/\*Q1. Consider there is a 3 Boolean variable called a, b, c. Check if at least two out of three Booleans are true\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_1 {

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

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

System.***out***.print("Enter boolean value for a:");

**boolean** a= sc.nextBoolean();

System.***out***.print("Enter boolean value for b:");

**boolean** b= sc.nextBoolean();

System.***out***.print("Enter boolean value for c:");

**boolean** c = sc.nextBoolean();

**if**(a==**true** && b == **true** && c==**true**)

{

System.***out***.println("a,b and c variables are true");

}

**else** **if**(a==**true** && b == **true**)

{

System.***out***.println("a and b variables are true");

}

**else** **if**(b==**true** && c== **true**)

{

System.***out***.println("b and c variables are true");

}

**else** **if**(a==**true** && c==**true**)

{

System.***out***.println("a and c variables are true");

}

**else**

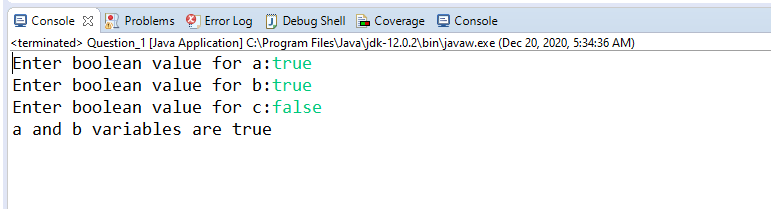
{

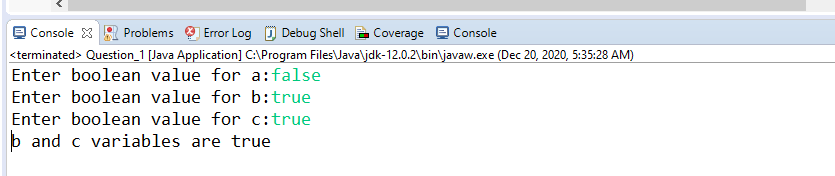
System.***out***.println("Enter two variables(a,b,c) atleast true");

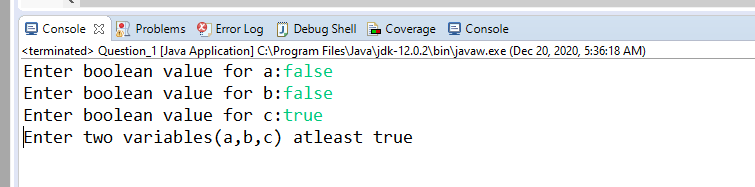
}

}

}







/\*Q2. write a program to find factorial (Non Recursive)\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_2 {

**static** **int** fact(**int** n)//Non-recursive function to find factorial of a given number

{

**int** result =1;

**while**(n!=1)

{

result = result\*n;

n--;

}

**return** result;

}

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

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

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

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

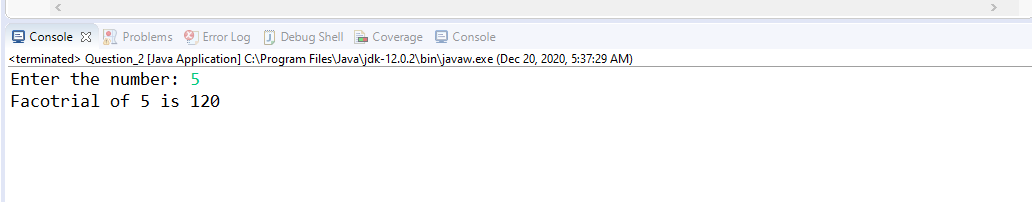
**int** num = scan.nextInt() ;

**int** result = *fact*(num);

System.***out***.println("Facotrial of " + num + " is " + result);

}

}



/\*Q3. Given an array of integers, sort the integer values.\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Arrays;

**public** **class** Question\_3 {

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

{

// Our arr contains 8 elements

**int**[] arr = {13, 7, 6, 45, 21, 9, 101, 102};

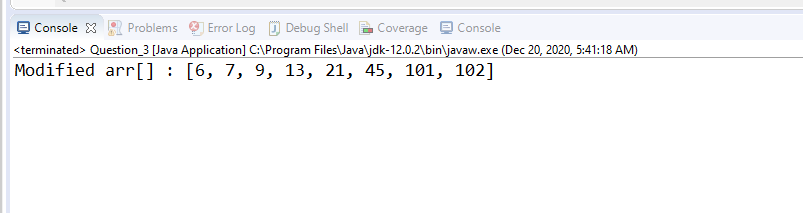
Arrays.*sort*(arr);

System.***out***.printf("Modified arr[] : %s",

Arrays.*toString*(arr));

}

}



/\*Q4. Given an array of integers check the Palindrome of the series.\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_4 {

**public** **static** **boolean** isPalindrome(**int**[] arr)

{

**int** i = 0, j = arr.length - 1;

// While there are characters toc compare

**while** (i <= j) {

// If there is a mismatch

**if** (arr[i] != arr[j])

**return** **false**;

// Increment first pointer and

// decrement the other

i++;

j--;

}

**return** **true**;

}

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

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

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

**int**[] array = **new** **int**[5];

System.***out***.print("Enter five integers: ");

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

{

//reading array elements from the user

array[i]=sc.nextInt();

}

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

**for**(**int** i:array)

{

System.***out***.print(i +"," );

}

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

**if** (*isPalindrome*(array))

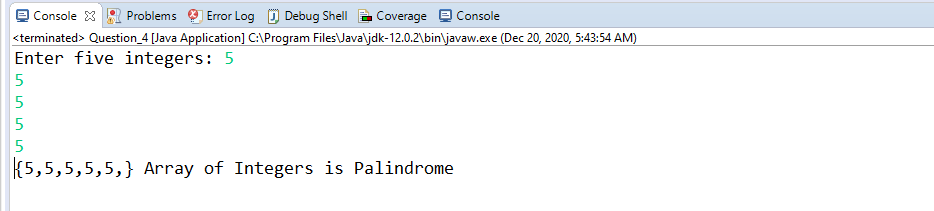
System.***out***.print(" Array of Integers is Palindrome");

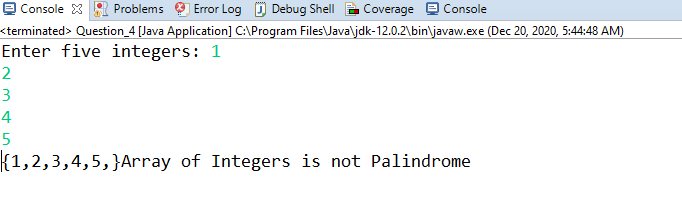
**else**

System.***out***.print("Array of Integers is not Palindrome");

}

}





/\*Q5. Given an array prints the unique numbers and also print the number of occurrences of duplicate numbers.\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_5 {

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

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

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

**int**[] array = **new** **int**[5];

System.***out***.print("Enter five integers: ");

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

{

//reading array elements from the user

array[i]=sc.nextInt();

}

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

**for**(**int** i:array)

{

System.***out***.print(i +"," );

}

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

System.***out***.println("Duplicate elements in given array: \n");

//Searches for duplicate element

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

{

**int** j;

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

{

**if**(array[i] == array[j])

**break**;

}

**if** (i!=j) //Print duplicate number only once

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

}

System.***out***.println("Unique elements in given array: \n");

//Searches for Unique element

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

{

// Check if the picked element

// is already printed

**int** j;

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

{

**if** (array[i] == array[j])

**break**;

}

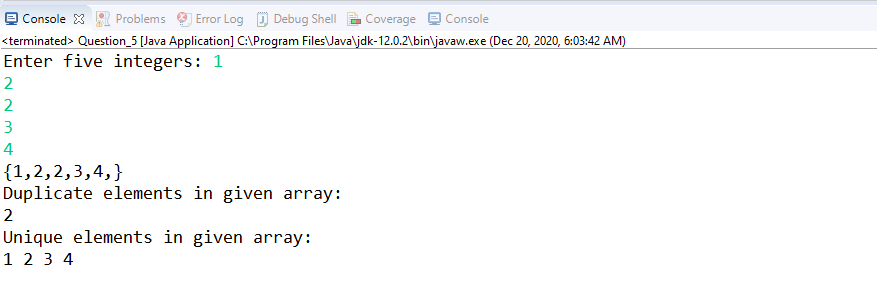
**if** (i == j)

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

}

}

}



/\*Q6. WJP to perform ascending order Selection sort\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_6 {

**static** **void** Selectionsort(**int** arr[])

{

**int** n= arr.length;

**int** temp;

**for** (**int** i = 0; i<n-1; i++) //First iteration is length of array minus 1 times

{

**int** min = i;

**for** (**int** j=i+1;j<n;j++)// second iteration is to find minimun value in the unsorted subarray

{

**if**( arr[j]<arr[min])

{

min = j; // end of the for loop will find the minimum value

}

}

**if** (min!=i)

{

temp = arr[i];

arr[i] = arr[min];

arr[min]= temp;

}

}

}

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

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

//int Arr[] = {1,2,1,2,8,6,7,3,7,5,2};

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

**int**[] Arr = **new** **int**[10];

System.***out***.print("Enter 10 integers: ");

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

{

//reading array elements from the user

Arr[i]=sc.nextInt();

}

System.***out***.print("Array before Selection sort: ");

**for**(**int** i : Arr)

{

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

}

*Selectionsort*(Arr);

System.***out***.print("\nArray after Selection sort: ");

**for**(**int** i : Arr)

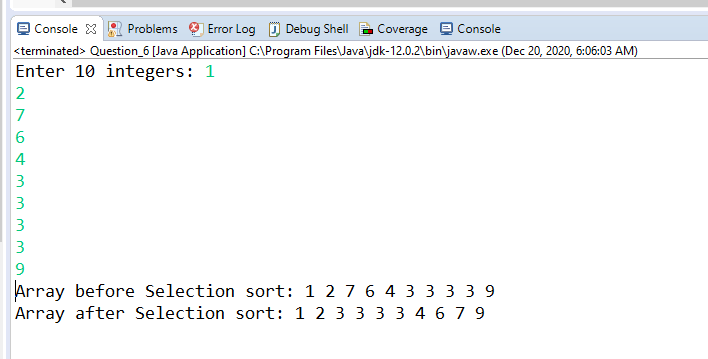
{

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

}

}

}



/\*Q7. What are different ways to create String Object?\*/

**package** com.training.Java\_Hackthon;

**public** **class** Question\_7 {

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

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

String str = "Java"; //Using String literal

System.***out***.println("Using Literal way:"+str);

String str1 = **new** String("Hello");//Using new keyword

System.***out***.println("Using New keyword:"+str1);

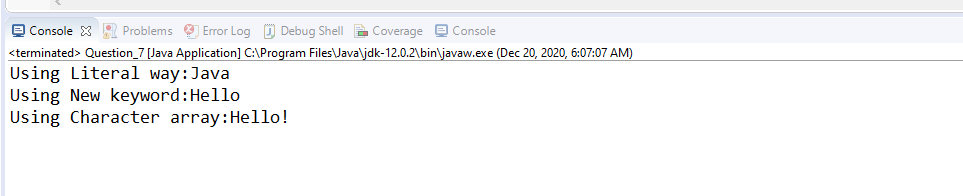
**char** ch[]={ 'H','e','l','l','o','!'}; //Using character array

String str2=**new** String(ch);

System.***out***.println("Using Character array:"+str2);

}

}



/\*Q8. How can we make String upper case to lower case?\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_8 {

**public** **static** String toLowerCase(String str)

{

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

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

{

**int** char1 = str.charAt(i);

**if** (char1>64&&char1<91) //Convert into lowercase

{

char1 = char1+32;

System.***out***.print((**char**)char1) ;

}

**else** **if** (char1>96&&char1<123)// for special keyboard keys Alt,tab

{

System.***out***.print((**char**)char1);

}

**else** **if** (char1==32) //For space character

{

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

}

}

**return** str;

}

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

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

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

System.***out***.print("Enter String in Uppercase: ");

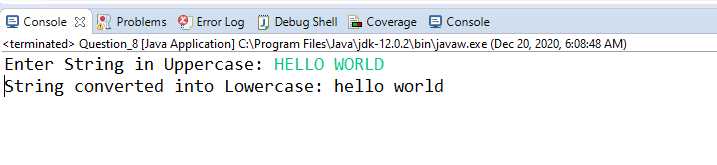
String str = sc.nextLine();

System.***out***.print("String converted into Lowercase: ");

*toLowerCase*(str);

}

}



/\*Q9. How can we make String Lower case to Upper case?\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_9 {

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

{

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

System.***out***.print("Enter String in Lowercase: ");

String str = sc.nextLine();

**char**[] ch = str.toCharArray();

**for**(**int** a = 0; a < ch.length; a++)

{

**if**(ch[a] >= 'a' && ch[a] <= 'z')

{

ch[a] = (**char**)((**int**)ch[a] -32);

}

}

System.***out***.print("Converted String in Uppercase: ");

**for**(**int** a = 0; a < ch.length; a++)

{

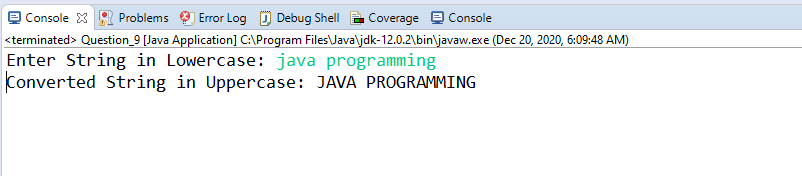
System.***out***.print(ch[a]);

}

sc.close();

}

}



//Q10. What is String subsequence method?

**package** com.training.Java\_Hackthon;

**public** **class** Question\_10 {

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

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

/\*String subSequence() method is in-built java Function that returns char sequence

\* subSequence starts with the char value at the specified index and ends with the char char value

\* at(end-1)

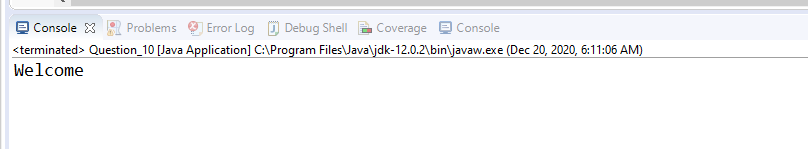
\*/

String str="Welcome Java";

System.***out***.println(str.subSequence(0, 7));//It will print Welcome

}

}



/\*Q11. How to Split String in java?\*/

**package** com.training.Java\_Hackthon;

**public** **class** Question\_11 {

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

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

String str = "3435-3773-3838";

/\* Split method breaks a given string around

matches of the given regular expression\*/

String[] arrOfStr = str.split("-");

**for** (String a : arrOfStr)

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

}

}

/\*Q11. How to Split String in java?\*/

**package** com.training.Java\_Hackthon;

**public** **class** Question\_11 {

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

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

String str = "3435-3773-3838";

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

/\* Split method breaks a given string around

matches of the given regular expression\*/

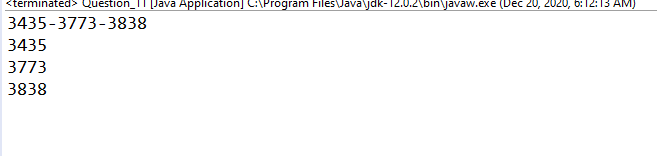
String[] arrOfStr = str.split("-");

**for** (String a : arrOfStr)

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

}

}



/\*Q12. Write a program to check palindrome (MalayalaM) for both numbers and string?\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_12 {

**static** **boolean** isPalindrome(String str)

{

// Pointers pointing to the beginning

// and the end of the string

**int** i = 0, j = str.length() - 1;

// While there are characters toc compare

**while** (i <= j) {

// If there is a mismatch

**if** (str.charAt(i) != str.charAt(j))

**return** **false**;

// Increment first pointer and

// decrement the other

i++;

j--;

}

// Given string is a palindrome

**return** **true**;

}

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

{

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

System.***out***.print("Enter a String :");

String str = sc.next();

**if** (*isPalindrome*(str))

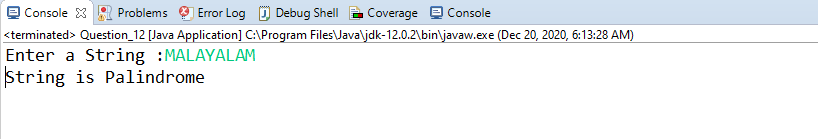
System.***out***.print("String is Palindrome");

**else**

System.***out***.print("String is not Palindrome");

}

}



/\*Q13. Given a string print the reverse of the string.(Input: Java Code Output: edoC avaJ)\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_13 {

**public** **static** String reverse(String Str)

{

String rev = "";

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

{

rev= rev + Str.charAt(i);

}

**return** rev;

}

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

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

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

System.***out***.print("Enter the String : ");

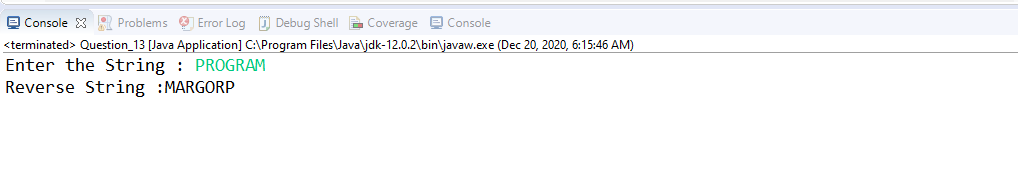
String str = sc.next();

// String rev =reverse(str);

System.***out***.println("Reverse String :" +*reverse*(str));

}

}



/\*Q14. Given a string print the reverse of the words string.(Input: Java Code Output: Code Java)\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_14 {

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

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

System.***out***.print("Enter the String :");

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

String str = sc.nextLine();

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

String revString = "";

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

{

String word = words[i];

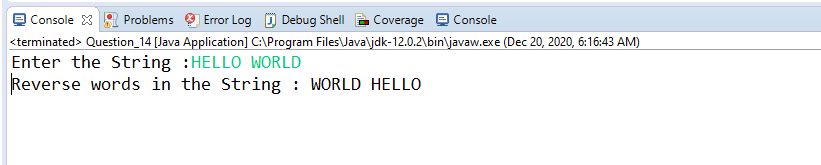
revString = revString + word + " ";

}

System.***out***.println("Reverse words in the String : "+ revString);

}

}



/\*Q15. Given a string print the unique words of the string.\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_15 {

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

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

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

System.***out***.println("Enter String :");

String str = sc.nextLine();

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

**int** count=0;

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

{ **if**(words[i].equals("") || words[i].equals(" "))

{

**continue**;

}

**else** {

**for**(**int** j=i+1;j<words.length;j++)

{

**if**(words[i].equals(words[j]))

{

count=1;

words[j]=" ";

**break**;

}

}

**if**(count==0)

{

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

}

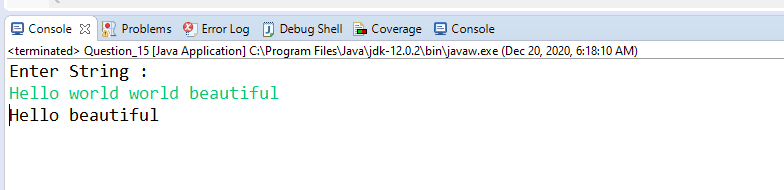
count=0;

}

}

}

}



/\*Q16. Write a method that will remove given character from the String?\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_16 {

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

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

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

System.***out***.println("Enter String :");

String str = sc.next();

System.***out***.println("Enter character to remove:");

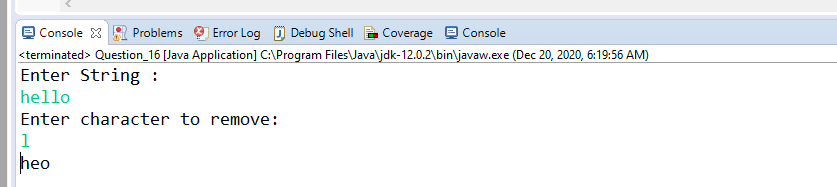
String ch = sc.next();

str = str.replaceAll(ch, "");

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

}

}



/\*Q17. WJP to find total number of integers, uppercase and lowercase character in the give string\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_17 {

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

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

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

System.***out***.println("Enter String :");

String str = sc.nextLine();

**int** totalChar = str.length();

**int** upperCase = 0;

**int** lowerCase = 0;

**int** digits = 0;

**int** others = 0;

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

**char** ch = str.charAt(i);

**if** (Character.*isUpperCase*(ch)) {

upperCase++;

}

**else** **if** (Character.*isLowerCase*(ch)) {

lowerCase++;

}

**else** **if** (Character.*isDigit*(ch)) {

digits++;

}

}

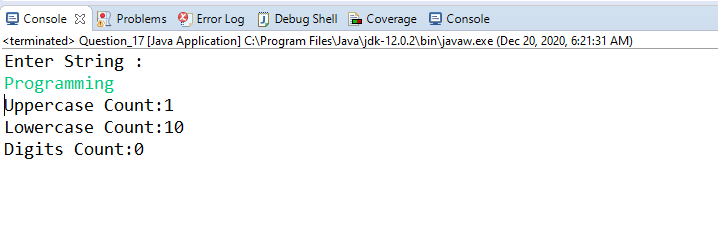
System.***out***.println("Uppercase Count:"+ upperCase);

System.***out***.println("Lowercase Count:"+ lowerCase);

System.***out***.println("Digits Count:"+ digits);

}

}



**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_18 {

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

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

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

System.***out***.println("Enter String :");

String str = sc.nextLine();

**char** string[] = str.toCharArray();

**int** count;

System.***out***.println("Duplicate characters in a given string: ");

//Counts each character present in the string

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

count = 1;

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

**if**(string[i] == string[j] && string[i] != ' ') {

count++;

//Set string[j] to 0 to avoid printing visited character

string[j] = '0';

}

}

//A character is considered as duplicate if count is greater than 1

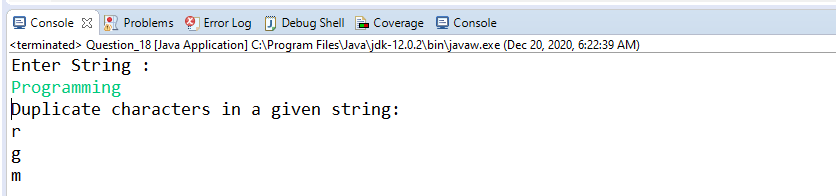
**if**(count > 1 && string[i] != '0')

System.***out***.println(string[i]);

}

}

}



/\*Q19. WJP to display number of occurrence of all character\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_19 {

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

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

System.***out***.println("Enter String :");

String str = sc.nextLine();

str= str.toLowerCase();

**for** ( **char** ch ='a';ch<='z';ch++)

{

**int** cnt = 0;

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

{

**if** (ch==str.charAt(i) && str.charAt(i)!=' ')

{

cnt++;

}

}

**if** (cnt!=0)

{

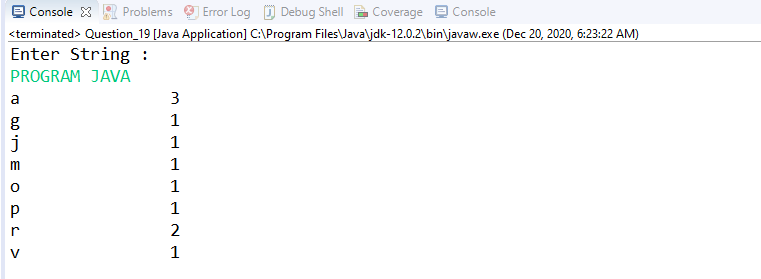
System.***out***.println(ch + "\t\t"+cnt);

}

}

}

}



/\*Q20. WJP to find total number of repeated integers, uppercase and lowercase character in the give string\*/

**package** com.training.Java\_Hackthon;

**import** java.util.HashMap;

**import** java.util.Map;

**import** java.util.Scanner;

**public** **class** Question\_20 {

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

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

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

System.***out***.println("Enter String :");

String str = sc.nextLine();

**int** cnt = 0;

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

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

{

**if** (map.containsKey(str.charAt(i)))

{

**int** count = map.get(str.charAt(i));

count++;

map.put(str.charAt(i), count);

}

**else**

{

map.put(str.charAt(i), 1);

}

}

**for** (Map.Entry<Character,Integer> entry : map.entrySet())

{

**if** (entry.getValue()>1 && entry.getKey()!=' ')

{

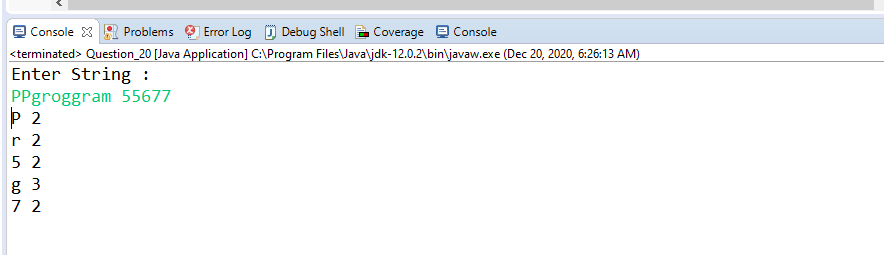
System.***out***.println(entry.getKey() + " " +entry.getValue());

}

}

}

}



/\*Convert String to int\*/

**package** com.training.Java\_Hackthon;

**public** **class** Question\_21 {

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

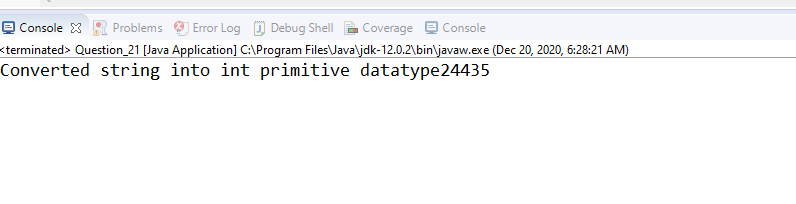
String str = "24435";

**int** i = Integer.*parseInt*(str);

System.***out***.println("Converted string into int primitive datatype"+ i);

}

}



/\*Q22. WJP to convert int to string\*/

**package** com.training.Java\_Hackthon;

**public** **class** Question\_22 {

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

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

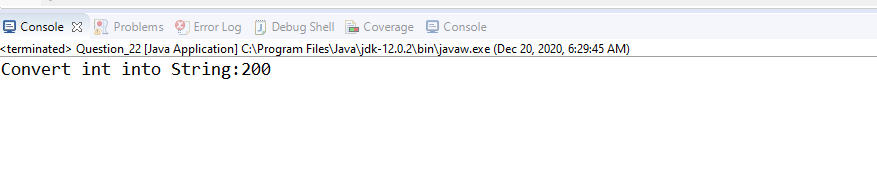
**int** i=200;

String s=String.*valueOf*(i);

System.***out***.println("Convert int into String:"+i); //it will return String "200"

}

}



/\*Q23. WJP to differentiate input as string, int or bool\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_23 {

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

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

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

System.***out***.println("Enter Input :");

**if** (input.hasNextInt()) System.***out***.println("This input is of type Integer.");

**else** **if** (input.hasNextFloat()) System.***out***.println("This input is of type Float.");

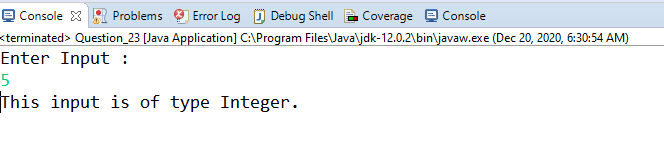
**else** **if** (input.hasNextLine()) System.***out***.println("This input is of type string.");

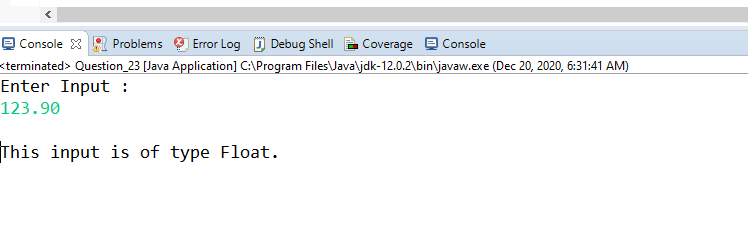
**else** **if** (input.hasNextDouble()) System.***out***.println("This input is of type Double.");

**else** **if** (input.hasNextBoolean()) System.***out***.println("This input is of type Boolean.");

}

}





/\*Q24. Write a program which inputs a positive natural number N and

\* prints the possible consecutive number combinations, which when added give N. INPUT: N = 9 OUTPUT: 4 + 5 2 + 3+ 4

\*/

package com.training.Java\_Hackthon;

import java.util.Scanner;

public class Question\_24 {

static void printSums(int N)

{

int start = 1, end = 1;

int sum = 1;

while (start <= N/2)

{

if (sum < N)

{

end += 1;

sum += end;

}

else if (sum > N)

{

sum -= start;

start += 1;

}

else if (sum == N)

{

for (int i = start;i <= end; ++i)

{

System.out.print(i + " ");

}

System.out.println();

sum -= start;

start += 1;

}

}

}

public static void main (String[] args)

{

Scanner input = new Scanner(System.in);

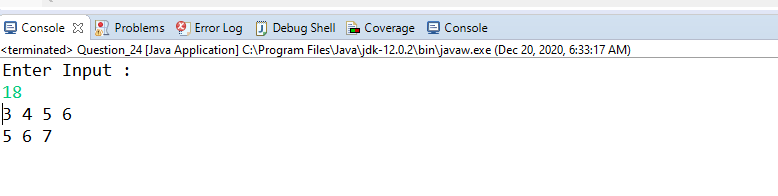
System.out.println("Enter Input :");

int num = input.nextInt();

printSums(num);

}

}



/\*Q25. Write a program for binary search. And 5 i/p has to take from user as binary elements.\*/

**package** com.training.Java\_Hackthon;

**public** **class** Question\_25 {

**public** **static** **int** binarySearch(**int** arr[], **int** first, **int** last, **int** key){

**if** (last>=first){

**int** mid = first + (last - first)/2;

**if** (arr[mid] == key){

**return** mid;

}

**if** (arr[mid] > key){

**return** *binarySearch*(arr, first, mid-1, key);//search in left subarray

}**else**{

**return** *binarySearch*(arr, mid+1, last, key);//search in right subarray

}

}

**return** -1;

}

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

**int** arr[] = {10,20,30,40,50};

**int** key = 30;

**int** last=arr.length-1;

**int** result = *binarySearch*(arr,0,last,key);

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

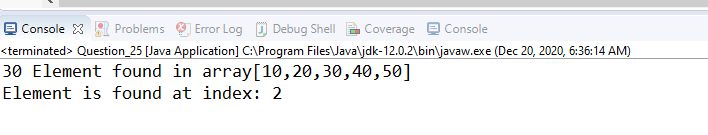
System.***out***.println("Element is not found!");

**else**

System.***out***.println("Element is found at index: "+result);

}

}



/\*Q26. WJP to merge two sorted array.(Do not use third array)

\* array1[10] = 1,2,4,6,9,10 array2[4] = 3, 5,7,8 After merge : array1[10] = 1,2,3,4,5,6,7,8,9,10

\*/

**package** com.training.Java\_Hackthon;

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

**public** **class** Question\_26 {

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

ArrayList<Integer> list1 = **new** ArrayList<Integer>();

{

list1.add(12);

list1.add(16);

list1.add(9);

list1.add(7);

list1.add(4);

}

System.***out***.println(list1);

ArrayList<Integer> list2 = **new** ArrayList<Integer>();

{

list2.add(9);

list2.add(16);

list2.add(9);

list2.add(7);

list2.add(4);

}

System.***out***.println(list2);

*merge*(list1,list2);

}

**public** **static** **void** merge (ArrayList<Integer> list1, ArrayList<Integer> list2)

{

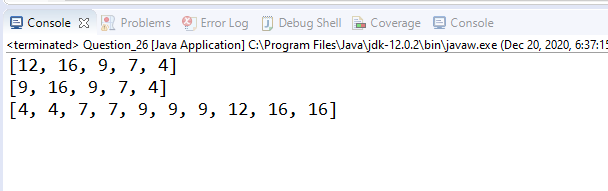
list1.addAll(list2);

Collections.*sort*(list1);

System.***out***.println(list1);

}

}



//WJP to perform ascending order Quick sort

**package** com.training.Java\_Hackthon;

**public** **class** Question\_27 {

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

**int**[] arr={90,23,101,45,65,23,67,89,34,23};

System.***out***.print("The array before Quick sort:");

**for**(**int** i: arr)

{

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

}

*quickSort*(arr, 0, 9);

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

System.***out***.print("The sorted array after Quick sort:");

**for**(**int** i: arr)

{

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

}

}

**public** **static** **int** partition(**int** a[], **int** beg, **int** end)

{

**int** left, right, temp, loc, flag;

loc = left = beg;

right = end;

flag = 0;

**while**(flag != 1)

{

**while**((a[loc] <= a[right]) && (loc!=right))

right--;

**if**(loc==right)

flag =1;

**else** **if**(a[loc]>a[right])

{

temp = a[loc];

a[loc] = a[right];

a[right] = temp;

loc = right;

}

**if**(flag!=1)

{

**while**((a[loc] >= a[left]) && (loc!=left))

left++;

**if**(loc==left)

flag =1;

**else** **if**(a[loc] <a[left])

{

temp = a[loc];

a[loc] = a[left];

a[left] = temp;

loc = left;

}

}

}

**return** loc;

}

**static** **void** quickSort(**int** a[], **int** beg, **int** end)

{

**int** loc;

**if**(beg<end)

{

loc = *partition*(a, beg, end);

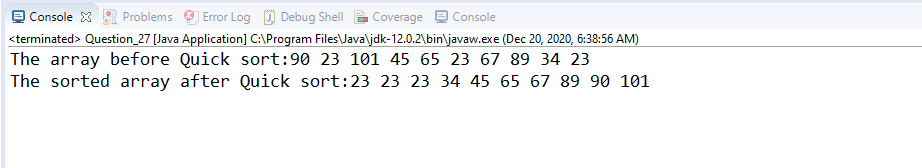
*quickSort*(a, beg, loc-1);

*quickSort*(a, loc+1, end);

}

}

}



**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_28 {

**static** **int** fact(**int** n)

{

**int** result =1;

**while**(n!=1)

{

result = result\*n;

n--;

}

**return** result;

}

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

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

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

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

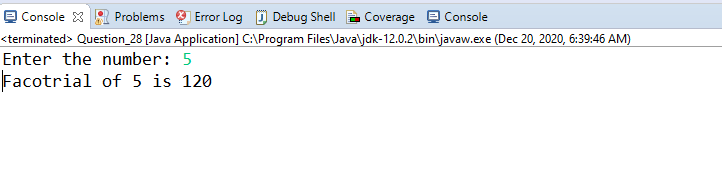
**int** num = scan.nextInt() ;

**int** result = *fact*(num);

System.***out***.println("Facotrial of " + num + " is " + result);

}

}



//. WJP to perform Merge sort using recursion

**package** com.training.Java\_Hackthon;

**public** **class** Question\_29 {

**static** **void** Mergesort(**int** arr[])

{

**int** newarr[] =**new** **int**[arr.length];

*Mergesort*(arr,newarr,0,arr.length-1);

}

**static** **void** Mergesort(**int** arr[],**int** newArr[],**int** lb,**int** ub)

{

**if** (lb<ub)

{

**int** med;

med= (lb+ub)/2;

*Mergesort*(arr,newArr,lb,med);//sort left half

*Mergesort*(arr,newArr,med+1,ub);//sort right half

*merge*(arr,newArr,lb,med,ub);// Merge them

}

}

**static** **void** merge(**int** arr[],**int** helper[],**int** lb,**int** med,**int** ub) {

//copy both halves into helper array

**for** (**int** i=lb; i<=ub;i++)

{

helper[i]= arr[i];

}

**int** helperleft = lb;

**int** helperright = med+1;

**int** current = lb;

/\* Iterate through helper array.compare left and right hand

\*

\*/

**while**(helperleft<=med && helperright<=ub)

{

**if** (helper[helperleft]<=helper[helperright])

{

arr[current]=helper[helperleft];

helperleft++;

}

**else**

{

arr[current]= helper[helperright];

helperright++;

}

current++;

}

//Copy the rest of the left side of the array into th target arrray

**int** remaining = med-helperleft;

**for** (**int** i=0;i<=remaining;i++)

{

arr[current+i]= helper[helperleft+i];

}

}

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

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

**int** Arr[] = {1,2,3,7,5,2};

System.***out***.print("Array before Merge sort :");

**for**(**int** i: Arr)

{

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

}

*Mergesort*(Arr);

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

System.***out***.print("Array after Merge sort : ");

**for**(**int** i : Arr)

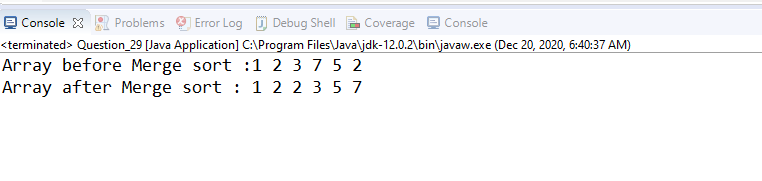
{

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

}

}

}



// Write a function to find out longest palindrome in a given string

**package** com.training.Java\_Hackthon;

**public** **class** Question\_30 {

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

**public** **static** String getLongestPalindrome(**final** String input) {

**int** rightIndex = 0, leftIndex = 0;

String currentPalindrome = "", longestPalindrome = "";

**for** (**int** centerIndex = 1; centerIndex < input.length() - 1; centerIndex++) {

leftIndex = centerIndex - 1; rightIndex = centerIndex + 1;

**while** (leftIndex >= 0 && rightIndex < input.length()) {

**if** (input.charAt(leftIndex) != input.charAt(rightIndex)) {

**break**;

}

currentPalindrome = input.substring(leftIndex, rightIndex + 1);

longestPalindrome = currentPalindrome.length() > longestPalindrome.length() ? currentPalindrome : longestPalindrome;

leftIndex--; rightIndex++;

}

}

**return** longestPalindrome;

}

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

String str = "madammomnoon1000001sdkjchjkd";

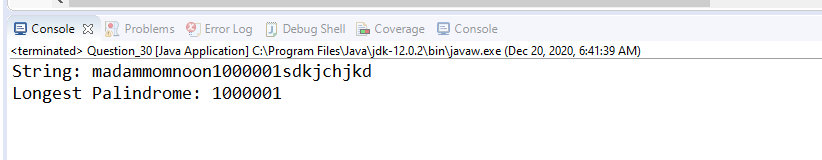
String longestPali = *getLongestPalindrome*(str);

System.***out***.println("String: " + str);

System.***out***.println("Longest Palindrome: " + longestPali);

}

}



/\*Q31. Read a file content and write it to a new file in reverse order.(reverse line 1-10 to line 10-1)\*/

**package** com.training.Java\_Hackthon;

**import** java.io.BufferedReader;

**import** java.io.BufferedWriter;

**import** java.io.FileReader;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** java.util.ArrayList;

**public** **class** Question\_31 {

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

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

String filepath = System.*getProperty*("user.dir")+"/Leena.txt";

// System.out.println(filepath);

ArrayList<String> str1 = **new** ArrayList<String>();

// String filepath="C:\\Users\\sagar\\eclipse-workspace\\Filedemo\\Leena.txt";

str1=*readFile*(filepath);

String wfilepath= System.*getProperty*("user.dir")+"/FiletoWrite.txt";

// String wfilepath ="C:\\Users\\sagar\\eclipse-workspace\\Filedemo\\FiletoWrite.txt";

// System.out.println(wfilepath);

*writeFile*(wfilepath,str1);

// loadingPropertiesFile();

}

**public** **static** ArrayList<String> readFile(String sPath) **throws** IOException

{

String sLine;

FileReader oFR = **new** FileReader(sPath);

BufferedReader oBR = **new** BufferedReader(oFR);

ArrayList<String> str = **new** ArrayList<String>();

**while**((sLine = oBR.readLine()) != **null**)

{

str.add(sLine);

}

**return** str;

}

**public** **static** **void** writeFile(String sPath,ArrayList<String> str) **throws** IOException {

//1. First you will go to location 2. you will open it 3. write it

BufferedWriter oBW = **null**;

**try** {

FileWriter oFW = **new** FileWriter(sPath,**false**);

oBW = **new** BufferedWriter(oFW);

**for**(**int** i=str.size()-1; i>=0;i--)

{

oBW.write(str.get(i));

oBW.newLine();

}

}**catch**(Exception a) {

System.***out***.println("Exception : "+a);

}**finally**

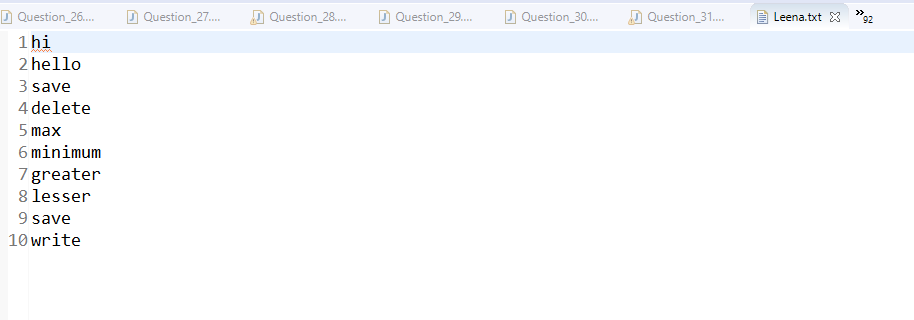
{

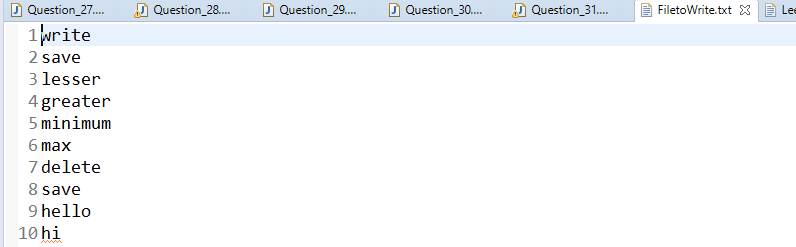
oBW.close();

}

}

}





Q.32You are given two sorted arrays, A and B, and A has a large enough bu#er at the end to hold B.

\* Write a method to merge B into A in sorted order.\*/

package com.training.Java\_Hackthon;

import java.io.\*;

public class Question\_32 {

static int NA =-1;

static void sortedMerge(int a[], int b[], int n, int m)

{

int i = n - 1;

int j = m - 1;

int lastIndex = n + m - 1;

// Merge a and b, starting

// from last element in each

while (j >= 0)

{

/\* End of a is greater than end of b \*/

if (i >= 0 && a[i] > b[j])

{

// Copy Element

a[lastIndex] = a[i];

i--;

} else

{

// Copy Element

a[lastIndex] = b[j];

j--;

}

// Move indices

lastIndex--;

}

}

// Helper function to print the array

static void printArray(int arr[], int n)

{

System.out.println ( "Array A after merging B in sorted order : " ) ;

for (int i = 0; i < n; i++)

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

}

public static void main (String[] args)

{

int a[] = {10, 12, 13, 14, 18, NA, NA, NA, NA, NA};

int n = 5;

int size\_a = 10;

int b[] = {16, 17, 19, 20, 22};

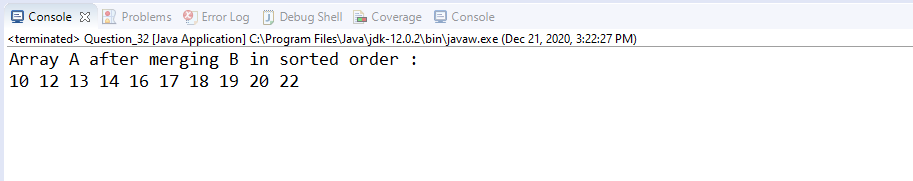
int m = 5;

sortedMerge(a, b, n, m);

printArray(a, size\_a);

}

}



**package** com.training.Java\_Hackthon;

/\*Q33. A circus is designing a tower routine consisting of people standing

\* atop one another’s shoulders. For practical and aesthetic reasons, each person must

\* be both shorter and lighter than the person below him or her. Given the heights and

\* weights of each person in the circus, write a method to compute the largest possible

\* number of people in such a tower. EXAMPLE: Input (ht, wt): (65, 100) (70, 150) (56, 90)

\* (75, 190) (60, 95) (68, 110) Output: The longest tower is length 6 and includes from

\* top to bottom: (56, 90) (60,95) (65,100) (68,110) (70,150) (75,190)

\*/

**public** **class** Question\_33 {

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

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

**double**[][] array= {

{65,100},

{70,150},

{56,90},

{75,190},

{60,95},

{68,110}

};

java.util.Arrays.*sort*(array, **new** java.util.Comparator<**double**[]>()

{

**public** **int** compare(**double**[] a, **double**[] b) {

**return** Double.*compare*(a[0], b[0]);

}

});

System.***out***.println("Circus Tower with heavy weight and tallest at the bottom");

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

{

**for**(**int** j = 0; j<array[0].length; j++)

{

System.***out***.print(array[i][j]+" ");

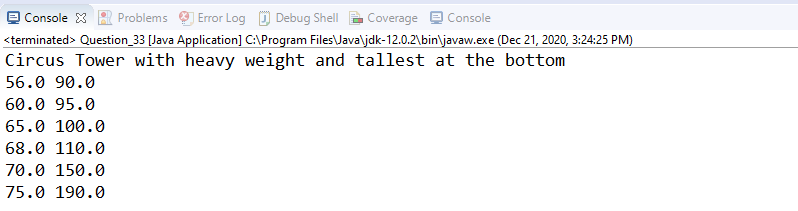
}

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

}

}

}



\*Q34. Write a method to implement \*, - , / operations. You should use only the + operator\*/

**package** com.training.Java\_Hackthon;

**import** java.util.Scanner;

**public** **class** Question\_34 {

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

//Function to flip the sign using only "+"

// operator (It is simple with '\*' allowed.

// We need to do a = (-1)\*a

**static** **int** flipSign(**int** a)

{

**int** neg = 0;

// If sign is + ve turn it -ve

// and vice-versa

**int** tmp = a < 0 ? 1 : -1;

**while** (a != 0)

{

neg += tmp;

a += tmp;

}

**return** neg;

}

// Check if a and b are of different signs

**static** **boolean** areDifferentSign(**int** a, **int** b)

{

**return** ((a < 0 && b > 0) || (a > 0 && b < 0));

}

// Function to subtract two numbers

// by negating b and adding them

**static** **int** sub(**int** a, **int** b)

{

// Negating b

**return** a + *flipSign*(b);

}

// Function to multiply a by b by

// adding a to itself b times

**static** **int** mul(**int** a, **int** b)

{

// because algo is faster if b<a

**if** (a < b)

**return** *mul*(b, a);

// Adding a to itself b times

**int** sum = 0;

**for** (**int** i = Math.*abs*(b); i > 0; i--)

sum += a;

// Check if final sign must be -ve or + ve

**if** (b < 0)

sum = *flipSign*(sum);

**return** sum;

}

// Function to divide a by b by counting

// how many times 'b' can be subtracted

// from 'a' before getting 0

**static** **int** division(**int** a, **int** b)

{

// Raise exception if b is 0

**if** (b == 0)

**throw** **new** ArithmeticException();

**int** quotient = 0, dividend;

// Negating b to subtract from a

**int** divisor = *flipSign*(Math.*abs*(b));

// Subtracting divisor from dividend

**for** (dividend = Math.*abs*(a); dividend >= Math.*abs*(divisor);

dividend += divisor)

quotient++;

// Check if a and b are of similar symbols or not

**if** (*areDifferentSign*(a, b))

quotient = *flipSign*(quotient);

**return** quotient;

}

// Driver code

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

{

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

System.***out***.println("Enter number:");

**int** a = sc.nextInt();

System.***out***.println("Enter number:");

**int** b = sc.nextInt();

System.***out***.println("Subtraction is " + *sub*(a, b));

System.***out***.println("Multiplication is " + *mul*(a, b));

**try**

{

System.***out***.println("Division is " + *division*(a, b));

}

**catch** (ArithmeticException e)

{

System.***out***.println("Exception :- Divide by 0");

}

}

}

