**PROGRAMMING FOR JAVA**

**EASY PROGRAMS:**

**1. TWO SUM.**

import java.util.HashMap;

import java.util.Map;

public class R192211656 {

public static int[] findTwoSum(int[] nums, int target) {

Map<Integer, Integer> map = new HashMap<>();

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

int complement = target - nums[i];

if (map.containsKey(complement)) {

return new int[] { map.get(complement), i };

}

map.put(nums[i], i);

}

throw new IllegalArgumentException("No two sum solution");

}

public static void main(String[] args) {

int[] nums = {3,2,4};

int target = 6;

int[] result = findTwoSum(nums, target);

System.out.println("Indices: " + result[0] + ", " + result[1]);

}

}

**OUTPUT:**

**Indices: 1, 2**

**2. PALINDROME NUMBER:**

import java.util.InputMismatchException;

import java.util.NoSuchElementException;

import java.util.Scanner;

public class R192211656 {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

try {

System.out.print("Enter an integer: ");

int x = scanner.nextInt();

boolean isPalindrome = isPalindrome(x);

if (isPalindrome) {

System.out.println("Output: true");

} else {

System.out.println("Output: false");

}

} catch (InputMismatchException e) {

System.out.println("Invalid input. Please enter a valid integer.");

} catch (NoSuchElementException e) {

System.out.println("No input was provided. Please try again.");

} finally {

scanner.close();

}

}

public static boolean isPalindrome(int x) {

if (x < 0 || (x % 10 == 0 && x != 0)) {

return false;

}

int reversed = 0;

int original = x;

while (x != 0) {

int digit = x % 10;

reversed = reversed \* 10 + digit;

x /= 10;

}

return original == reversed;

}

}

**OUTPUT:**

**Enter an integer: 121**

**True.**

**3.MERGE TWO SHORTED LIST.**

public class R192211656 {

public static void main(String[] args) {

int[] arr1 = {1, 2, 4};

int[] arr2 = {1, 3, 4};

int[] mergedArray = mergeSortedArrays(arr1, arr2);

System.out.println("Merged array:");

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

System.out.print(mergedArray[i] + " ");

}

}

public static int[] mergeSortedArrays(int[] arr1, int[] arr2) {

int[] mergedArray = new int[arr1.length + arr2.length];

int i = 0, j = 0, k = 0;

while (i < arr1.length && j < arr2.length) {

if (arr1[i] <= arr2[j]) {

mergedArray[k++] = arr1[i++];

} else {

mergedArray[k++] = arr2[j++];

}

}

while (i < arr1.length) {

mergedArray[k++] = arr1[i++];

}

while (j < arr2.length) {

mergedArray[k++] = arr2[j++];

}

return mergedArray;

}

}

**OUTPUT:**

**Merged array: 1 1 2 3 4 4**

**4.PERFECT NUMBER.**

public class R192211656 {

public static void main(String[] args) {

int number = 28;

int sumOfDivisors = 0;

for (int i = 1; i < number; i++) {

if (number % i == 0) {

sumOfDivisors += i;

}

}

if (sumOfDivisors == number) {

System.out.println(number + " is a perfect number.");

} else {

System.out.println(number + " is not a perfect number.");

}

}

}

**OUTPUT:**

**28 is a perfect number.**

**5.PALINDROME.**

public class R192211656 {

public static void main(String[] args) {

String inputString = "racecar";

boolean isPalindrome = true;

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

if (inputString.charAt(i) != inputString.charAt(j)) {

isPalindrome = false;

break;

}

}

if (isPalindrome) {

System.out.println("The string is a palindrome.");

} else {

System.out.println("The string is not a palindrome.");

}

}

}

**OUTPUT:**

**The string is a palindrome.**

**6.SORT – I**

import java.util.Arrays;

class R192211656 {

public static void main(String[] args) {

int[] arr = {9, 3, 7, 5, 6, 4, 2, 8};

for (int i = 1; i < arr.length; i++) {

int key = arr[i];

int j = i - 1;

while (j >= 0 && arr[j] > key) {

arr[j + 1] = arr[j];

j = j - 1;

}

arr[j + 1] = key;

}

System.out.println("Sorted array: " + Arrays.toString(arr));

}

}

**OUTPUT:**

**Sorted array: [2, 3, 4, 5, 6, 7, 8, 9]**

**7.** **ROMAN TO INTEGERS.**

public class R192211656 {

public static String intToRoman(int num) {

String[] romanSymbols = {"M", "CM", "D", "CD", "C", "XC", "L", "XL", "X", "IX", "V", "IV", "I"};

int[] values = {1000, 900, 500, 400, 100, 90, 50, 40, 10, 9, 5, 4, 1};

StringBuilder roman = new StringBuilder();

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

while (num >= values[i]) {

roman.append(romanSymbols[i]);

num -= values[i];

}

}

return roman.toString();

}

public static void main(String[] args) {

int number = 1994;

String romanNumeral = intToRoman(number);

System.out.println("The Roman numeral for " + number + " is: " + romanNumeral);

}

}

**OUTPUT:**

**The Roman numeral for 1994 is: MCMXCIV**

**8.LONGEST COMMON PREFIX.**

public class R192211656 {

public static String longestCommonPrefix(String[] strs) {

if (strs == null || strs.length == 0) {

return "";

}

String prefix = strs[0];

for (int i = 1; i < strs.length; i++) {

while (strs[i].indexOf(prefix) != 0) {

prefix = prefix.substring(0, prefix.length() - 1);

if (prefix.isEmpty()) {

return "";

}

}

}

return prefix;

}

public static void main(String[] args) {

String[] strs = {"flower", "flow", "flight"};

String result = longestCommonPrefix(strs);

System.out.println("The longest common prefix is: " + result);

}

}

**OUTPUT:**

**The longest common prefix is: fl**

**9.ADD BINARY.**

public class R192211656 {

public static String addBinary(String a, String b) {

int sum = 0;

int i = a.length() - 1;

int j = b.length() - 1;

int carry = 0;

StringBuilder result = new StringBuilder();

while (i >= 0 || j >= 0 || carry != 0) {

sum = carry;

if (i >= 0) {

sum += a.charAt(i--) - '0';

}

if (j >= 0) {

sum += b.charAt(j--) - '0';

}

carry = sum / 2;

result.append(sum % 2);

}

return result.reverse().toString();

}

public static void main(String[] args) {

String a = "11";

String b = "1";

String result = addBinary(a, b);

System.out.println("The sum of " + a + " and " + b + " is " + result);

}

}

**OUTPUT:**

**The sum of 11 and 1 is 100.**

**10.sqrt(X).**

public class R192211656 {

public static void main(String[] args) {

int x = 8;

int sqrt = sqrtRounded(x);

System.out.println("The square root of " + x + " rounded to the nearest integer is " + sqrt);

}

public static int sqrtRounded(int x) {

if (x < 0) {

throw new ArithmeticException("Square root of negative number");

}

if (x == 0 || x == 1) {

return x;

}

int start = 1;

int end = x;

int result = 0;

while (start <= end) {

int mid = start + (end - start) / 2;

long square = (long) mid \* mid;

if (square == x) {

return mid;

} else if (square < x) {

start = mid + 1;

result = mid;

} else {

end = mid - 1;

}

}

return result;

}

}

**OUTPUT:**

**The square root of 8 rounded to the nearest integer is 2**