Table of Contents

[1. Java Question 2](#_Toc7427841)

[2. Java Program Questions 4](#_Toc7427842)

[3. Data Structure 5](#_Toc7427843)

[4. Spring/Spring Boot Questions 5](#_Toc7427844)

[5. WebService Questions 6](#_Toc7427845)

[6. Hibernate Questions 6](#_Toc7427846)

[7. Database Questions 6](#_Toc7427847)

[8. SQL 7](#_Toc7427848)

[9. JMS Questions 7](#_Toc7427849)

[10. Design Patterns 7](#_Toc7427850)

[11. Others 8](#_Toc7427851)

12. Angular……………………………………………………………………………………………………..8

13. JavaScript………………………………………………………………………………………………….8

# Java Question

1. What is polymorphism and abstraction?
2. How will you make a transaction is atomic in Java (use atomic variables)?
3. How will you capture the running time of all methods individually and store them somewhere?
4. What is runtime polymorphism?
5. Checked and unchecked exception?
6. User define exception pre-requisites.
7. What are types of exception?
8. How to create user define unchecked exception?
9. Immutable class and how do you achieve and a scenario based example on the same.(An Employee class with two fields , how do you achieve immutability)
10. Immutable class with a reference of list of another class (e.g-List<Address>) which is not immutable. (Hint: inside getAddresses(), we have to create a clone and return that. Do not return this.addresses.
11. About immutable methods or final methods.
12. About hashCode and equals methods.
13. Why Serialization is used in java?
14. In case of serialization, how do you achieve like if a data member should not be serialized (Transient Keyword)
15. How about static, will it be Serialized or not, then what will be the value when you Deserialize the static variable. (What will happen actually, and what value will we get and from where). [static and transient variables do not take part in serialization. Final transient variable will be serialized if it was instantiated at the time of declaration.]
16. Scenario :-
    1. class Employee {
       1. transient String name; ------> (“ABC”)
       2. static int empId; --------- > (100)
       3. final int empSal; --------- > (5000)
       4. }

If the Employee class is serialized, then post deserialization what will be the values of the data members.[name = “”, empId = 0, empSal = 5000]

1. What will happen if a property value is changed before deserialization?
2. What is Serial Version UID? What is the use of it?
3. What is encapsulation? Did u implement encapsulation in your last project? Explain the scenario that you have used encapsulation in your project.
4. What is the significance of abstract classes?
5. What is the feature that you like in java 8, explain why? Have you used that feature in your previous project? Explain the scenario.
6. What is functional interface and its advantage?
7. Explain queue and stack data structures?
8. What is the difference between queue and array list?
9. Write a program to implement a queue and stack data structure.
10. What are the methods in queue data structure in detail?
11. Optional purpose and before java 1.8 how to handle?
12. Create immutable class?
13. Why string pool?
14. Use of functional interface?
15. Difference between anonymous inner class and lambda expression.
16. What is the best way to compare objects?
17. Polymorphism – method Overloading and method Overriding, Static polymorphism?

Class A {

public static String getName()

}

Class B {

Public static String getName()

}

Class Test{

A a = new A(); /// A b = new B();

a.getName(); /// b.getName();

}

Class A{

Public List M1(){}

}

Class B{

Public List M1() {}

Public List M2(){}

}

Test {

A a = new B();

a.M2();

}

1. What is the order of initializing between, static variables, instance variables and a constructor?
2. Sorting --- Comparable and Comparator?
3. Java 7 and 8 features.
4. Enhancement in Exception handling in Java7.
5. What is Polymorphism? What is Overloading and Overriding?
6. Can static method be overridden? i.e. Class A having static method and Class B having same name non static method.
7. Can an instance method be overridden by static? i.e. Class A having non-static method and Class B having same name static method.
8. What is a String Pool?
9. Can we declare a String without it going to string pool?
10. Different ways of creating a String.
11. In which scenario we will create a string using new operator?
12. What is the use of a String Pool?
13. How is string stored in memory?
14. Can an abstract class be marked final?
15. Can a private method be overridden? Can static method be overridden?
16. Explain memory management in java.
17. How the memory gets allocated for a member variable in java?
18. Where the memory will be allocated for reference variables and objects inside a method?
19. Have you ever used finally block? What is the use of it?
20. What is try with resource in java?
21. Exception in overridden method.
22. What is the use of static and default methods in functional interface?
23. What is hashing?
24. Which class we extends while writing custom exception? (If checked then Exception, if unchecked then RuntimeException)
25. Try, catch, finally concept with throw new exception and return statement.
26. What is Enum? What is the use of Enum?
27. What are IS A (Inheritance) and HAS A (Composition) relationships?
28. Explain Association, Aggregation and Composition.
29. Why is String immutable? (Benefit: caching, security, synchronization, performance)
30. Difference between ClassNotFoundException and NoClassDefError.
31. Why new DateTime API is added in Java8, when we already had one?
32. My class is implementing 2 interfaces with same default methods. How to resolve ambiguity?
33. What is externalization? What is it and how to use Externalizable?
34. Can main method be final?
35. Can the String[] argument in main method be final?
36. Can main method throw Throwable or Error?
37. There are overloaded methods, one with String argument and another with Object argument. If I call the method with null as parameter, then what will happen?

# Java Collection

1. What is generics? Usage in Java.
2. Difference btw LIST/SET/MAP.
3. What is ConcurrentHashMap and its importance?
4. Difference between ConcurrentHashMap and Hashtable?
5. Difference between ConcurrentHashMap and SynchronizedMap
6. About ConcurrencyLevel in ConcurrentHashMap
7. What is ConcurrentModificationException and How to avoid the same?
8. What are fail safe and fail fast?
9. Internal implementation of HashMap.
10. Internal implementation of HashSet.
11. What should we do if we need synchronized Hashmap?
12. Internal implementation of LIST interface implementation classes?
13. ArrayList vs LinkedList. When to use what?
14. If ArrayList doesn’t have sufficient space in memory for new element then what will happen?

# Java Programming Questions

1. We have 2 strings. Find out the lines in which the second string occurs in the first string without using contains.
2. What is the way if you have 1 million of data store inside DB…what all condition will you apply in code for faster execution?
3. If there are four processes, how will you make sure fourth process will start first three processes finish their processing?
4. Suppose you have Singleton class, you have a method getInstance() that returns same instance each time the method is invoked. But, I need multiple instances to be formed and after a certain number of instances created, an error should be thrown.
5. There is series of numbers (1 2 3 4 1 1 5 6 7 ……..) , find the repeated number (the numbers are continuous)
6. You have palindrome String =”abcdcba” , a new character ‘x’ is added , now what is the logic to find the inserted character (return the character inserted and the position at which the character is inserted)
7. In the above question if the string length is odd.
8. Write a program for string palindrome
9. Create a thread using java8
10. Write a program to find the index of first repeating occurrence in a string

Ex: String =”BANGALORE”;

We need to find the index position of first repeating character

1. Array {2 2 1 1 3}, find the first non-repeated number.
2. Given a paragraph, find the number of repeated words.
3. Find the second largest integer in the integer array.
4. How to do validation of java code, say number of braces {}, semicolon ; etc ?
5. Method M1 get operation on MAP and M2 is put operation on MAP. How to synchronize?
6. String Regular expression – String s = https://server:port/path/path1?type=a&type=b get all the values?
7. Write program using lambda expression and filter?
8. Create a queue using 2 stacks.

# Java Multithreading

1. What is Multithreading?
2. How would you create a thread pool?
3. How to add tasks to thread pool?
4. What is the difference between Thread and Process?
5. When to implement Runnable and when to extend Thread while creating a thread?
6. Difference between start() and run() method of Thread class.
7. Difference between Runnable and Callable?
8. What is race condition?
9. How to stop a thread?
10. How will you share data between two threads?
11. Why wait(), notify() and notifyAll() methods are in Object class and not in Thread class?
12. What is ThreadLocal variable?
13. What is synchronization? If we have one synchronization method and I need to call another synchronization from existing one, will it work? If yes, then how? (Tell me a scenario)
14. Can we call multiple start()? (Tell the scenarios)
15. What is deadlock?
16. What is the solution for deadlock?
17. How can create thread using callable interface(indirectly)
18. Difference between CyclicBarrier and CountDownLatch.
19. Difference between join and CountDownLatch.
20. Difference beet wait and sleep.
21. Future concept in java?
22. What is the difference between synchronized function and synchronized block?
23. Method A is synchronized. Method B in same class is static and synchronized. Can two threads simultaneously access both the methods?
24. Where can we use synchronization?
25. How to ensure for thread safety without synchronization? (Hint: ThreadLocal)
26. What is class level lock?
27. How to synchronize an ArrayList?
28. What is join and fork? When to use it?
29. What is Semaphore and BlockingQueue?
30. What is the difference between livelock and deadlock?
31. How do you check if a Thread holds a lock or not? (Thread.holdsLock() – returns true/false)
32. There are three threads T1, T2, and T3? How do you ensure sequence T1, T2, T3 in Java?
33. What does yield method of Thread class do?
34. Thread states.
35. What is daemon thread? How to create a thread daemon and how to check?
36. What is Executor Framework (Executor and ExecutorService interfaces)?

# Design Pattern

1. What are the design patterns?
2. What is Gang of Four?
3. What are Core Java design patterns?
4. What are J2EE design patterns?
5. What is Singleton pattern?
6. What is Factory pattern?
7. What is Façade pattern?
8. What is Decorator pattern?
9. What is Prototype pattern?
10. What is MVC pattern?
11. What is Front Controller pattern?
12. What is Intercepting Filter pattern?
13. What is DAO and TO pattern?
14. What are the design patterns you have used till now?

# Data Structure Programming Questions

1. Reverse a singly linked List through Recursive method
2. Reverse a singly linked List through iterative method
3. Find the frequency of character in below sentence with most appropriate data structure
   1. “India is largest democracy country in world”
4. Sort the given word of sentences on the basis of length
   1. “JPMC is investment banking but it also looking forward to another banking system”
   2. Condition:-
   3. If length is same the need to keep dictionary order.
5. Find the shortest path in 3 dimensional room with 6 corner.
6. Detect the loop in linked list.
7. Explain Quick sort.
8. What is time complexity of Quick sort algorithms?
9. Create a queue using 2 stacks.

# Spring/Spring Boot Questions

1. What is Spring Boot?
2. How will you implement spring batch behavior in single java process?
3. What is Dependency Injection, how do you achieve it
4. Different types of Dependency Injection, how do you decide on what basis the type of injection to be used?
5. Why @Qualifiers are used and also mentioned about @Primary, Explain with an example.
6. Explain about Cyclic Dependency with an Example.( Explain with code like class A , class B)
7. Tell me about @Component, @Controller, @Service, @Repository and importance of the same.
8. If you move the bean from @Repository to @Service and vice versa, will you get any error or the code will work.
9. @PathVariable, @PathParam, @RequestParam, explain with actual example.
10. Design an entity, rest controller, service layer, and persistence layer in spring boot.
11. What is the entry point for restful webservice?
12. @Restconroller use in webservice?
13. You create multiple instances of one bean, if you modify any one instance of bean, will it affect other beans or not?
14. What is dependency injection? How dependency injection is working?
15. How setter injection is working?
16. Without setter injection and constructor injection how to assign values to spring bean?
17. Spring Circular dependency, way to avoid it?
18. If you create singleton spring bean more than one time, how many times spring bean object is created.
19. Spring Circular dependency, way to avoid it?
20. Difference between @Component and @Bean?
21. Bean Lifecycle? How to avoid initialization of beans while start up?
22. Difference btw bean factory and application context?
23. How bean initialization works?
24. What are the various bean scopes?
25. What is Prototype?
26. What is singleton?
27. When to use setter and when to use constructor injection?
28. Difference between Spring singleton and Core java singleton.
29. Use of @Inject.
30. @RestController vs @Controller.
31. Explain Spring Transaction management.
32. What is the default strategy for autowiring? (Answer: byType)

# WebService Questions

1. What is rest web service?
2. What is difference between SOAP and REST
3. What are the advantages of REST over SOAP?
4. Implement a rest API to get the details from employee and update the details.
5. How do you consume a restful web service?
6. Why do we need to use rest template?
7. How did you use micro services in your project?
8. How do you deploy micro services?
9. What is idempotent and non-idempotent in rest?(POST – non-idempotent)
10. Example methods for idempotent and non-idempotent.
11. Difference between put and post requests.
12. Write a program to implement restful web service.
13. What are the advantages of micro services?
14. What are the disadvantages of micro services?
15. Restful webservices -- Design a system, UI sends the request to System A, and System A interacts with System B and renders the response to UI.
16. What will happen if we use GET for insertion, PUT for deletion, etc.?
17. Explain stateless and stateful.
18. What is versioning in REST API?

# Hibernate Questions

1. Can we have two config file in Hibernate, if yes, then how we can use it and why do we use multiple hibernate config files?
2. Difference between get() and load().
3. Hibernate configuration file (regarding properties, xml tags)
4. Hibernate states.
5. Difference between save() and persist(). Which is better and why?
6. Hibernate mapping.
7. Hibernate cache.
8. Creating Session & Transaction.
9. save() vs merge().
10. What is Expression and Restriction.
11. How to pass list of parameters in Spring JPA? (using In, e.g- findByUserIdIn(List<String> userIdList))
12. Load balancer.
13. Lazy vs Eager.
14. Service has @Transactional and DAO also has @Transactional. If service is throwing error, will commits in DAO rollback?
15. How to remove cache in Hibernate?
16. How to handle object if session cache has dirty (old) value in the object.

# Database Questions

1. How many types of indexes are there?
2. How many types of locks are there?
3. How do indexes work?
4. What is ACID?
5. What is transaction management inside database?
6. What are transactions?
7. How do you manage transactions?
8. Difference between Primary Key and Unique Key in database?
9. What is join?
10. What is left outer join?
11. Difference between left join and left outer join.
12. What is composite key?
13. How to create table for many to many relationship?

# SQL

1. Find the 2nd largest salary (Do not use Rownum, DenseRank, Limit)
2. What is the advantage of using joins in sql query?
3. Write an SQL query using joins.
4. Write an SQL query to fetch the last 5 records from a table.
5. Scenario: assume employee and department are two database tables I want to fetch the data from those two tables. Write a query using left join and what will be the output?
6. What is the use of Union and Union all in sql, write a query using union.

# JMS Questions

1. JMS Security

# Others

1. Explain the scenario in your project where you got challenges and what was your approach to overcome.
2. What are the version control tools that you have used in your project?
3. Difference between SVN and Git. (cherrypick, reset, 2 layers of repo, stash)
4. How to change commit message in Git? (git commit --amend)
5. What are the servers that you have used in your last project?
6. Cookie and Session Management.
7. How two users will be identified in the browser?
8. In a website, how to calculate number of hits happened for the URL?
9. What is the use of Mockito?
10. How to mock static methods?
11. How can we bypass a REST call in JUnit testing?

# Angular

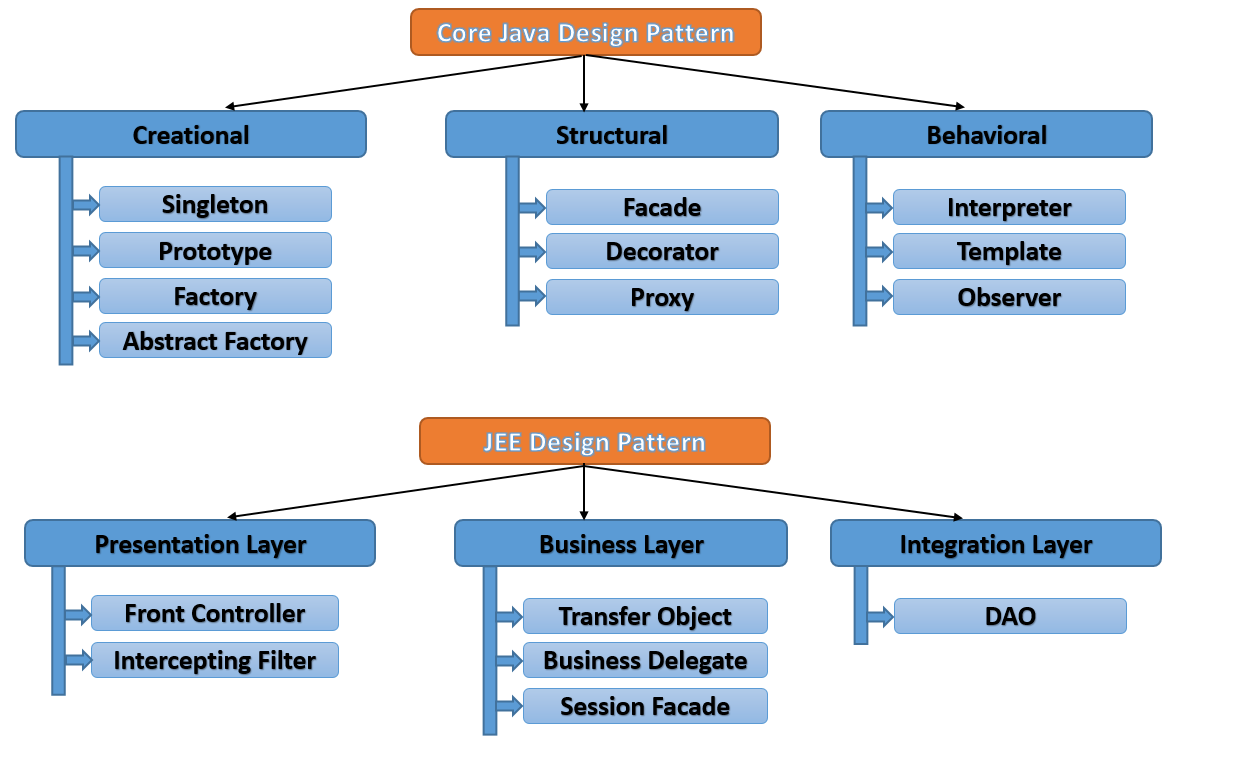
1. Angular work flow.
2. How to use component of one module in another module?
3. Difference between JavaScript and TypeScript.
4. Directives.
5. Data binding.
6. Observable design pattern.
7. Routing and Guard.
8. How to pass value in routing? ({path = ‘api/:id/get’, component: AbcComponent})
9. How to implement Routing between parent and child component?
10. How data is being passed from parent component to child component and vice versa?
11. How to implement file download feature in Angular.

# JavaScript

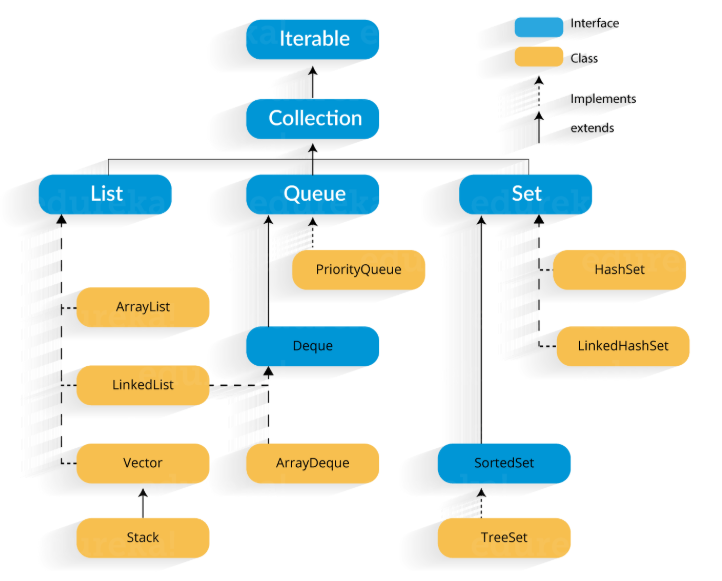
1. How to remove an entry from a list?
2. Difference vs == and ===.
3. Difference between slice() and splice().
4. How to create a popup?
5. Fetch API.

|  |  |  |
| --- | --- | --- |
| Data Structure | Time Complexity | Space Complexity |
| Bubble Sort | O(n2) | O(1) |
| Insertion Sort | O(n2) | O(1) |
| Selection Sort | O(n2) | O(1) |
| Quick Sort | O(n2) [Average: O(n\*log n)] | O(n\*log n) |
| Merge Sort | O(n\*log n) | O(n) |
| Heap Sort | O(n\*log n) | O(1) |
|  |  |  |
| Linear Search | O(n) |  |
| Binary Search | O(log n) |  |
|  |  |  |
| Stack push | O(1) |  |
| Stack pop | O(1) |  |
| Stack search | O(n) |  |
|  |  |  |
| Queue enqueue | O(1) |  |
| Queue dequeue | O(1) |  |
| Queue size | O(1) |  |
|  |  |  |
| Arrays.sort() | O(n\*log n) [Quick Sort for primitives and Merge Sort for objects] |  |
| Collections.sort() | O(n\*log n) [Merge Sort] |  |
|  |  |  |
| ArrayList |  |  |
| add() | O(1) |  |
| add(index, element) | O(n) |  |
| get() | O(1) |  |
| remove() | O(n) |  |
| indexOf() | O(n) |  |
| contains() | O(n) |  |
|  |  |  |
| LinkedList |  |  |
| add() | O(1) |  |
| remove() | O(1) |  |
| get() | O(n) |  |
| contains() | O(n) |  |
|  |  |  |
| HashSet/LinkedHashSet |  |  |
| add() | O(1) |  |
| remove() | O(1) |  |
| contains() | O(1) |  |
|  |  |  |
| TreeSet |  |  |
| add() | O(log n) |  |
| remove() | O(log n) |  |
| contains() | O(log n) |  |
|  |  |  |
| HashMap/LinkedHashMap |  |  |
| put() | O(1) |  |
| get() | O(1) |  |
|  |  |  |
| TreeMap |  |  |
| put() | O(log n) |  |
| get() | O(log n) |  |
| remove() | O(log n) |  |
| containsKey() | O(log n) |  |

# Important design patterns (for interview):



# Java Collection Hierarchy



# Serialization Important Points:

1. Serialization is used to transfer object into byte stream.
2. Serializable is a marker interface.
3. Keyword transient is used for instance variable. A transient variable doesn’t take part in serialization. After deserialization we get the default value for transient variables.
4. A static variable also doesn’t serialize. We get the default value after deserialization.
5. If a final instance variable is initialized during declaration, then after deserialization we get the initialized value.
6. If a final instance variable is initialized inside instance block or constructor, then after deserialization we get default value.
7. Check the case: final transient variable, initialized during declaration.
8. Check the case: final static variable, initialized during declaration.
9. Keyword volatile is related to multithreading, not serialization.
10. We can control serialization using Externalizable interface.

Data Structure Coding question and solutions:

1. **Bubble Sort:**

private static void bubbleSort(int[] array, int n) {

for(int i=0; i<n; i++) {

for(int j=0; j<n-i-1; j++) {

if(array[j] > array[j+1]) {

int temp = array[j];

array[j] = array[j+1];

array[j+1] = temp;

}

}

}

}

1. **Selection Sort:**

for(int i=0; i<arr.length; i++) {

int min = arr[i];

int minIndex = 0;

boolean flag = false;

for(int j=i+1; j<arr.length; j++) {

if(arr[j] < min) {

min = arr[j];

minIndex = j;

flag = true;

}

}

if (flag) {

swap(arr, i, minIndex);

}

}

1. **String Permutation:**

public class StringPermutation {

public static void main(String as[]) {

permutation("ABC");

}

private static void permutation(String word) {

permutation("", word);

}

private static void permutation(String prefix, String word) {

if(word.isEmpty()) {

System.out.println(prefix);

} else {

int n = word.length();

for (int i=0; i<n; i++) {

permutation(prefix + word.charAt(i), word.substring(0, i) + word.substring(i+1, n) );

}

}

}

}

1. **Queue From Two Stacks:**

public class QueueFromTwoStacks {

private Stack<Integer> stack1 = new Stack<>();

private Stack<Integer> stack2 = new Stack<>();

public void enqueue(int data) {

stack1.push(data);

}

public Integer dequeue() {

if (stack2.isEmpty()) {

shift();

}

return stack2.pop();

}

public Integer peek() {

if (stack2.isEmpty()) {

shift();

}

return stack2.peek();

}

private void shift() {

while(!stack1.isEmpty()) {

stack2.push(stack1.pop());

}

}

}

1. **Insert And Reverse Singly LinkedList:**

public class LinkedList {

Node head;

static class Node {

public int data;

public Node next;

public Node() { }

public Node(int data) {

this.data = data;

this.next = null;

}

}

public static LinkedList insertNode(LinkedList list, int data) {

Node newNode = new Node(data);

if (list.head == null) {

list.head = newNode;

return list;

} else {

Node last = list.head;

while (last.next != null) {

last = last.next;

}

last.next = newNode;

}

return list;

}

public static void printLinkedList(LinkedList list) {

Node head = list.head;

while(head != null) {

System.out.print(head.data + " ");

head = head.next;

}

}

public static void main(String as[]) {

LinkedList list = new LinkedList();

list = insertNode(list, 1);

list = insertNode(list, 2);

list = insertNode(list, 3);

list = insertNode(list, 4);

list = insertNode(list, 5);

printLinkedList(list);

System.out.println();

System.out.println("================");

LinkedList reverseList = reverse(list);

printLinkedList(reverseList);

}

private static LinkedList reverse(LinkedList list) {

Node head = list.head;

Node last = null;

while (head != null) {

Node temp = new Node();

temp.data = head.data;

temp.next = last;

last = temp;

head = head.next;

}

list.head = last;

return list;

}

}