**Coding Interview Questions**

Table of Contents

[1. Write the program to print the pyramid pattern of the symbol with right alignment? 2](#_Toc482425426)

[2. Write the program to sort the array in ascending order using Bubble sort? 3](#_Toc482425427)

[3. Write the program to sort the array in descending order using Bubble sort? 4](#_Toc482425428)

[4. Write the program to reverse a string with maintaining special character position? 5](#_Toc482425429)

[5. Example for Comparable interface? 7](#_Toc482425430)

[6. Example for Comparator interface? 10](#_Toc482425431)

[7. Write the code to print the duplicate character in the given string? 14](#_Toc482425432)

[8. Find Missing Number in the integer array? 15](#_Toc482425433)

[9. How to convert Map into List? 16](#_Toc482425434)

[10. How to sort HashMap based on values? 17](#_Toc482425435)

# Write the program to print the pyramid pattern of the symbol with right alignment?

#

##

###

####

#####

public class Pattern {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.println("Enter the input: ");

int numOfRows = scanner.nextInt();

// print pramid pattern of #

String symbol = "#";

pramidPattern(numOfRows, symbol);

}

public static void pramidPattern(int rows, String symbol) {

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

System.out.print(print(rows-i, " "));

System.out.println(print(rows-(rows-i), symbol));

}

}

public static String print(int count, String symbol) {

String str = new String(new char[count]).replace("\0", symbol);

return str;

}

}

# Write the program to sort the array in ascending order using Bubble sort?

import java.util.Arrays;

public class BubbleSort {

public static void main(String[] args) {

int[] array = {5,4,2,3,9,1};

System.out.println( "Un sorted Array :" + Arrays.toString(array));

bubbleSort(array);

}

public static void bubbleSort(int[] unsorted) {

int length = unsorted.length;

for (int i = 0; i < length-1; i++) {

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

if(unsorted[j] > unsorted[j+1]) {

int tmp = unsorted[j];

unsorted[j] = unsorted[j+1];

unsorted[j+1] = tmp;

}

}

}

System.out.printf("Sorted Array", Arrays.toString(unsorted));

}

}

# Write the program to sort the array in descending order using Bubble sort?

import java.util.Arrays;

public class BubbleSortDsc {

public static void main(String[] args) {

int[] inputArray = {34,62,6,89,3,7,23};

System.out.println("Unsorted Array : " + Arrays.toString(inputArray));

int[] output = sort(inputArray);

System.out.println("Sorted Array : " + Arrays.toString(output));

}

public static int[] sort(int[] inputArray) {

int length = inputArray.length;

for (int i = 0; i < length -1; i++) {

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

if (inputArray[j] < inputArray[j+1]) {

int tmp = inputArray[j];

inputArray[j] = inputArray[j+1];

inputArray[j+1] = tmp;

}

}

}

return inputArray;

}

}

# Write the program to reverse a string with maintaining special character position?

public class ReverseString1 {

public static void main(String[] args) {

String input = "#Regu@gmail.com#";

System.out.println("Input String : " + input);

String output = reverse(input);

System.out.println("Reversed String : " + output);

}

/\*\*

\* Reverse a string with maintaining special character position.

\* Algorithm :

\* 1) Let input string be 'inputArray[]' and length of string be 'length'

\* 2) left = 0, right = length-1

\* 3) While left is smaller than right, do following

\* a) If inputArray[left] is not an alphabetic character, do left++

\* b) Else If inputArray[right] is not an alphabetic character, do right--

\* c) Else swap inputArray[left] and inputArray[right]

\*

\* @param input : Input string

\* @return reverse string

\*/

public static String reverse(String input) {

char[] inputArray = input.toCharArray();

// length of the array

int length = inputArray.length;

int left = 0;

int right = length -1;

while (left < right) {

// check if the first character is special character

if (String.valueOf(inputArray[left]).matches("[^a-z A-Z 0-9]")) {

left++;

// check if the last character is special character

} else if (String.valueOf(inputArray[right]).matches("[^a-z A-Z 0-9]")) {

right--;

} else {

char temp = inputArray[left];

inputArray[left] = inputArray[right];

inputArray[right] = temp;

left++;

right--;

}

}

return String.valueOf(inputArray);

}

}

# Example for Comparable interface?

public class ComparableExample {

public ComparableExample() {

}

public static void main(String[] args) {

ComparableExample comparable = new ComparableExample();

Employee employee1 = comparable.new Employee();

employee1.setEmployeeName("Regu");

employee1.setEmployeeId(1);

employee1.setAddess("Bangalore");

Employee employee2 = comparable.new Employee();

employee2.setEmployeeName("Anand");

employee2.setEmployeeId(3);

employee2.setAddess("Mumbai");

Employee employee3 = comparable.new Employee();

employee3.setEmployeeName("Santhosh");

employee3.setEmployeeId(2);

employee3.setAddess("Chennai");

List<Employee> list = new ArrayList<>();

list.add(employee1);

list.add(employee2);

list.add(employee3);

// before java1.8

//Collections.sort(list);

// from java 1.8

// sort by employee id

list.sort(Comparator.comparing(e -> e.getEmployeeId()));

System.out.println("Sort Employee by employee id");

list.forEach(emp -> System.out.println("Emp Id : " + emp.getEmployeeId() + " "+

"Emp Name : " + emp.getEmployeeName() + " " +

"Emp Addredd : " + emp.getAddess() ));

// sort by employee name

list.sort(Comparator.comparing(e -> e.getEmployeeName()));

System.out.println("Sort Employee by employee name");

list.forEach(emp -> System.out.println("Emp Id : " + emp.getEmployeeId() + " "+

"Emp Name : " + emp.getEmployeeName() + " " +

"Emp Addredd : " + emp.getAddess() ));

}

class Employee implements Comparable {

private int employeeId;

private String employeeName;

private String addess;

public Employee() {

}

public int getEmployeeId() {

return employeeId;

}

public void setEmployeeId(int employeeId) {

this.employeeId = employeeId;

}

public String getEmployeeName() {

return employeeName;

}

public void setEmployeeName(String employeeName) {

this.employeeName = employeeName;

}

public String getAddess() {

return addess;

}

public void setAddess(String addess) {

this.addess = addess;

}

@Override

public int compareTo(Object obj) {

Employee emp = (Employee) obj;

return this.employeeId - emp.employeeId;

}

}

}

# Example for Comparator interface?

package com.exxample.coding;

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

public class ComparatorExample {

public ComparatorExample() {

}

public static void main(String[] args) {

ComparatorExample comparator = new ComparatorExample();

Employee employee1 = comparator.new Employee();

employee1.setEmployeeName("Regu");

employee1.setEmployeeId(1);

employee1.setAddess("Bangalore");

Employee employee2 = comparator.new Employee();

employee2.setEmployeeName("Anand");

employee2.setEmployeeId(3);

employee2.setAddess("Mumbai");

Employee employee3 = comparator.new Employee();

employee3.setEmployeeName("Santhosh");

employee3.setEmployeeId(2);

employee3.setAddess("Chennai");

List<Employee> list = new ArrayList<>();

list.add(employee1);

list.add(employee2);

list.add(employee3);

Collections.sort(list);

System.out.println("Sort Employee by employee id");

list.forEach(emp -> System.out.println("Emp Id : " + emp.getEmployeeId() + " "+

"Emp Name : " + emp.getEmployeeName() + " " +

"Emp Addredd : " + emp.getAddess() ));

Collections.sort(list, comparator.new sortEmployeeName());

System.out.println("Sort Employee by employee name");

list.forEach(emp -> System.out.println("Emp Id : " + emp.getEmployeeId() + " "+

"Emp Name : " + emp.getEmployeeName() + " " +

"Emp Addredd : " + emp.getAddess() ));

}

class Employee implements Comparable {

private int employeeId;

private String employeeName;

private String addess;

public Employee() {

}

public int getEmployeeId() {

return employeeId;

}

public void setEmployeeId(int employeeId) {

this.employeeId = employeeId;

}

public String getEmployeeName() {

return employeeName;

}

public void setEmployeeName(String employeeName) {

this.employeeName = employeeName;

}

public String getAddess() {

return addess;

}

public void setAddess(String addess) {

this.addess = addess;

}

@Override

public int compareTo(Object obj) {

Employee emp = (Employee) obj;

return this.employeeId - emp.employeeId;

}

}

class sortEmployeeName implements Comparator {

@Override

public int compare(Object o1, Object o2) {

Employee emp1 = (Employee)o1;

Employee emp2 = (Employee)o2;

return emp1.getEmployeeName().compareTo(emp2.getEmployeeName());

}

}

}

# Write the code to print the duplicate character in the given string?

public class FindDuplicateCharacterInString {

public static void main(String[] args) {

// input value

String input = "java programming";

printDulicateCharacter(input);

}

public static void printDulicateCharacter (String input) {

char[] charArray = input.toCharArray();

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

for(char c: charArray) {

if(map.containsKey(c)) {

map.put(c, map.get(c)+1);

} else {

map.put(c, 1);

}

}

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

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

if (entry.getValue() > 1) {

System.out.println(entry.getKey() + " - " + entry.getValue());

}

}

}

}

# Find Missing Number in the integer array?

package com.example.datastructure;

import java.util.Arrays;

import java.util.BitSet;

public class MissingNumberInArray {

public static void main(String[] args) {

int[] inputArray = {1,2,4,6,3,7,8};

int count = 10;

System.out.println("Input Array: " + Arrays.toString(inputArray));

printMissingNumers(inputArray, count);

}

public static void printMissingNumers(int[] inputArray, int count) {

int missingCount = count-inputArray.length;

BitSet bitSet = new BitSet(count); // initialize the bitset

for (int number: inputArray) {

bitSet.set(number-1); // Index will always start from 0

}

int missingIndex = 0;

for (int i=0; i<missingCount; i++) {

missingIndex = bitSet.nextClearBit(missingIndex); // missingIndex - fromIndex

System.out.println(++missingIndex);

}

}

}

# How to convert Map into List?

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map.Entry;

/\*\*

 \***Converting HashMap into ArrayList in Java**

 \*/

public class MaptoListJava {

    public static void main(String... args) {

        HashMap<String, String> personalLoanOffers = new HashMap<String, String>();

**// preparing hashmap with keys and values**

        personalLoanOffers.put("personal loan by DBS", "Interest rate low");

        personalLoanOffers.put("personal loan by Standard Charted", "Interest rate low");

        personalLoanOffers.put("HSBC personal loan by DBS", "14%");

        personalLoanOffers.put("Bankd of America Personal loan", "11%");

        System.out.println("Size of personalLoanOffers Map: " + personalLoanOffers.size());

**//Converting HashMap keys into ArrayList**

        List<String> keyList = new ArrayList<String>(personalLoanOffers.keySet());

        System.out.println("Size of Key list from Map: " + keyList.size());

**//Converting HashMap Values into ArrayList**

        List<String> valueList = new ArrayList<String>(personalLoanOffers.values());

        System.out.println("Size of Value list from Map: " + valueList.size());

        List<Entry> entryList = new ArrayList<Entry>(personalLoanOffers.entrySet());

        System.out.println("Size of Entry list from Map: " + entryList.size());

    }

}

**Output:**

Size of personalLoanOffers Map: 4

Size of Key list from Map: 4

Size of Value list from Map: 4

Size of Entry list from Map: 4

# How to sort HashMap based on values?

package com.example.collection;

import java.util.Collections;

import java.util.Comparator;

import java.util.HashMap;

import java.util.LinkedHashMap;

import java.util.LinkedList;

import java.util.List;

import java.util.Map;

import java.util.Map.Entry;

public class SortMapByValue {

public static void main(String[] args) {

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

map.put("India", 1);

map.put("America", 3);

map.put("UK", 2);

System.out.println("Sorted map by values: " + sortByValue(map));

// output Sorted map by values: {India=1, UK=2, America=3}

System.out.println("Sort map by keys: " + sortByKey(map));

}

public static <K extends Comparable, V extends Comparable> Map<K, V> sortByValue(Map<K, V> map) {

Map<K, V> sortedMap = new LinkedHashMap<>(); // store the keys in order

List<Map.Entry<K, V>> list = new LinkedList<>(map.entrySet());

// keep the values in order

Collections.sort(list, new Comparator<Map.Entry<K, V>>() {

@Override

public int compare(Entry<K, V> e1, Entry<K, V> e2) {

return e1.getValue().compareTo(e2.getValue());

}

});

// iterate list and store into map

for (Map.Entry<K, V> entry : list) {

sortedMap.put(entry.getKey(), entry.getValue());

}

return sortedMap;

}

public static <K extends Comparable, V extends Comparable> Map<K, V> sortByKey(Map<K, V> map) {

Map<K, V> sortedMap = new LinkedHashMap<>();

List<K> list = new LinkedList<>(map.keySet());

Collections.sort(list);

list.forEach(key -> {

sortedMap.put(key, map.get(key));

});

return sortedMap;

}

}