**Q1**

package Stack;

import java.util.Stack;

public class PalidromeChecker

{

// crating static method for palindrome

public static boolean isPalindrome(String str)

{

Stack<Character> stack = new Stack<>();

int mid = str.length() / 2;

int i;

// inserting element in stack

for (i = 0; i < mid; i++)

{

stack.push(str.charAt(i));

}

if (str.length() % 2 != 0)

{

i++;

}

// checking stack length

while (i < str.length())

{

if (stack.isEmpty() || stack.pop() != str.charAt(i))

{

return false;

}

i++;

}

return true;

}

public static void main(String[] args)

{

String testString = "racecar";

// printing palindrome or not

if (isPalindrome(testString))

{

System.out.println(testString + " is a palindrome.");

}

else

{

System.out.println(testString + " is not a palindrome.");

}

}

}

**Output**

**racecar is a palindrome.**

**Q2**

package Stack;

import java.util.Stack;

public class MinStack

{

private Stack<Integer> stack;

private Stack<Integer> minStack;

// crating stack and min stack object

public MinStack()

{

stack = new Stack<>();

minStack = new Stack<>();

}

// perform insert operation

public void push(int x)

{

stack.push(x);

if (minStack.isEmpty() || x <= getMin())

{

minStack.push(x);

}

}

// removing element from stack

public void pop()

{

if (stack.peek().equals(getMin()))

{

minStack.pop();

}

stack.pop();

}

public int top()

{

return stack.peek();

}

public int getMin() {

return minStack.peek();

}

// check if stack is empty

public boolean isEmpty()

{

return stack.isEmpty();

}

public static void main(String[] args)

{

MinStack minStack = new MinStack();

// Push elements onto the stack

minStack.push(3);

minStack.push(5);

minStack.push(2);

minStack.push(1);

// Display the minimum element

System.out.println("Minimum Element: " + minStack.getMin());

// Pop an element

minStack.pop();

// Display the top element

System.out.println("Top Element: " + minStack.top());

// Display if the stack is empty

System.out.println("Is Stack Empty: " + minStack.isEmpty());

}

}

**Output**

**Minimum Element: 1**

**Top Element: 2**

**Is Stack Empty: false**