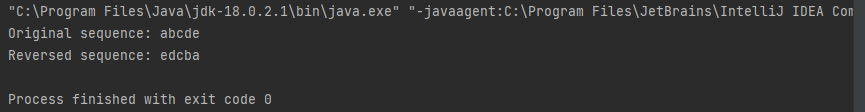
1.

public class Main {  
 public static void main(String[] args) {  
 String string = "abcde";  
 Stack stack = new Stack(string.length());  
   
 for (int i = 0; i < string.length(); i++) {  
 stack.push(string.charAt(i));  
 }  
   
 StringBuilder newString = new StringBuilder();  
 for (int i = 0; i < string.length(); i++) {  
 newString.append(stack.pop());  
 }  
   
 System.*out*.println("Original sequence: " + string);  
 System.*out*.println("Reversed sequence: " + newString);  
 }  
}

class Stack {  
 private int top;  
 private int maxSize;  
 private char[] stackArray;   
  
 Stack(int size) {  
 maxSize = size;  
 stackArray = new char[maxSize];  
 top = -1;  
 }  
  
 public boolean isStackEmpty() {  
 return top == -1;  
 }  
  
 public boolean isStackFull() {  
 return top == maxSize - 1;  
 }  
  
 public void push(char x) {  
 if (isStackFull()) {  
 System.*out*.println("Stack is Full");  
 } else {  
 stackArray[++top] = x;  
 }  
 }  
  
 public char pop() {  
 if (isStackEmpty()) {  
 System.*out*.println("STACK EMPTY");  
 return '\0';   
 }  
 return stackArray[top--];  
 }  
}



2.

package Folder2;  
  
public class Stack {  
 private int top;  
 private int maxSize;  
 private String[] stackArray;  
  
 public Stack(int size){  
 maxSize = size;  
 stackArray = new String[maxSize];  
 top = -1;  
 }  
  
 public boolean isStackEmpty(){  
 return top == -1;  
 }  
  
 public boolean isStackFull(){  
 return top == maxSize - 1;  
 }  
  
 public void push(String word){  
 if (isStackFull()){  
 System.*out*.println("Stack is Full");  
 } else {  
 stackArray[++top] = word;  
 }  
 }  
  
 public String pop(){  
 if (isStackEmpty()){  
 System.*out*.println("Stack is Empty");  
 return null;  
 } else {  
 return stackArray[top--];  
 }  
 }  
}

package Folder2;  
  
import java.util.Scanner;  
  
public class Q2 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 // Input sentence  
 System.*out*.print("Enter a sentence: ");  
 String sentence = scanner.nextLine();  
  
 // Split sentence into words  
 String[] words = sentence.split(" ");  
  
 // Create stack  
 Stack stack = new Stack(words.length);  
  
 // Push all words onto the stack  
 for (String word : words) {  
 stack.push(word);  
 }  
  
 // Pop and print words in reverse order  
 System.*out*.print("Output: ");  
 while (!stack.isStackEmpty()) {  
 System.*out*.print(stack.pop() + " ");  
 }  
  
 System.*out*.println(); // For newline  
 }  
}

A black screen with many small objects

AI-generated content may be incorrect.

3.

package Folder3;  
  
import java.util.Scanner;  
  
public class Q3 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 // Input string  
 System.*out*.print("Enter a string: ");  
 String input = scanner.nextLine().toLowerCase();  
  
 // Remove spaces and non-letter characters (optional)  
 input = input.replaceAll("[^a-z]", "");  
  
 Stack stack = new Stack(input.length());  
  
 // Push each character to the stack  
 for (int i = 0; i < input.length(); i++) {  
 stack.push(input.charAt(i));  
 }  
  
 // Build reversed string  
 StringBuilder reversed = new StringBuilder();  
 while (!stack.isStackEmpty()) {  
 reversed.append(stack.pop());  
 }  
  
 // Compare  
 if (input.equals(reversed.toString())) {  
 System.*out*.println("The string is a palindrome.");  
 } else {  
 System.*out*.println("The string is not a palindrome.");  
 }  
 }  
}

package Folder3;  
  
public class Stack {  
 private int top;  
 private int maxSize;  
 private char[] stackArray;  
  
 public Stack(int size) {  
 maxSize = size;  
 stackArray = new char[maxSize];  
 top = -1;  
 }  
  
 public boolean isStackEmpty() {  
 return top == -1;  
 }  
  
 public boolean isStackFull() {  
 return top == maxSize - 1;  
 }  
  
 public void push(char ch) {  
 if (isStackFull()) {  
 System.*out*.println("Stack is Full");  
 } else {  
 stackArray[++top] = ch;  
 }  
 }  
  
 public char pop() {  
 if (isStackEmpty()) {  
 System.*out*.println("Stack is Empty");  
 return '\0'; // null character  
 } else {  
 return stackArray[top--];  
 }  
 }  
}

A screen shot of a computer

AI-generated content may be incorrect.

4.

package Folder4;  
  
import java.util.Scanner;  
  
public class Q4 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 // Input  
 System.*out*.print("Enter a string: ");  
 String input = scanner.nextLine().toUpperCase();  
  
 // Create two stacks  
 Stack stack1 = new Stack(input.length());  
 Stack stack2 = new Stack(input.length());  
  
 // Collect vowels in stack1  
 for (int i = 0; i < input.length(); i++) {  
 char ch = input.charAt(i);  
 if (*isVowel*(ch)) {  
 stack1.push(ch);  
 }  
 }  
  
 // Sort vowels into stack2 (ascending order)  
 while (!stack1.isEmpty()) {  
 char temp = stack1.pop();  
  
 // Move elements from stack2 back to stack1 until correct spot is found  
 while (!stack2.isEmpty() && stack2.peek() > temp) {  
 stack1.push(stack2.pop());  
 }  
  
 stack2.push(temp);  
 }  
  
 // Output sorted vowels  
 System.*out*.print("Output: ");  
 while (!stack2.isEmpty()) {  
 System.*out*.print(stack2.pop() + " ");  
 }  
 System.*out*.println();  
 }  
  
 // Helper method to check if a character is a vowel  
 public static boolean isVowel(char ch) {  
 return "AEIOU".indexOf(ch) != -1;  
 }  
}

package Folder4;  
  
public class Stack {  
 private int top;  
 private int maxSize;  
 private char[] stackArray;  
  
 public Stack(int size) {  
 maxSize = size;  
 stackArray = new char[maxSize];  
 top = -1;  
 }  
  
 public boolean isEmpty() {  
 return top == -1;  
 }  
  
 public boolean isFull() {  
 return top == maxSize - 1;  
 }  
  
 public void push(char ch) {  
 if (!isFull()) {  
 stackArray[++top] = ch;  
 } else {  
 System.*out*.println("Stack is full.");  
 }  
 }  
  
 public char pop() {  
 if (!isEmpty()) {  
 return stackArray[top--];  
 } else {  
 System.*out*.println("Stack is empty.");  
 return '\0'; // null character  
 }  
 }  
  
 public char peek() {  
 if (!isEmpty()) {  
 return stackArray[top];  
 } else {  
 return '\0';  
 }  
 }  
}

A black screen with many lights

AI-generated content may be incorrect.

5.

package Folder5;  
  
import java.util.Scanner;  
  
public class Q5 {  
 public static void main(String[] args) {  
 Scanner scanner = new Scanner(System.*in*);  
  
 // Input  
 System.*out*.print("Enter an integer: ");  
 int number = scanner.nextInt();  
  
 System.*out*.print("Binary: ");  
 *convertAndPrint*(number, 2);  
  
 System.*out*.print("Octal: ");  
 *convertAndPrint*(number, 8);  
  
 System.*out*.print("Hexadecimal: ");  
 *convertAndPrint*(number, 16);  
 }  
  
 public static void convertAndPrint(int number, int base) {  
 String digits = "0123456789ABCDEF";  
 Stack stack = new Stack(50); // Arbitrary size  
  
 int temp = number;  
 if (temp == 0) {  
 System.*out*.println("0");  
 return;  
 }  
  
 // Conversion logic using remainders  
 while (temp > 0) {  
 int remainder = temp % base;  
 stack.push(digits.charAt(remainder));  
 temp /= base;  
 }  
  
 // Print in correct order by popping from stack  
 while (!stack.isEmpty()) {  
 System.*out*.print(stack.pop());  
 }  
 System.*out*.println();  
 }  
}

package Folder5;  
  
public class Stack {  
 private char[] stackArray;  
 private int top;  
 private int maxSize;  
  
 public Stack(int size) {  
 maxSize = size;  
 stackArray = new char[maxSize];  
 top = -1;  
 }  
  
 public boolean isEmpty() {  
 return top == -1;  
 }  
  
 public boolean isFull() {  
 return top == maxSize - 1;  
 }  
  
 public void push(char ch) {  
 if (!isFull()) {  
 stackArray[++top] = ch;  
 } else {  
 System.*out*.println("Stack is full.");  
 }  
 }  
  
 public char pop() {  
 if (!isEmpty()) {  
 return stackArray[top--];  
 } else {  
 System.*out*.println("Stack is empty.");  
 return '\0';  
 }  
 }  
}

A black screen with many small objects

AI-generated content may be incorrect.

6.

package Folder6;  
  
public class Q6 {  
 public static void main(String[] args) {  
 int[] elements = {3, 12, 9, 27, 15};  
  
 Stack mainStack = new Stack(elements.length);  
 Stack maxStack = new Stack(elements.length);  
  
 int currentMax = Integer.*MIN\_VALUE*;  
  
 System.*out*.println("Tracking max values:");  
 for (int num : elements) {  
 mainStack.push(num);  
 if (num > currentMax) {  
 currentMax = num;  
 }  
 maxStack.push(currentMax);  
 System.*out*.println("Pushed: " + num + ", Current Max: " + currentMax);  
 }  
  
 // Sorting elements in descending order using a temporary stack  
 Stack sortedStack = *sortDescending*(mainStack);  
  
 // Output  
 System.*out*.print("Output (Descending): {");  
 while (!sortedStack.isEmpty()) {  
 System.*out*.print(sortedStack.pop());  
 if (!sortedStack.isEmpty()) System.*out*.print(", ");  
 }  
 System.*out*.println("}");  
 }  
  
 // Sort using stack logic (selection sort style)  
 public static Stack sortDescending(Stack inputStack) {  
 Stack tempStack = new Stack(inputStack.size());  
 Stack sortedStack = new Stack(inputStack.size());  
  
 while (!inputStack.isEmpty()) {  
 int max = Integer.*MIN\_VALUE*;  
  
 // Find the max value in inputStack  
 while (!inputStack.isEmpty()) {  
 int val = inputStack.pop();  
 if (val > max) max = val;  
 tempStack.push(val);  
 }  
  
 boolean maxPushed = false;  
  
 // Push back to inputStack except max value  
 while (!tempStack.isEmpty()) {  
 int val = tempStack.pop();  
 if (val == max && !maxPushed) {  
 sortedStack.push(val);  
 maxPushed = true;  
 } else {  
 inputStack.push(val);  
 }  
 }  
 }  
  
 return sortedStack;  
 }  
}

package Folder6;  
  
public class Stack {  
 private int[] stackArray;  
 private int top;  
 private int maxSize;  
  
 public Stack(int size) {  
 maxSize = size;  
 stackArray = new int[maxSize];  
 top = -1;  
 }  
  
 public boolean isEmpty() {  
 return top == -1;  
 }  
  
 public boolean isFull() {  
 return top == maxSize - 1;  
 }  
  
 public void push(int item) {  
 if (!isFull()) {  
 stackArray[++top] = item;  
 } else {  
 System.*out*.println("Stack is full.");  
 }  
 }  
  
 public int pop() {  
 if (!isEmpty()) {  
 return stackArray[top--];  
 } else {  
 System.*out*.println("Stack is empty.");  
 return -1;  
 }  
 }  
  
 public int peek() {  
 if (!isEmpty()) {  
 return stackArray[top];  
 } else {  
 return -1;  
 }  
 }  
  
 public int size() {  
 return top + 1;  
 }  
  
 public void printStack() {  
 for (int i = top; i >= 0; i--) {  
 System.*out*.print(stackArray[i] + " ");  
 }  
 System.*out*.println();  
 }  
}

A screen shot of a computer

AI-generated content may be incorrect.