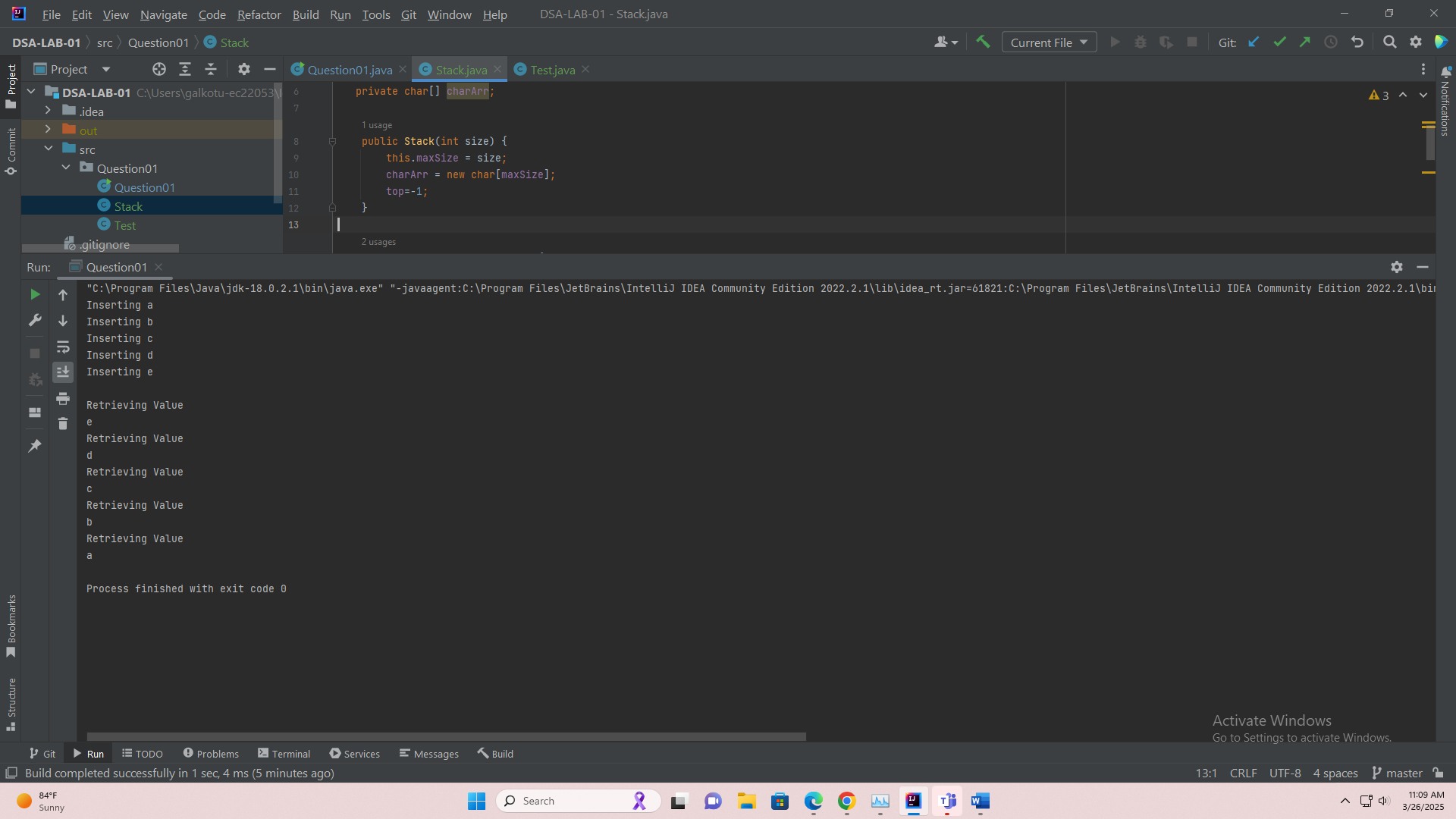
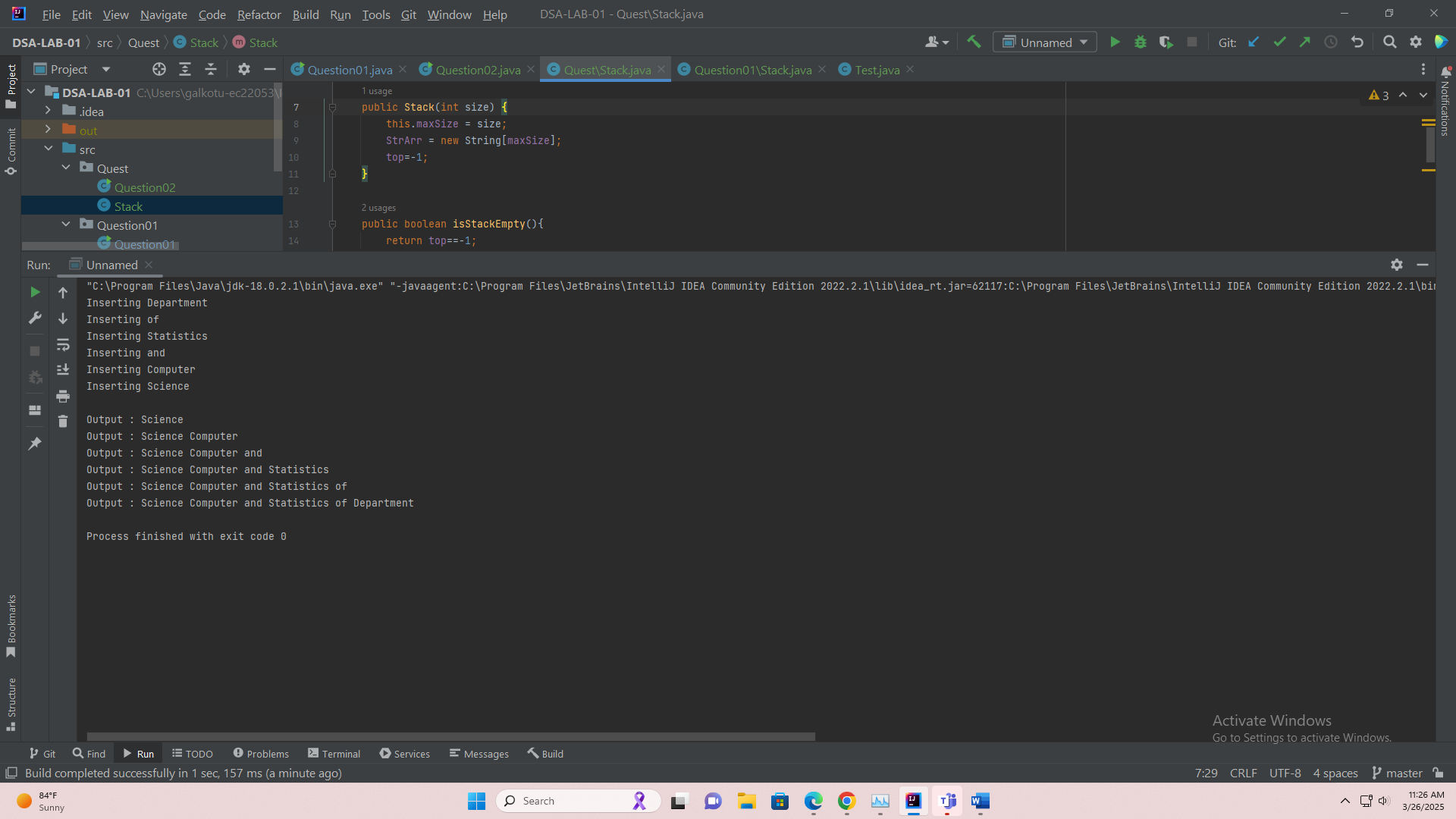
Question 01.

package Question01;  
public class Question01 {  
 public static void main(String[] args) {  
 Stack stackS = new Stack(5);  
 char wordArray[] = {'a','b','c','d','e'};  
 for (char a: wordArray) {  
 stackS.push(a);  
 }  
 System.*out*.println("");  
 while (!stackS.isStackEmpty()){  
 System.*out*.println(stackS.pop());  
 }  
  
 }  
}



package Question01;  
  
public class Stack {  
 private int top;  
 private int maxSize;  
 private char[] charArr;  
  
 public Stack(int size) {  
 this.maxSize = size;  
 charArr = new char[maxSize];  
 top=-1;  
 }  
  
 public boolean isStackEmpty(){  
 return top==-1;  
 }  
  
  
 public boolean isStackFull(){  
 return top==maxSize-1;  
 }  
  
 public int getTop() {  
 return top;  
 }  
  
 public void push(char newValue){  
 if (isStackFull()){  
 System.*out*.println("Stack Already Full");  
 }  
 else {  
 System.*out*.println("Inserting "+newValue);  
 charArr[++top] = newValue;  
 }  
 }  
  
 public char pop(){  
 if (isStackEmpty()){  
 System.*out*.println("Stack is Already Empty");  
 return charArr[top];  
 }  
 else {  
 System.*out*.println("Retrieving Value");  
 return charArr[top--];  
 }  
 }  
  
}

Question 02.

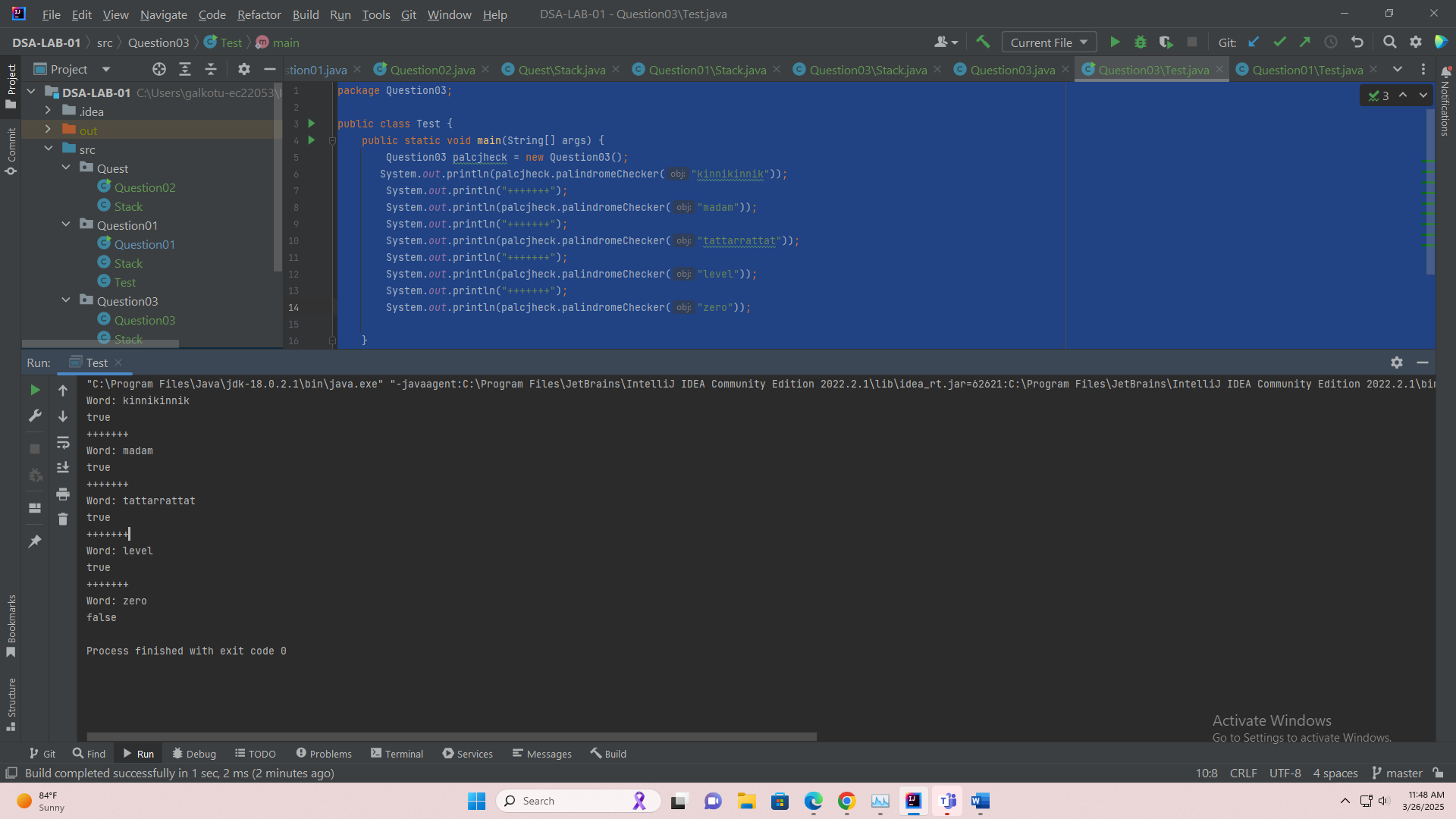
package Question02;  
  
public class Question02 {  
 public static void main(String[] args) {  
 Stack stackS = new Stack(6);  
 String wordArray[] = {"Department", "of", "Statistics", "and", "Computer", "Science"};  
 for (String a : wordArray) {  
 stackS.push(a);  
 }  
 System.*out*.println("");  
 String fOut = "";  
 while (!stackS.isStackEmpty()) {  
 fOut += stackS.pop();  
 fOut += " ";  
 System.*out*.println("Output : " + fOut);  
 }  
  
 }  
}

package Question02;  
public class Stack {  
 private int top;  
 private int maxSize;  
 private String[] StrArr;  
  
 public Stack(int size) {  
 this.maxSize = size;  
 StrArr = new String[maxSize];  
 top=-1;  
 }  
  
 public boolean isStackEmpty(){  
 return top==-1;  
 }  
  
  
 public boolean isStackFull(){  
 return top==maxSize-1;  
 }  
  
 public int getTop() {  
 return top;  
 }  
  
 public void push(String newValue){  
 if (isStackFull()){  
 System.*out*.println("Quest.Question02.Stack Already Full");  
 }  
 else {  
 System.*out*.println("Inserting "+newValue);  
 StrArr[++top] = newValue;  
 }  
 }  
  
 public String pop(){  
 if (isStackEmpty()){  
 System.*out*.println("Quest.Question02.Stack is Already Empty");  
 return StrArr[top];  
 }  
 else {  
 return StrArr[top--];  
 }  
 }  
}

Question 03.

package Question03;  
  
public class Question03 {  
  
 public boolean IsPalindromeOrNot;  
// Write a program to check whether a given string is palindrome using Stack Operations.  
// Example: madam, mom, rotator (Any word that reads the same forward or backward)  
 public boolean palindromeChecker(String obj) {  
 char[] c = obj.toCharArray();  
 Stack stackS = new Stack(c.length);  
 for (char a:obj.toCharArray()){  
 stackS.push(a);  
 }  
 System.*out*.println("Word: "+obj);  
 while (!stackS.isStackEmpty()) {  
 for (char a : obj.toCharArray()) {  
 if (a != stackS.pop()) {  
 IsPalindromeOrNot = false;  
 } else {  
 IsPalindromeOrNot = true;  
 }  
  
 }  
 }  
  
 return IsPalindromeOrNot;  
 }  
  
}

package Question03;  
public class Stack {  
 private int top;  
 private int maxSize;  
 private char[] charArr;  
  
 public Stack(int size) {  
 this.maxSize = size;  
 charArr = new char[maxSize];  
 top=-1;  
 }  
  
 public boolean isStackEmpty(){  
 return top==-1;  
 }  
  
  
 public boolean isStackFull(){  
 return top==maxSize-1;  
 }  
  
 public int getTop() {  
 return top;  
 }  
  
 public void push(char newValue){  
 if (isStackFull()){  
 System.*out*.println("Stack Already Full");  
 }  
 else {  
 // System.out.println("Inserting "+newValue);  
 charArr[++top] = newValue;  
 }  
 }  
  
 public char pop(){  
 if (isStackEmpty()){  
 System.*out*.println("Stack is Already Empty");  
 return charArr[top];  
 }  
 else {  
 // System.out.println("Retrieving Value");  
 return charArr[top--];  
 }  
 }  
  
}

package Question03;  
  
public class Test {  
 public static void main(String[] args) {  
 Question03 palcjheck = new Question03();  
 System.*out*.println(palcjheck.palindromeChecker("kinnikinnik"));  
 System.*out*.println("+++++++");  
 System.*out*.println(palcjheck.palindromeChecker("madam"));  
 System.*out*.println("+++++++");   
 System.*out*.println(palcjheck.palindromeChecker("tattarrattat"));  
 System.*out*.println("+++++++");  
 System.*out*.println(palcjheck.palindromeChecker("level"));  
 System.*out*.println("+++++++");   
 System.*out*.println(palcjheck.palindromeChecker("zero"));  
  
 }  
}

Question 04.

package Question04;  
public class Stack {  
 private int top;  
 private int maxSize;  
 private char[] charArr;  
  
 public Stack(int size) {  
 this.maxSize = size;  
 charArr = new char[maxSize];  
 top = -1;  
 }  
  
 public boolean isStackEmpty() {  
 return top == -1;  
 }  
  
 public boolean isStackFull() {  
 return top == maxSize - 1;  
 }  
  
 public int getTop() {  
 return top;  
 }  
  
 public void push(char newValue) {  
 if (isStackFull()) {  
 System.*out*.println("Stack Already Full");  
 } else {  
 // System.out.println("Inserting "+newValue);  
 charArr[++top] = newValue;  
 }  
 }  
 @Override  
 public String toString() {  
 System.*out*.println(charArr);  
 return charArr.toString();  
 }  
 public char pop() {  
 if (isStackEmpty()) {  
 System.*out*.println("Stack is Already Empty");  
 return charArr[top];  
 } else {  
 // System.out.println("Retrieving Value");  
 return charArr[top--];  
 }}  
}

package Question04;

import java.util.ArrayList;

import java.util.Collection;

public class Question04 {

int aCount=0;

int eCount=0;

int iCount=0;

int oCount=0;

int uCount=0;

public void VowelChecker(String inpString){

Stack stackForWord = new Stack(inpString.length());

Stack stackForVowel = new Stack(inpString.length());

char[] inpStringArray = inpString.toCharArray();

for (char c : inpStringArray) {

stackForWord.push(c);

}

ArrayList<Character> charArrayList = new ArrayList<>();

for (char c : inpStringArray) {

if (c=='a' && aCount == 0) {

charArrayList.add(c);

aCount++;

} else if(c=='e'&& eCount==0){

charArrayList.add(c);

eCount++;

} else if(c=='i'&& iCount==0){

charArrayList.add(c);

iCount++;

} else if(c=='o'&& oCount==0){

charArrayList.add(c);

oCount++;

} else if(c=='u'&& uCount==0){

charArrayList.add(c);

uCount++;

}

}

java.util.Collections.sort(charArrayList);

for (Character character : charArrayList.reversed()) {

stackForVowel.push(character);

}

while (!stackForVowel.isStackEmpty()) {

System.out.printf(stackForVowel.pop()+"");

}

System.out.println();

}

}

package Question04;

public class Test {

public static void main(String[] args) {

Question04 example01 = new Question04();

Question04 example02 = new Question04();

Question04 example03 = new Question04();

Question04 example04 = new Question04();

Question04 example05 = new Question04();

Question04 example06 = new Question04();

Question04 example07 = new Question04();

Question04 example08 = new Question04();

Question04 example09 = new Question04();

Question04 example10 = new Question04();

Question04 example11 = new Question04();

Question04 example12 = new Question04();

example01.VowelChecker("apple"); // Example 1

example02.VowelChecker("zebra"); // Example 2

example03.VowelChecker("school"); // Example 3

example04.VowelChecker("banana"); // Example 4

example05.VowelChecker("library"); // Example 5

example06.VowelChecker("elephant"); // Example 6

example07.VowelChecker("orange"); // Example 7

example08.VowelChecker("unicorn"); // Example 8

example09.VowelChecker("grape"); // Example 9

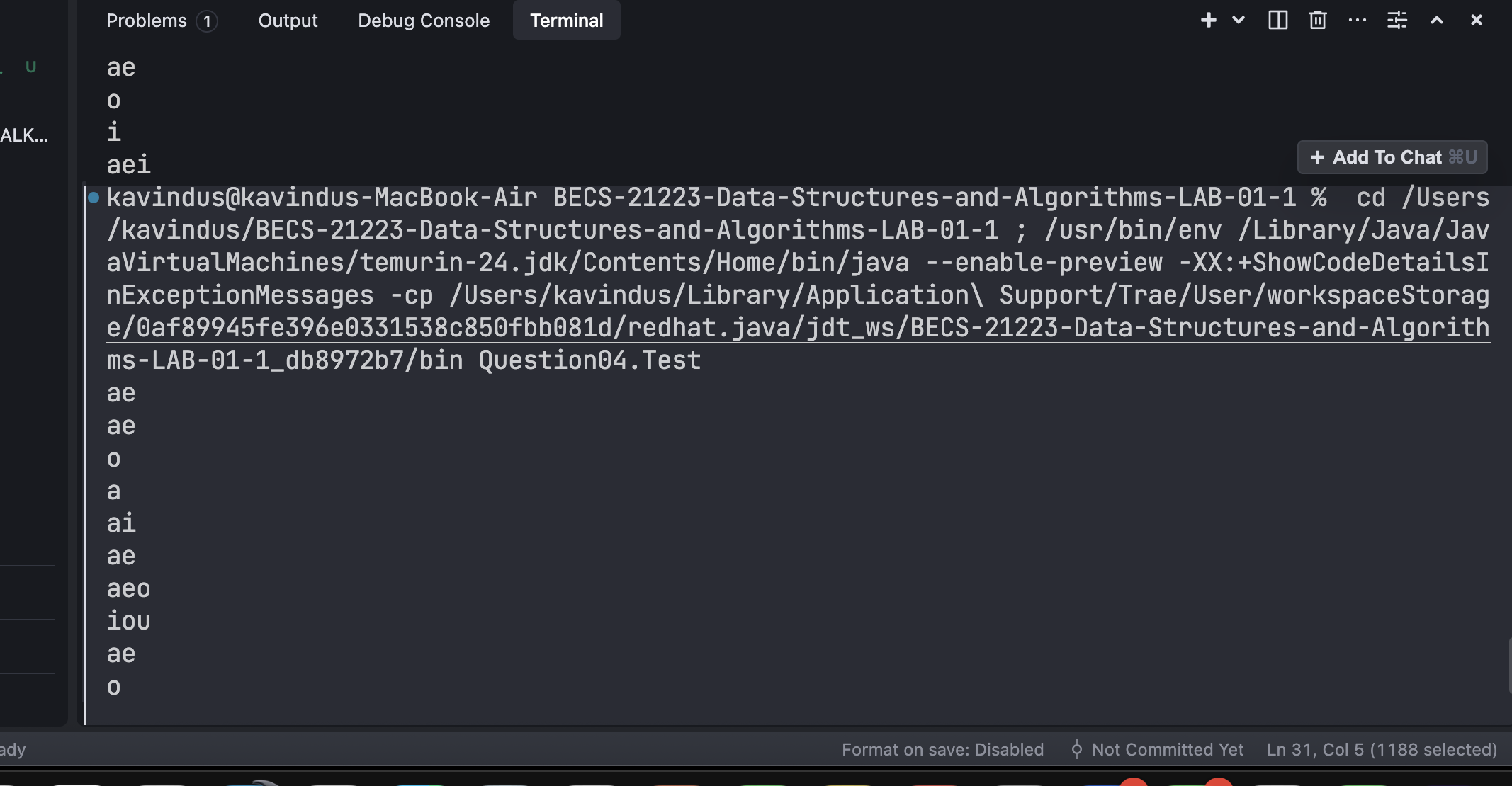
example10.VowelChecker("python"); // Example 10

example11.VowelChecker("kiwi"); // Example 11

example12.VowelChecker("jasmine"); // Example 12

}

}



Question 05.

package Question05;  
public class Stack {  
 private int top;  
 private int maxSize;  
 private String[] StrArr;  
  
 public Stack(int size) {  
 this.maxSize = size;  
 StrArr = new String[maxSize];  
 top=-1;  
 }  
  
 public boolean isStackEmpty(){  
 return top==-1;  
 }  
  
  
 public boolean isStackFull(){  
 return top==maxSize-1;  
 }  
  
 public int getTop() {  
 return top;  
 }  
  
 public void push(int newValue){  
 if (isStackFull()){  
 System.*out*.println("Quest.Question02.Stack Already Full");  
 }  
 else {  
 StrArr[++top] = String.*valueOf*(newValue);  
 }  
 }  
  
 public String pop(){  
 if (isStackEmpty()){  
 System.*out*.println("Quest.Question02.Stack is Already Empty");  
 return StrArr[top];  
 }  
 else {  
 // System.out.println("Retrieving Value");  
 return StrArr[top--];  
 }  
  
 }  
}

package Question05;  
public class StackString {  
 private int top;  
 private int maxSize;  
 private String[] StrArr;  
  
 public StackString(int size) {  
 this.maxSize = size;  
 StrArr = new String[maxSize];  
 top=-1;  
 }  
  
 public boolean isStackEmpty(){  
 return top==-1;  
 }  
  
  
 public boolean isStackFull(){  
 return top==maxSize-1;  
 }  
  
 public int getTop() {  
 return top;  
 }  
  
 public void push(String newValue){  
 if (isStackFull()){  
 System.*out*.println("Quest.Question02.Stack Already Full");  
 }  
 else {  
 StrArr[++top] = String.*valueOf*(newValue);  
 }  
 }  
  
 public String pop(){  
 if (isStackEmpty()){  
 System.*out*.println("Quest.Question02.Stack is Already Empty");  
 return StrArr[top];  
 }  
 else {  
 // System.out.println("Retrieving Value");  
 return StrArr[top--];  
 }  
  
 }  
  
  
}

package Question05;  
public class Question05 {  
 Stack stackforBin = new Stack(32);  
 Stack stackforOctal = new Stack(18);  
 StackString stack = new StackString(12);  
  
 public void binaryConvert(int no){  
 while(!(no==0)){  
 stackforBin.push((char) (no%2));  
 no /=2;  
 }  
 while (!stackforBin.isStackFull()){  
 stackforBin.push(0);  
 }  
 while (!stackforBin.isStackEmpty()){  
 System.*out*.printf(stackforBin.pop()+"");  
 }  
 System.*out*.println();  
 }  
  
 public void octalConvert(int no){  
 while(!(no==0)){  
 stackforOctal.push((char) (no%8));  
 no /=8;  
 }  
 while (!stackforOctal.isStackFull()){  
 stackforOctal.push(0);  
 }  
 while (!stackforOctal.isStackEmpty()){  
 System.*out*.printf(stackforOctal.pop()+"");  
 }  
 System.*out*.println();  
  
 }  
  
 public void hexaConvert(int no) {  
 while (!(no == 0)) {  
 if (no % 16 == 10) {  
 stack.push("A");  
 no /= 16;  
 } else if (no % 16 == 11) {  
 stack.push("B");  
 no /= 16;  
 } else if (no % 16 == 12) {  
 stack.push("C");  
 no /= 16;  
 } else if (no % 16 == 13) {  
 stack.push("D");  
 no /= 16;  
 } else if (no % 16 == 14) {  
 stack.push("E");  
 no /= 16;  
 } else if (no % 16 == 15) {  
 stack.push("F");  
 no /= 16;  
 } else {  
 stack.push(String.*valueOf*(no % 16));  
 no /= 16;  
 }  
 }  
 while (!stack.isStackFull()) {  
 stack.push(String.*valueOf*(0));  
 }  
 while (!stack.isStackEmpty()) {  
 System.*out*.printf(stack.pop() + "");  
 }  
 System.*out*.println();  
  
  
 }  
}

package Question05;  
  
public class Test {  
 public static void main(String[] args) {  
 Question05 question05 = new Question05();  
 question05.binaryConvert(77);  
 question05.octalConvert(77);  
 question05.hexaConvert(77);  
  
 }  
  
}

