**1.     Write a java programme to sort the integers 8, 4, 3, 5, 6 and the alphabetical string C, O, I, P, U, in ascending order. Show the resulting output.**

**package** lab6;

**import** java.util.Arrays;

**public** **class** SortIntegersAndStrings {

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

// Integer array

Integer[] integers = { 8, 4, 3, 5, 6 };

// String array

String[] strings = { "C", "O", "I", "P", "U" };

// Sort the integer array

Arrays.*sort*(integers);

// Sort the string array

Arrays.*sort*(strings);

// Print the sorted arrays

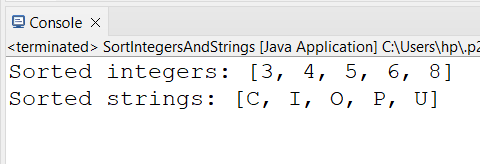
System.***out***.println("Sorted integers: " + Arrays.*toString*(integers));

System.***out***.println("Sorted strings: " + Arrays.*toString*(strings));

}

}

**OutPut:**

****

**2.     Write a Java program to implement the bubble sort algorithm to sort an array of integers in ascending order.**

**package** lab6;

**public** **class** Bubble\_Sorting {

// Method to perform bubble sort on an array

**public** **static** **void** bubbleSort(**int**[] array) {

**int** n = array.length;

**boolean** swapped;

// Loop over each element in the array

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

swapped = **false**;

// Inner loop for comparing adjacent elements

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

// Swap if the element is greater than the next element

**if** (array[j] > array[j + 1]) {

**int** temp = array[j];

array[j] = array[j + 1];

array[j + 1] = temp;

swapped = **true**;

}

}

// If no two elements were swapped in the inner loop, the array is sorted

**if** (!swapped) {

**break**;

}

}

}

// Main method to test the bubble sort algorithm

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

**int**[] array = {64, 34, 25, 12, 22, 11, 90};

System.***out***.println("Original array:");

**for** (**int** num : array) {

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

}

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

*bubbleSort*(array);

System.***out***.println("Sorted array:");

**for** (**int** num : array) {

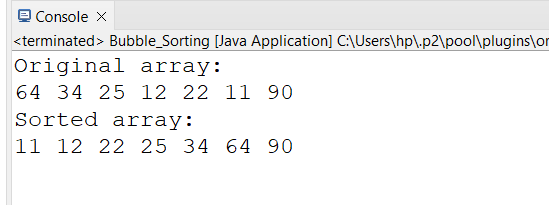
System.***out***.print(num + " ");

}

}

}

**OutPut:**

****

**3.     Write a program to input an array 10 elements and print the cube of prime numbers in it.**

**package** lab6;

**import** java.util.Scanner;

**public** **class** PrimeCubes {

// Method to check if a number is prime

**public** **static** **boolean** isPrime(**int** num) {

**if** (num <= 1) {

**return** **false**;

}

**for** (**int** i = 2; i <= Math.*sqrt*(num); i++) {

**if** (num % i == 0) {

**return** **false**;

}

}

**return** **true**;

}

// Method to calculate the cube of a number

**public** **static** **int** cube(**int** num) {

**return** num \* num \* num;

}

// Main method

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

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

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

// Input 10 elements into the array

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

**for** (**int** i = 0; i < 10; i++) {

array[i] = scanner.nextInt();

}

// Print the cube of prime numbers in the array

System.***out***.println("Cubes of prime numbers in the array:");

**for** (**int** num : array) {

**if** (*isPrime*(num)) {

System.***out***.println("Cube of " + num + " is " + *cube*(num));

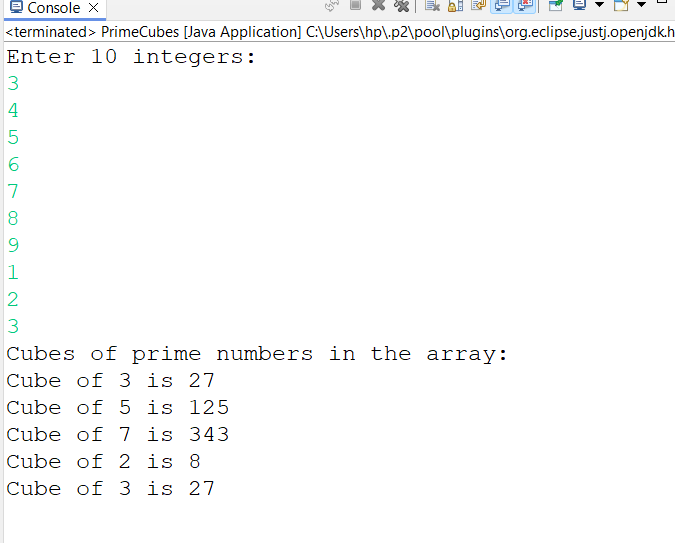
}

}

}

}

**Output:**



**4.     Write a java program to implement integer wrapper class methods. (Any 5 methods)**

**package** lab6;

**public** **class** IntegerWrapperClassMethods {

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

// Using the Integer.valueOf() method

Integer intObj = Integer.*valueOf*(123);

System.***out***.println("Integer.valueOf(123): " + intObj);

// Using the Integer.parseInt() method

**int** intValue = Integer.*parseInt*("456");

System.***out***.println("Integer.parseInt(\"456\"): " + intValue);

// Using the Integer.toString() method

String intString = Integer.*toString*(789);

System.***out***.println("Integer.toString(789): " + intString);

// Using the Integer.compare() method

**int** compareResult = Integer.*compare*(10, 20);

System.***out***.println("Integer.compare(10, 20): " + compareResult); // Output will be negative because 10 < 20

// Using the Integer.equals() method

Integer intObj1 = Integer.*valueOf*(100);

Integer intObj2 = Integer.*valueOf*(100);

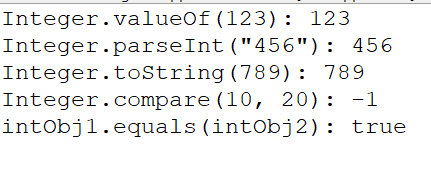
**boolean** isEqual = intObj1.equals(intObj2);

System.***out***.println("intObj1.equals(intObj2): " + isEqual); // Output will be true because 100 equals 100

}

}

**Output:**

****

**5.     Write a java program to implement double wrapper class methods. (Any 5 methods)**

**package** lab6;

**public** **class** DoubleWrapperClassMethods {

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

// Using the Double.valueOf() method

Double doubleObj = Double.*valueOf*(123.45);

System.***out***.println("Double.valueOf(123.45): " + doubleObj);

// Using the Double.parseDouble() method

**double** doubleValue = Double.*parseDouble*("456.78");

System.***out***.println("Double.parseDouble(\"456.78\"): " + doubleValue);

// Using the Double.toString() method

String doubleString = Double.*toString*(789.01);

System.***out***.println("Double.toString(789.01): " + doubleString);

// Using the Double.compare() method

**int** compareResult = Double.*compare*(10.5, 20.5);

System.***out***.println("Double.compare(10.5, 20.5): " + compareResult); // Output will be negative because 10.5 <

// 20.5

// Using the Double.isNaN() method

Double NaNValue = Double.***NaN***;

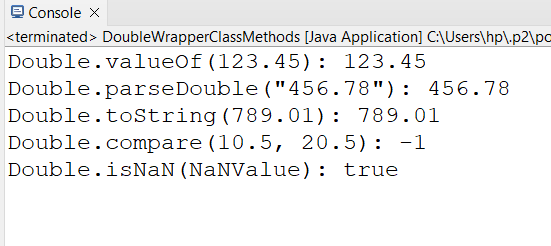
**boolean** isNaN = Double.*isNaN*(NaNValue);

System.***out***.println("Double.isNaN(NaNValue): " + isNaN); // Output will be true because NaNValue is Not-a-Number

}

}

**OutPut:**

****

**6.     Write a java program to implement float wrapper class methods. (Any 5 methods)**

**package** lab6;

**public** **class** FloatWrapperClassMethods {

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

// Using the Float.valueOf() method

Float floatObj = Float.*valueOf*(123.45f);

System.***out***.println("Float.valueOf(123.45f): " + floatObj);

// Using the Float.parseFloat() method

**float** floatValue = Float.*parseFloat*("456.78");

System.***out***.println("Float.parseFloat(\"456.78\"): " + floatValue);

// Using the Float.toString() method

String floatString = Float.*toString*(789.01f);

System.***out***.println("Float.toString(789.01f): " + floatString);

// Using the Float.compare() method

**int** compareResult = Float.*compare*(10.5f, 20.5f);

System.***out***.println("Float.compare(10.5f, 20.5f): " + compareResult); // Output will be negative because 10.5 <

// 20.5

// Using the Float.isNaN() method

Float NaNValue = Float.***NaN***;

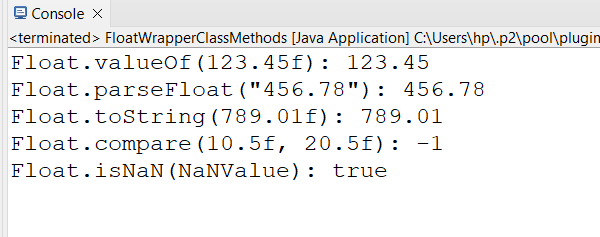
**boolean** isNaN = Float.*isNaN*(NaNValue);

System.***out***.println("Float.isNaN(NaNValue): " + isNaN); // Output will be true because NaNValue is Not-a-Number

}

}

**OutPut:**

****

**7.     Write a Java program to validate email addresses using regular expressions. The email should have the format username@domain.com where username and domain can contain alphanumeric characters, dots, and hyphens.**

**package** lab6;

**import** java.util.Scanner;

**import** java.util.regex.Matcher;

**import** java.util.regex.Pattern;

**public** **class** EmailValidator {

// Method to validate email addresses

**public** **static** **boolean** isValidEmail(String email) {

// Regular expression for validating an email address

String emailRegex = "^[a-zA-Z0-9.\_-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,6}$";

// Compile the regular expression into a pattern

Pattern pattern = Pattern.*compile*(emailRegex);

// Match the input email address against the pattern

Matcher matcher = pattern.matcher(email);

// Return true if the email matches the pattern, false otherwise

**return** matcher.matches();

}

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

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

// Prompt the user to enter an email address

System.***out***.println("Enter an email address to validate:");

String email = s.nextLine();

**if** (*isValidEmail*(email)) {

System.***out***.println(email + " is a valid email address");

} **else** {

System.***out***.println(email + " is not a valid email address.");

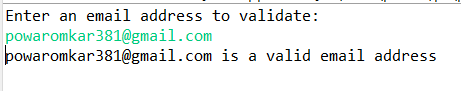
}

}

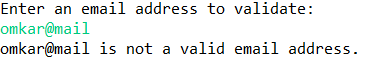
}

**OutPut:**

**Right output.**

****

**Wrong Out put:**

****

**8. Create a Java program to validate phone numbers. The format should be (xxx) xxx-xxxx where x is a digit**.

**package** lab6;

**import** java.util.Scanner;

**import** java.util.regex.Matcher;

**import** java.util.regex.Pattern;

**public** **class** PhoneNumberValidator {

// Method to validate phone numbers

**public** **static** **boolean** isValidPhoneNumber(String phoneNumber) {

// Regular expression for validating a phone number

String phoneNumberRegex = "^\\(\\d{3}\\) \\d{3}-\\d{4}$";

// Compile the regular expression into a pattern

Pattern pattern = Pattern.*compile*(phoneNumberRegex);

// Match the input phone number against the pattern

Matcher matcher = pattern.matcher(phoneNumber);

// Return true if the phone number matches the pattern, false otherwise

**return** matcher.matches();

}

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

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

// Prompt the user to enter a phone number

System.***out***.println("Enter a phone number to validate (format: (xxx) xxx-xxxx):");

String phoneNumber = scanner.nextLine();

// Validate the phone number

**if** (*isValidPhoneNumber*(phoneNumber)) {

System.***out***.println(phoneNumber + " is a valid phone number.");

} **else** {

System.***out***.println(phoneNumber + " is not a valid phone number.");

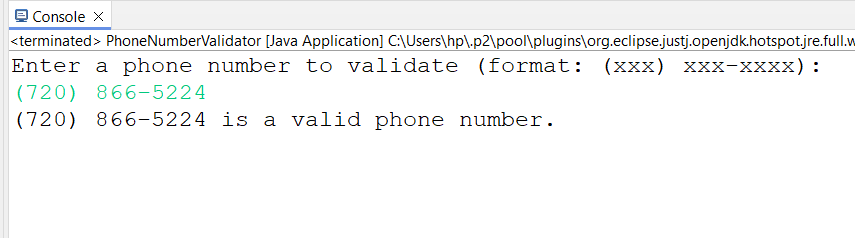
}

}

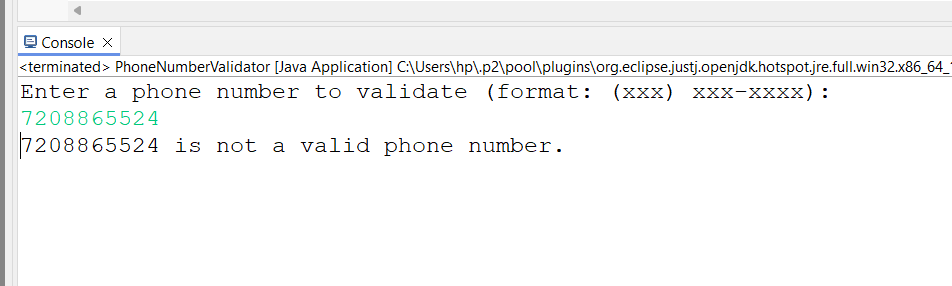
}

**OutPut:**

**Right OutPut:**

****

**Wrong Output:**

****