**1. HalfPyramidStars:**

**public** **class** RHalfPyramidStars {

**public** **static** **void** main(String[] args) {

**int** i, j, rows;

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print("Enter number of rows: ");

rows = scanner.nextInt();

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \* \* \*

**for** (i = 1; i <= rows; ++i) {

**for** (j = 1; j <= i; ++j) {

System.***out***.print("\* ");

}

System.***out***.println();

}

scanner.close();

}

}

2. **HalfPyramidNumbers**:

**public** **class** SHalfPyramidNumbers {

**public** **static** **void** main(String[] args) {

**int** i, j, rows;

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print("Enter number of rows: ");

rows = scanner.nextInt();

**for** (i = 1; i <= rows; ++i) {

1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

**for** (j = 1; j <= i; ++j) {

System.***out***.print(j + " ");

}

System.***out***.println();

}

scanner.close();

}

}

3. **HalfPyramidAlphabets:**

**public** **class** THalfPyramidAlphabets {

**public** **static** **void** main(String[] args) {

**int** i, j;

**char** input, alphabet = 'A';

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print("Enter an uppercase character you want to print in the last row: ");

input = scanner.next().charAt(0);

**for** (i = 1; i <= (input - 'A' + 1); ++i) {

A

B B

C C C

D D D D

E E E E E

**for** (j = 1; j <= i; ++j) {

System.***out***.print(alphabet + " ");

}

++alphabet;

System.***out***.println();

}

scanner.close();

}

}

4. **InvertedHalfPyramidStars**:

**public** **class** UInvertedHalfPyramidStars {

**public** **static** **void** main(String[] args) {

**int** i, j, rows;

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print("Enter the number of rows: ");

rows = scanner.nextInt();

\* \* \* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

**for** (i = rows; i >= 1; --i) {

**for** (j = 1; j <= i; ++j) {

System.***out***.print("\* ");

}

System.***out***.println();

}

scanner.close();

}

}

5. **InvertedHalfPyramidNumbers**:

**public** **class** VInvertedHalfPyramidNumbers {

**public** **static** **void** main(String[] args) {

**int** i, j, rows;

Scanner scanner = **new** Scanner(System.***in***);

System.***out***.print("Enter the number of rows: ");

rows = scanner.nextInt();

**for** (i = rows; i >= 1; --i) {

1 2 3 4 5

1 2 3 4

1 2 3

1 2

1

**for** (j = 1; j <= i; ++j) {

System.***out***.print(j + " ");

}

System.***out***.println();

}

scanner.close();

}

}

6. **FullPyramidStars**:

\*

\* \* \*

\* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \* \* \* \* \*

**public** **class** WFullPyramidStars {

**public** **static** **void** main(String[] args) {

**int** rows;

System.***out***.print("Enter the number of rows: ");

rows = ***STDIN\_SCANNER***.nextInt();

**for**(**int** i = 1; i <= rows; ++i) {

**for**(**int** space = 1; space <= rows - i; ++space) {

System.***out***.print(" ");

}

**for**(**int** k =1;k <= 2 \* i - 1;k++) {

System.***out***.print("\* ");

}

System.***out***.println();

}

}

**public** **final** **static** Scanner ***STDIN\_SCANNER*** = **new** Scanner(System.***in***);

}

7. **FullPyramidNumbers**:

**public** **class** XFullPyramidNumbers {

**public** **static** **void** main(String[] args) {

1

2 3 2

3 4 5 4 3

4 5 6 7 6 5 4

5 6 7 8 9 8 7 6 5

**int** rows=0;

System.***out***.print("Enter the number of rows: ");

rows = ***STDIN\_SCANNER***.nextInt();

**for** (**int** i = 1; i <= rows; ++i) {

**int** temp=i;

**for** (**int** space = 1; space <= rows - i; ++space) {

System.***out***.print(" ");

}

**for** (**int** k = 1; k <= 2 \* i - 1; k++) {

**if** (k <= i) {

System.***out***.print(temp++ + " ");

} **else** {

System.***out***.print((--temp-1) + " ");

}

}

System.***out***.println();

}

}

**public** **final** **static** Scanner ***STDIN\_SCANNER*** = **new** Scanner(System.***in***);

}

\* \* \* \* \* \* \* \* \*

\* \* \* \* \* \* \*

\* \* \* \* \*

\* \* \*

\*

8. **InvertedFullPyramidStars**:

**public** **class** YInvertedFullPyramidStars {

**public** **static** **void** main(String[] args) {

**int** rows;

System.***out***.print("Enter the number of rows: ");

rows = ***STDIN\_SCANNER***.nextInt();

**for**(**int** i = rows; i >= 1; --i) {

**for**(**int** space = 1; space <= rows - (i-1); ++space) {

System.***out***.print(" ");

}

**for**(**int** j = 1; j <= 2 \* i - 1; ++j) {

System.***out***.print("\* ");

}

System.***out***.println();

}

}

**public** **final** **static** Scanner ***STDIN\_SCANNER*** = **new** Scanner(System.***in***);

}

9. **DiamondPattern**:

**public** **class** ZDiamondPattern {

**public** **static** **void** main(String args[]) {

**int** n, i, j, space = 1;

System.***out***.print("Enter the number of rows: ");

Scanner s = **new** Scanner(System.***in***);

n = s.nextInt();

space = n - 1;

**for** (j = 1; j <= n; j++) {

**for** (i = 1; i <= space; i++) {

\*

\*\*\*

\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*

\*\*\*\*\*

\*\*\*

\*

System.***out***.print(" ");

}

space--;

**for** (i = 1; i <= 2 \* j - 1; i++) {

System.***out***.print("\*");

}

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

}

space = 1;

**for** (j = 1; j <= n - 1; j++) {

**for** (i = 1; i <= space; i++) {

System.***out***.print(" ");

}

space++;

**for** (i = 1; i <= 2 \* (n - j) - 1; i++) {

System.***out***.print("\*");

}

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

}

}

}

10. **FloydTriangle:**

**public** **class** ZFloydTriangle {

**public** **static** **void** main(String[] args) {

1

2 3

4 5 6

7 8 9 10

11 12 13 14 15

**int** rows, number = 1;

System.***out***.print("Enter the number of rows: ");

rows = ***STDIN\_SCANNER***.nextInt();

**for**(**int** i = 1; i <= rows; i++) {

**for**(**int** j = 1; j <= i; ++j) {

System.***out***.print(number + " ");

++number;

}

System.***out***.println();

}

}

**public** **final** **static** Scanner ***STDIN\_SCANNER*** = **new** Scanner(System.***in***);

}

11. **PascalTriangle**:

**public** **class** ZPascalTriangle {

**public** **static** **void** main(String[] args) {

**int** rows, coef = 1;

System.***out***.print("Enter the number of rows: ");

rows = ***STDIN\_SCANNER***.nextInt();

**for**(**int** i = 0; i < rows; i++) {

**for**(**int** space = 1; space <= rows - i; space++) {

System.***out***.print(" ");

}

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

**for**(**int** j = 0; j <= i; j++) {

**if**(j == 0 || i == 0) {

coef = 1;

} **else** {

coef = coef \* (i - j + 1) / j;

}

System.***out***.printf("%4d", coef);

}

System.***out***.println();

}

}

**public** **final** **static** Scanner ***STDIN\_SCANNER*** = **new** Scanner(System.***in***);

}

1.**EqualsString**:

public class NEqualsString {

static void equals(String input1, String input2) {

boolean isEqual = true;

if(input1.length() != input2.length()) {

System.out.println(input1 + " & " + input2 + " are not equal");

return;

}

for (int i = 0; i <= input1.length() - 1 && i <= input2.length() - 1; i++) {

if (input1.charAt(i) != input2.charAt(i)) {

isEqual = false;

break;

}

}

*if (isEqual) {*

*System.out.println(input1 + " & " + input2 + " are equal");*

*} else {*

*System.out.println(input1 + " & " + input2 + " are not equal");*

*}*

*}*

*public static void main(String[] args) {*

*Scanner sc = new Scanner(System.in);*

*System.out.print("Enter your String1: ");*

*String str1 = sc.nextLine();*

*System.out.print("Enter your String2: ");*

*String str2 = sc.nextLine();*

*equals(str1, str2);*

*} }*

1. **ReverseString:**

**import** java.util.\*;

**class** OReverseString {

**public** **static** **void** main(String args[]) {

*stringReverseNewString*();

*stringReverseConcatNewString*();

*stringReverseStringBuilder*();

*stringReverseCharArray*();

}

**static** **void** stringReverseCharArray() {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter string to reverse");

String input = sc.nextLine();

**char**[] charArray = input.toCharArray();

**for** (**int** i = 0, j = charArray.length - 1; i < j; i++, j--) {

**char** temp = charArray[i];

charArray[i] = charArray[j];

charArray[j] = temp;

}

//String string = new String(charArray);

String stringRev = String.*valueOf*(charArray);

System.***out***.println(stringRev);

sc.close();

}

**static** **void** stringReverseStringBuilder() {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter string to reverse");

String str = sc.nextLine();

StringBuilder s = **new** StringBuilder(str);

**for** (**int** i = 0, j = s.length() - 1; i < j; i++, j--) {

**char** temp = s.charAt(j);

s.setCharAt(j, s.charAt(i));

s.setCharAt(i, temp);

}

System.***out***.println(s);

sc.close();

}

**private** **static** **void** stringReverseNewString() {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter string to reverse:");

String original = sc.nextLine();

String reverse = "";

**int** length = original.length();

**for** (**int** i = length - 1; i >= 0; i--) {

reverse = reverse + original.charAt(i);

}

System.***out***.println("Reverse of entered string is: " + reverse);

}

**static** **void** stringReverseConcatNewString() {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.println("Enter string to reverse::");

String s = sc.nextLine();

String reverse = "";

**for** (**int** j = s.length() - 1; j >= 0; j--) {

reverse = reverse.concat(s.charAt(j) + "");

}

System.***out***.println("rev string is" +reverse);

}}

3. **PalindromeString**:

**public** **class** PPalindromeString {

**public** **static** **void** main(String[] args) {

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter your Statement: ");

String input = sc.nextLine();

*checkPalindrome*(input);

}

**static** **void** checkPalindrome(String input) {

**boolean** isPalindrom = **true**;

**for** (**int** i = 0, j = input.length() - 1; i < j; i++, j--) {

**if** (input.charAt(i) != input.charAt(j)) {

isPalindrom = **false**;

**break**;

}

}

**if**(isPalindrom) {

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

} **else** {

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

}

}}

4. **RepeatElementCount**:

public class QRepeatElementCount {

public static int NO\_OF\_CHARS = 256;

public static void main(String[] args) {

String s = "aaabbbbbccd";

charCount(s);

charCountUsingMap(s);

}

static void charCount(String s) {

int count[] = new int[NO\_OF\_CHARS];

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

char element = s.charAt(i);

count[element] = count[element] + 1;

}

System.out.println("Repeated counts are : ");

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

char element = s.charAt(i);

// if (count[element] > 1) for duplicate chars in a string

if (count[element] > 0) {

System.out.printf("%c = %d", element, count[element]);

System.out.println();

count[element] = 0;

}

}

System.out.println();

}

static void charCountUsingMap(String s) {

Map<Character, Integer> m = new HashMap<>();

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

char charAt = s.charAt(i);

if (m.get(charAt) == null) {

m.put(charAt, 1);

} else {

m.put(charAt, m.get(charAt) + 1);

}

}

System.out.println("Repeated counts are : ");

Set<Entry<Character, Integer>> entrySet = m.entrySet();

for (Entry<Character, Integer> e : entrySet) {

System.out.println(e.getKey() + " = " + e.getValue());

}

// Set<Character> keySet = m.keySet();

// for (Character c : keySet) {

// System.out.println(c + " = " + m.get(c));

// }

// } }

5. **AnagramString**:

**public** **class** RAnagramString {

**public** **static** **int** *NO\_OF\_CHARS* = 256;

**public** **static** **void** main(String[] args) {

**char** str1[] = ("geeksforgeek").toCharArray();

**char** str2[] = ("forgeeksgeeks").toCharArray();

System.***out***.println(*areAnagram*(str1, str2));

}

**static** **boolean** areAnagram(**char** input1[], **char** input2[]) {

**int** count[] = **new** **int**[*NO\_OF\_CHARS*];

**if** (input1.length != input2.length) {

**return** **false**;

}

**int** i;

**for** (i = 0; i < input1.length && i < input2.length; i++) {

**char** c1 = input1[i];

**char** c2 = input2[i];

count[c1] = count[c1]+1;

count[c2] = count[c2]-1;

}

**for** (i = 0; i < *NO\_OF\_CHARS*; i++) {

**if** (count[i] != 0) {

**return** **false**;

}

}

**return** **true**;

}

}

6. **IsSubString:**

// Java program to check if a string is

// substring of other.

**public** **class** SIsSubString {

// Function to find if S1 is a substring of S2

**static** **int** findSubstring(String S1, String S2) {

**int** M = S1.length();

**int** N = S2.length();

// Iterate through S2

**for** (**int** i = 0; i <= N - M; i++) {

// Check for substring match

**int** j;

**for** (j = 0; j < M; j++) {

// Mismatch found

**if** (S2.charAt(i + j) != S1.charAt(j)) {

**break**;

}

}

// If we completed the inner loop, we found a match

**if** (j == M) {

// Return starting index

**return** i;

}

}

// No match found

**return** -1;

}

**public** **static** **void** main(String[] args) {

String S1 = "for";

String S2 = "geeksforgeeks";

**int** result = *findSubstring*(S1, S2);

**if** (result != -1) {

System.***out***.println(result);

} **else** {

System.***out***.println(-1);

}

}

}

7. **ReverseWordsString:**

**class** TReverseWordsString {

**public** **static** **void** main(String[] args) {

String str = "I AM A GEEK";

// String str = "GEEK A AM I";

String wordReverse = *wordReverse*(str);

System.***out***.print(wordReverse);

*wordReverseStringBuffer*();

*printWordsReverse*();

}

**static** String wordReverse(String str) {

**int** i = str.length() - 1;

**int** start, end = i + 1;

String result = "";

**while** (i >= 0) {

**if** (str.charAt(i) == ' ') {

start = i + 1;

**while** (start != end)

result += str.charAt(start++);

result += ' ';

end = i;

}

i--;

}

System.***out***.println(result);

start = 0;

**while** (start != end)

result += str.charAt(start++);

**return** result;

}

**public** **static** **void** wordReverseStringBuffer() {

String str = "Hello World Sridhar";

String[] words = str.split(" ");

StringBuilder reversed = **new** StringBuilder();

**for** (**int** i = words.length - 1; i >= 0; i--) {

reversed.append(words[i]).append(" ");

}

String substring = reversed.substring(0, reversed.length()-1);

System.***out***.println("Reversed Words: " + substring);

}

**public** **static** **void** printWordsReverse() {

// **TODO** Auto-generated method stub

String str = "Welcome To Edureka";

String[] strArray = str.split(" ");

String result="";

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

String word=strArray[i];

result += *reverseString*(word) + " ";

}

String resultStr = result.substring(0, result.length()-1);

String reverseString = *reverseString*(resultStr);

System.***out***.println(reverseString);

}

**private** **static** String reverseString(String str) {

StringBuilder strBuilder= **new** StringBuilder(str);

**for**(**int** i=0, j=strBuilder.length()-1;i<j;i++,j--) {

**char** charAt = strBuilder.charAt(i);

strBuilder.setCharAt(i, strBuilder.charAt(j));

strBuilder.setCharAt(j, charAt);

}

**return** strBuilder.toString();

}

}

8. **ReverseWordsWithoutSplit**:

**public** **class** TReverseWordsWithoutSplit {

**public** **static** **void** main(String args[]) {

/\*String s=new String("abcd xyz");

char[] charArray = s.toCharArray();

int n=charArray.length;

for (int i=0;i<charArray.length/2; i++) {

char c = charArray[n-i-1];

charArray[n-i-1]=charArray[i];

charArray[i]=c;

}

System.out.println(String.valueOf(charArray));

\*/

//indexOf,charAt,setCharAt,length

StringBuilder s=**new** StringBuilder("abcd xyz");

System.***out***.println(s);

**int** index=0;

**int** n=s.length();

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

**char** c = s.charAt(i);

**if**(c==' '||i==n-1) {

**if**(i==n-1)

i=i+1;

/\*for (int j=index,k=0;j<(i+index)/2; j++,k++) {

char c1 = s.charAt(i-k-1);

s.setCharAt(i-k-1, s.charAt(j));

s.setCharAt(j,c1);

}\*/

**for** (**int** j=index,k=i-1;j<k; j++,k--) {

**char** c1 = s.charAt(j);

s.setCharAt(j, s.charAt(k));

s.setCharAt(k,c1);

}

index=i+1;

}

}

System.***out***.println(s);

*stringReveerse*(s);

System.***out***.println(s);

}

**private** **static** **void** stringReveerse(StringBuilder s) {

**int** n=s.length();

**for** (**int** i=0;i<s.length()/2; i++) {

**char** c = s.charAt(n-i-1);

s.setCharAt(n-i-1, s.charAt(i));

s.setCharAt(i,c);

}

}

}