program to demonstrate lazy evaluation in streams.
* Use Optional to handle null values in a stream operation.
* Write a program to parallelize stream operations for performance.
* Demonstrate the use of the takeWhile and dropWhile methods.
* Write a program to calculate the average age of employees grouped by department using streams.
* Write a program to filter duplicate elements from a stream.
* Demonstrate the use of Collectors.toUnmodifiableList().
* Use Stream.iterate() to generate an infinite stream of numbers.
* Write a program to find the longest string in a list using streams.
* Use Collectors.joining() to concatenate strings from a stream.

**17. Java Concurrency and Parallelism**

* Write a program to demonstrate the use of synchronized for thread safety.
* Implement a thread-safe Singleton using synchronized and volatile.
* Demonstrate the use of the Callable and Future interfaces.
* Use ExecutorService to create a thread pool and execute tasks.
* Demonstrate the use of CompletableFuture for asynchronous programming.
* Write a program to handle a deadlock situation and resolve it.
* Use ReentrantLock to implement a thread-safe operation.
* Write a program to demonstrate the use of Semaphore for limiting concurrent access.
* Use CountDownLatch to synchronize threads.
* Write a program to demonstrate the use of CyclicBarrier for thread coordination.
* Use BlockingQueue to implement the producer-consumer problem.
* Demonstrate the use of the Phaser class for advanced thread coordination.
* Write a program to schedule periodic tasks using ScheduledExecutorService.
* Implement a thread-safe linked list using CopyOnWriteArrayList.
* Use ForkJoinPool to implement parallel divide-and-conquer tasks.
* Write a program to demonstrate atomic operations using AtomicInteger.
* Demonstrate the use of ThreadLocal variables.
* Use ExecutorCompletionService to handle multiple tasks.
* Write a program to demonstrate the use of StampedLock.
* Implement a multi-threaded sorting algorithm using parallel streams.
* Use ConcurrentHashMap to handle concurrent data modifications.
* Write a program to demonstrate thread interruptions.
* Write a program to calculate the sum of a large array using parallel computation.
* Use ThreadFactory to create custom-named threads.
* Demonstrate the use of ParallelStream in Java 8 for concurrent processing.

**18. Advanced Java Networking**

* Write a program to create a simple client-server communication using Socket and ServerSocket.
* Write a program to send and receive files over a socket connection.
* Implement a multi-threaded server to handle multiple clients concurrently.
* Write a program to resolve domain names into IP addresses using InetAddress.
* Write a program to send a GET request to a REST API using HttpURLConnection.
* Implement a chat application using Java sockets.
* Write a program to download the content of a web page.
* Demonstrate the use of DatagramSocket and DatagramPacket for UDP communication.
* Write a program to create an HTTP server using Java.
* Use URL and URLConnection classes to fetch data from a URL.
* Write a program to demonstrate multicast communication.
* Implement a simple email client using JavaMail API.
* Write a program to send and receive JSON data over HTTP.
* Demonstrate secure socket communication using SSLServerSocket and SSLSocket.
* Write a program to implement a time server using sockets.
* Use HttpClient to send asynchronous HTTP requests.
* Write a program to upload files to a server using HTTP.
* Implement a simple DNS client in Java.
* Write a program to demonstrate the use of non-blocking I/O with Selector.
* Use WebSocket to implement real-time communication.
* Write a program to handle redirection in HTTP requests.
* Demonstrate the use of proxy servers with Proxy class in Java.
* Write a program to download an image from a URL and save it locally.
* Use HttpClient to handle HTTP headers and cookies.
* Implement an FTP client to upload and download files.

**19. Java Memory Management and Performance**

* Write a program to demonstrate the use of SoftReference and WeakReference.
* Write a program to analyze memory usage using Runtime class methods.
* Demonstrate how to trigger garbage collection using System.gc().
* Write a program to demonstrate the working of finalize() method.
* Use VisualVM or JConsole to monitor memory usage in a Java application.
* Write a program to analyze heap memory allocation.
* Demonstrate the impact of string immutability on memory usage.
* Write a program to use the intern() method for string optimization.
* Write a program to demonstrate object pooling.
* Use ThreadMXBean to monitor thread performance.
* Write a program to demonstrate memory leaks and how to fix them.
* Optimize a program with inefficient ArrayList usage.
* Write a program to demonstrate escape analysis for objects.
* Use a profiler tool to identify performance bottlenecks.
* Write a program to compare the performance of StringBuilder and String.
* Demonstrate the use of JVM flags to monitor memory.
* Write a program to implement custom caching for frequently accessed data.
* Use weak references for implementing a cache to avoid memory leaks.
* Write a program to monitor garbage collection logs.
* Demonstrate the impact of large object creation on performance.
* Optimize a program with poor thread synchronization.
* Write a program to analyze and optimize CPU usage in a multithreaded program.
* Use Java Flight Recorder to monitor application performance.
* Write a program to manage memory consumption for large datasets.
* Optimize a recursive program to avoid stack overflow.