Assignment 2

# 21AIE111

Data Structure and Algorithms – SEM-II

Professor – Dr. Sachin Sir

Submitted By: Vikhyat Bansal [CB.EN.U4AIE21076]

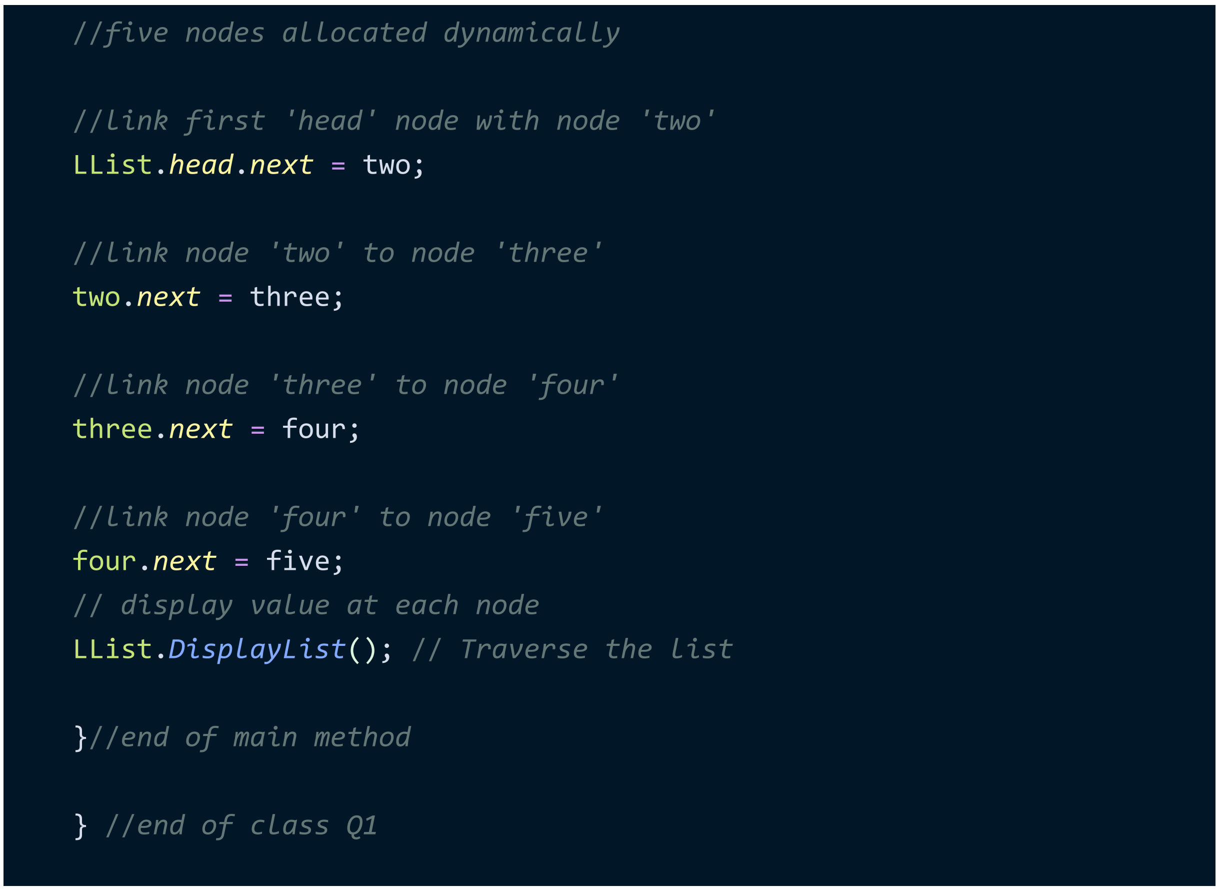
AMRITA VISHWA VIDYAPEETHAM

DEEMED TO BE UNIVERSITY

1. Write a java code to create a singly linked list for the sentence "I am writing java code". Each node will contain the words. Display the value of each nodes.

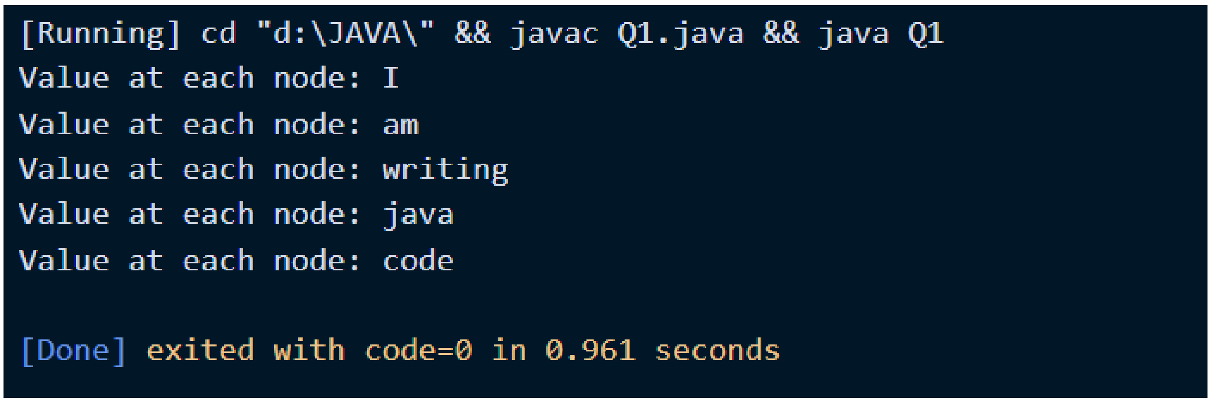
CODE:

```
public class Q1 {

    Node head; // 'head' of the linked list

    // node in the Linked list is a class
    static class Node{
    String data;
    Node next;

    //constructor is used to create a new Node and Next is by default is
initialized as null
    Node(String d){
          data = d;
          next = null;
    } //end of constructor


    } //end of static class node
    public void DisplayList() {
        Node node = head;

        while(node!=null) {
              System.out.println("Value at each node: "+node.data+" ");
              node = node.next;
        }
        }//end of DisplayList function
        //main method
public static void main(String[] args) {

    Q1 LList = new Q1(); // create an empty Linked list

    // create 5 nodes
    LList.head = new Node("I");
    Node two = new Node("am");
    Node three = new Node("writing");
    Node four = new Node("java");
    Node five = new Node("code");
```



```
    //five nodes allocated dynamically

    //link first 'head' node with node 'two'
    LList.head.next = two;

    //link node 'two' to node 'three'
    two.next = three;

    //link node 'three' to node 'four'
    three.next = four;

    //link node 'four' to node 'five'
    four.next = five;
    // display value at each node
    LList.DisplayList(); // Traverse the list

    }//end of main method

    } //end of class Q1
```

OUTPUT:



```
[Running] cd "d:\JAVA\" && javac Q1.java && java Q1
Value at each node: I
Value at each node: am
Value at each node: writing
Value at each node: java
Value at each node: code

[Done] exited with code=0 in 0.961 seconds
```

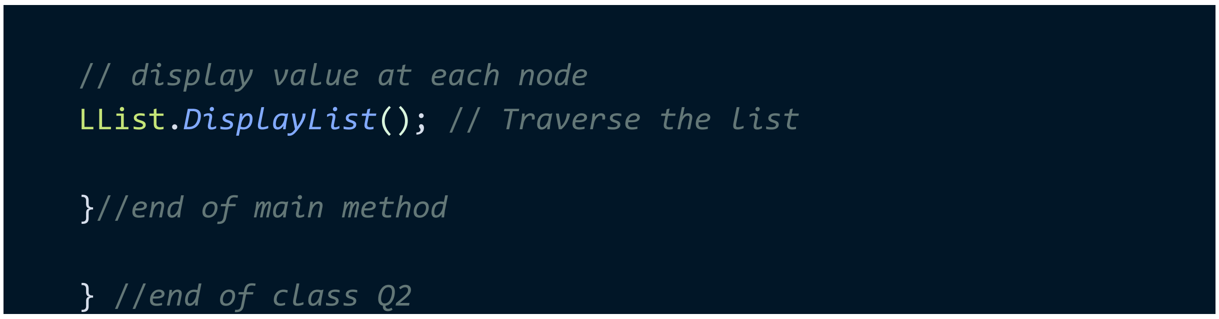
Explanation:

- As it is known that a single linked list contains head, nodes and a tail. We will start by creating a head of linked list, then we will create a class for Node with datatype string.
- Node next will determine the number of nodes being created.
- We will display our linked list using DisplayList() where a while condition is set to see the values of each node and that condition will also check whether empty or not.

- 2. Write a java code to display the value of each node in reverse order (use the linked list created in Q1).

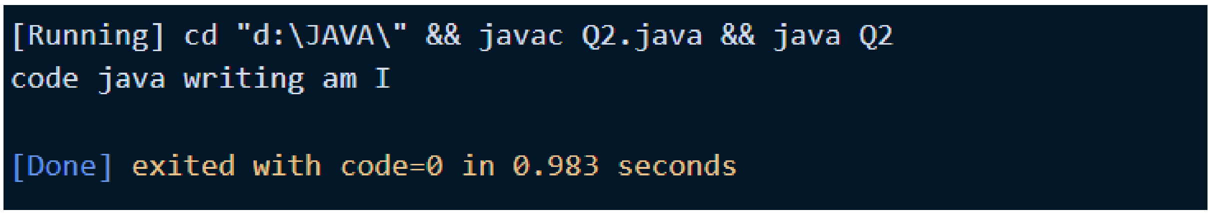
CODE:

```java
public class Q2 {

    Node head; // 'head' of the linked list

    // node in the Linked list is a class
    static class Node{
    String data;
    Node next;

    //constructor is used to create a new Node and Next is by
default is initialized as null
    Node(String string){
          data = string;
          next = null;
    } //end of constructor

    } //end of static class node

    public void DisplayList() {
        Node node = head;



// A tempNode to hold the value in node temporarily.
Node tempNode = new Node("random string");
/*
While node is not null and the data in node is not same as the data
in tempNode, keep looping.
*/
while(node != null && !node.data.equals(tempNode.data)){
// If next node is same as tempNode then print the value in node.
if(node.next == tempNode){
tempNode = node;
System.out.print(tempNode.data+" ");
node = head;
/* If next node is null then clone node to tempNode and also print
the value.
Node is saved to tempNode so in the next iteration the program can
check if it reached the last value
and print the value just before it.
This is possible because tempNode always holds the last value.
*/
} else if(node.next == null){
tempNode = node;
System.out.print(tempNode.data+" ");
node = head;
}
node = node.next;
}
// Finally, if head is not null print the value in it.
if(head != null){
System.out.println(head.data);
}

    }
    //main method
public static void main(String[] args) {

    Q2 LList = new Q2(); // create an empty Linked list

    // create 5 nodes
    LList.head = new Node("I");
    Node two = new Node("am");
    Node three = new Node("writing");
    Node four = new Node("java");
    Node five = new Node("code");
    //five nodes allocated dynamically

    //link first 'head' node with node 'two'
    LList.head.next = two;

    //link node 'two' to node 'three'
    two.next = three;

    //link node 'three' to node 'four'
    three.next = four;

    //link node 'four' to node 'five'
    four.next = five;
```



```
    // display value at each node
    LList.DisplayList(); // Traverse the list

    }//end of main method

    } //end of class Q2
```

OUTPUT:



```
[Running] cd "d:\JAVA\" && javac Q2.java && java Q2
code java writing am I

[Done] exited with code=0 in 0.983 seconds
```

NOTE: I have not use extended class to solve and relate the Q but the code used in solving the Q is similar to what used in previous Q.

Explanation:

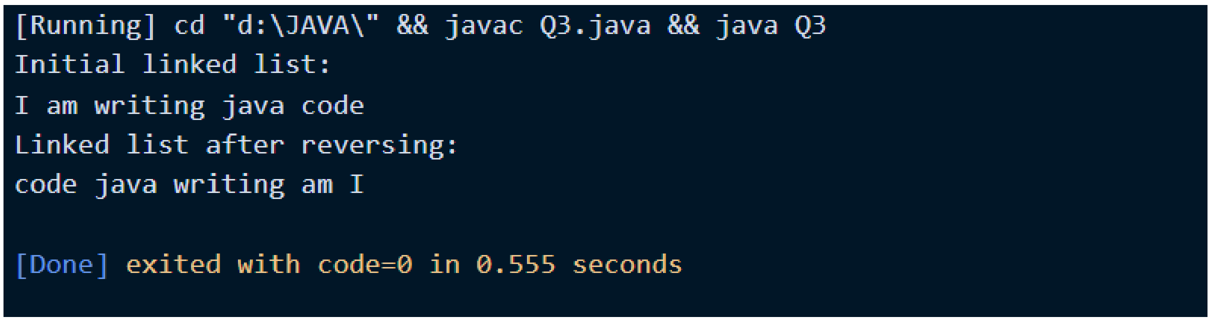
- It was asked to display the value of each node in reverse order(without reversing the Linked List itself) means the linked list will remain same but the values that are being hold by node will reverse.
- We created a tempNode which will store our node data temporarily and after fulfilling all conditions the tempNode will print the values present inside it.

3. Write a java code to reverse the order of singly linked list and display the value in each node (use the linked list used in Q1).

CODE:

```java
public class Q3 {

    static Node head; // head of list

    static class Node {
        String data;
        Node next;
        Node(String string)
        {
            data = string;
            next = null;
        }
    }

    static Node reverse(Node head)
    {
        if (head == null || head.next == null)
            return head;

        /* reverse the rest list and put
        the first element at the end */
        Node rest = reverse(head.next);
        head.next.next = head;

        head.next = null;

        /* head is null now and rest of list is returned */
        return rest;
    }

    /* Function to print linked list */
    static void print()
    {
        Node temp = head;
        while (temp != null) {
            System.out.print(temp.data + " ");
            temp = temp.next;
        }
        System.out.println();
    }


     //end of static class node

        //main method
public static void main(String[] args) {

    Q3 LList = new Q3(); // create an empty Linked list

    // create 5 nodes
    LList.head = new Node("I");
    Node two = new Node("am");
    Node three = new Node("writing");
    Node four = new Node("java");
    Node five = new Node("code");
    //five nodes allocated dynamically

    //link first 'head' node with node 'two'
    LList.head.next = two;

    //link node 'two' to node 'three'
    two.next = three;

    //link node 'three' to node 'four'
    three.next = four;

    //link node 'four' to node 'five'
    four.next = five;
    System.out.println("Initial linked list:");
    print();

    head = reverse(head);

    System.out.println("Linked list after reversing:");
    print();

    }//end of main method

    } //end of class Q3
```

OUTPUT:

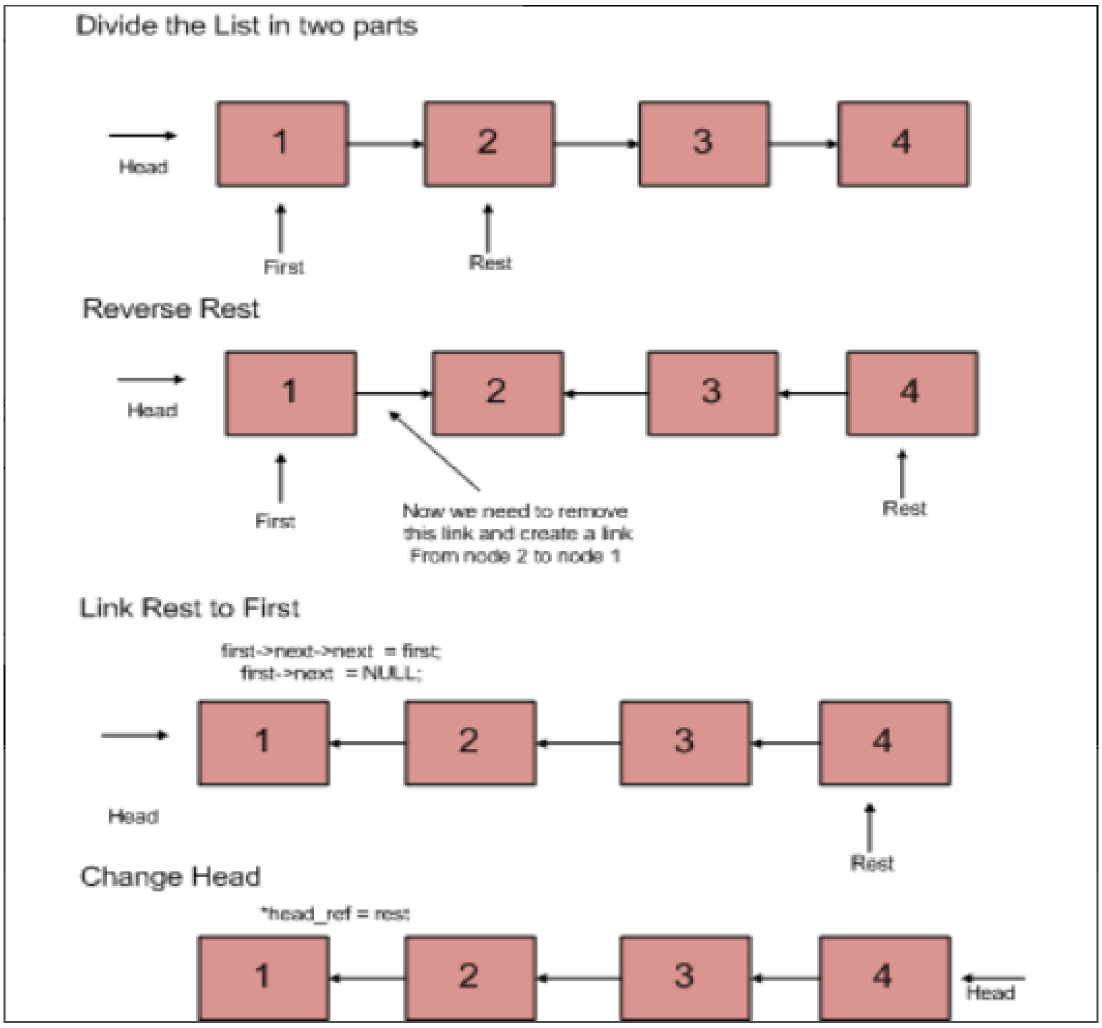


```
[Running] cd "d:\JAVA\" && javac Q3.java && java Q3
Initial linked list:
I am writing java code
Linked list after reversing:
code java writing am I

[Done] exited with code=0 in 0.555 seconds
```

NOTE: I have not use extended class to solve and relate the Q but the code used in solving the Q is similar to what used in previous Q.

Explanation:

- We have divided linked list into two parts i.e., first node and rest of nodes. Here, we will reverse the rest of nodes and link it to the first node.
- Finally, we will fix the head pointer and will display the code.



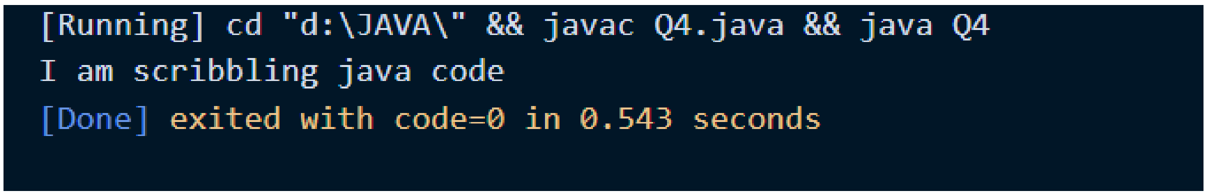
4. Use the linked list in Q1. Write a java code to replace the word 'writing' with 'scribbling' and display the value in each node.

CODE:

```java
public class Q4 {

    Node head; // 'head' of the linked list

    // node in the Linked list is a class
    static class Node{
    String data;
    Node next;

    //constructor is used to create a new Node and Next is by
default is initialized as null
    Node(String string){
          data = string;
          next = null;
    } //end of constructor

    } //end of static class node

    public void DisplayList(){
        // Start from head
        Node node = head;
        /* While loop to traverse through the linked list.
        It will stop execution once the node is null.
        */
        while(node!=null){
        // node.data is used to access the data inside node
        System.out.print(node.data + " ");
        // proceed to next node
        node = node.next;
        }
        }
    // replace writing with scribbling
void replaceWord(String oldWord, String newWord){
    Node node = head;
    /* While loop to iterate through linked list to check if old
word is string in linkedlist. */
    while (node != null){
```

```java
    if(node.data.equals(oldWord)){
    node.data = newWord;
    }
    node = node.next;
    }
}
    //main method
public static void main(String[] args) {

    Q4 LList = new Q4(); // create an empty Linked list

    // create 5 nodes
    LList.head = new Node("I");
    Node two = new Node("am");
    Node three = new Node("writing");
    Node four = new Node("java");
    Node five = new Node("code");
    //five nodes allocated dynamically

    //link first 'head' node with node 'two'
    LList.head.next = two;

    //link node 'two' to node 'three'
    two.next = three;

    //link node 'three' to node 'four'
    three.next = four;

    //link node 'four' to node 'five'
    four.next = five;

    // Call replaceWord method
    LList.replaceWord("writing", "scribbling");

    // display value at each node
    LList.DisplayList(); // Traverse the list

    }//end of main method

    } //end of class Q4
```

OUTPUT:



```
[Running] cd "d:\JAVA\" && javac Q4.java && java Q4
I am scribbling java code
[Done] exited with code=0 in 0.543 seconds
```

NOTE: I have not use extended class to solve and relate the Q but the code used in solving the Q is similar to what used in previous Q's.

Explanation:

- A method named replaceWord is created which will take two strings as input, one string will take old word and other string take the new word.
- A while loop is created to traverse through linked list and an if condition is used to check whether the old word is present in linked list and if this condition follows old word gets replaced with new word.
- In 'Main' class we will use replaceWord method to give old string and new string and will finally display the OUTPUT.

5. Write a java code to check whether the parentheses are balanced in the given expression " [[]{}]((){[]} " . Parentheses are balanced if there is a closing bracket corresponding to an opened one. Implement with Stack.

CODE:

```java
public class Q5 {
    // The stack is created using arrays
    public char [] array;
    public int top;
    public static int length;
    // Constructor to create stack of dimension provided as
argument.
    Q5(int dim){
    array = new char[dim];
    length = dim;
    // Top is maintained, the default value is -1 (no elements)
    top = -1;
    }
    // Method for pushing element to Stack
    public void push(char data){
    // Check if Stack is full. If yes, exit
    if(isFull()){
    System.out.println("Stack Full");
    System.exit(1);
    }
    // Increment the Stack pointer
    top = top+1;
    array[top] = data;
    }
    // Method to pop element from Stack
    public char pop(){
    // Check if stack is empty. If yes, exit.
    if(isEmpty()){
    System.out.println("Stack Empty");
    System.exit(1);
    }
    // Decrement the stack pointer
    top = top - 1;
    // Return the popped value
    return array[top+1];
    }
    // Method to check if the Stack is full
    public boolean isFull(){
    /* If Stack Pointer is = length -1 then the Stack is full
    and this method will return true. */
    return top==(length-1);
    }
    // Method to check if the Stack is full
    public boolean isEmpty(){
    // If Stack Pointer is -1, the Stack is empty.
    return top==-1;
    }
    // Method to print the values in Stack
    public void print(){
    // Same as traversing an array
    for(int i = 0; i < top; i++){
    System.out.println(array[i]);
    }
}
public static boolean balancedParenthesis(String str) {
        Q5 stack = new Q5(str.length());
        for (int i = 0; i < str.length(); i++) {
            char x = str.charAt(i);
            if (x == '(' || x == '[' || x == '{') {
                stack.push(x);
                continue;
            }
            if (stack.isEmpty())
            return false;
// To store the char value that's stored in the stack
            char fromStack;
// Switch case to check
            switch (x) {
/*
The program is checking if the stack contains a right parenthesis.
If the right parenthesis is not the matching one, it will return
false.
If the current char is ')' and the value popped from stack is '{'
When you combine them it will be '{)'
*/
```

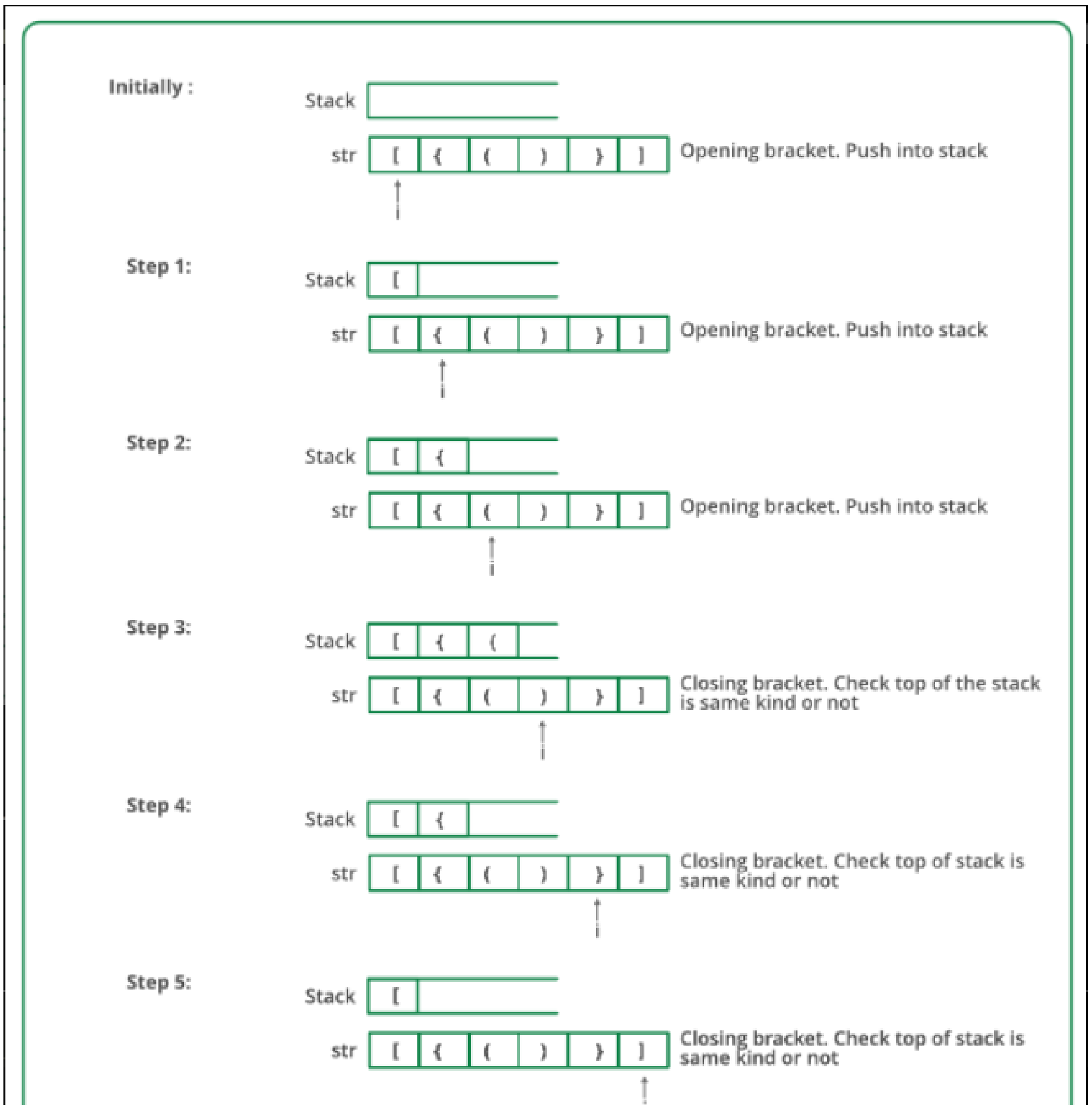
```
             case ')' -> {
             fromStack = stack.pop();
            if (fromStack == '{' || fromStack == '[')
            return false;
}
            case '}' -> {
            fromStack = stack.pop();
            if (fromStack == '(' || fromStack == '[')
            return false;
}
            case ']' -> {
            fromStack = stack.pop();
            if (fromStack == '(' || fromStack == '{')
            return false;
}
}
}
// If the traversal is complete and there's still some string
remaining, then strings are unbalanced

return (stack.isEmpty());
}
    public
    static void main(String[] args) {
        String str = "[[]{}]((){[]}";
        if (balancedParenthesis(str))
            System.out.println("Balanced");
        else
            System.out.println("Unbalanced");
    }
}
```

OUTPUT:

```
[Running] cd "d:\JAVA\" && javac Stack.java && java Stack
Unbalanced


[Done] exited with code=0 in 0.504 seconds
```

Explanation:

- Initially a stack is being created using array where method for push, pop and Boolean isfull() are written.
- Another method is created to check whether parentheses are balanced or not. Initially in the method parentheses are pushed into stack.
- The switch statement is used to perform different actions based on different conditions. In switch statement it is checked whether right parentheses is closed with correct left parentheses. If not, then return false and hence, parentheses are said to be unbalanced.
- Refer diagram below for easy understanding:-



Initially :
Stack
str [ { ( ) } ]
Opening bracket. Push into stack
Step 1:
Stack [
str [ { ( ) } ]
Opening bracket. Push into stack
Step 2:
Stack [ {
str [ { ( ) } ]
Opening bracket. Push into stack
Step 3:
Stack [ { (
str [ { ( ) } ]
Closing bracket. Check top of the stack is same kind or not
Step 4:
Stack [ {
str [ { ( ) } ]
Closing bracket. Check top of stack is same kind or not
Step 5:
Stack [
str [ { ( ) } ]
Closing bracket. Check top of stack is same kind or not

THANK YOU