**LAB 6**

**1. Write the program 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 Session;

import java.util.Arrays;

public class SortingIntAlpha {

public static void main(String[] args) {

int[] arr1= new int[]{8,4,3,5,6};

System.*out*.println("Before Sorting : -" + Arrays.*toString*(arr1));

Arrays.*sort*(arr1);

System.*out*.println("After Sorting : -" + Arrays.*toString*(arr1));

String[] arr2= new String[]{"C", "O", "I", "P", "U"};

System.*out*.println("Before Sorting : -" + Arrays.*toString*(arr2));

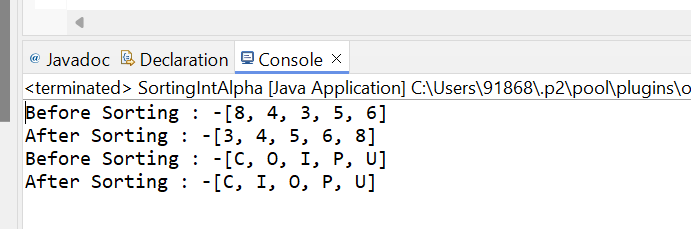
Arrays.*sort*(arr2);

System.*out*.println("After Sorting : -" + Arrays.*toString*(arr2));

}

}

**Output:-**

****

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

package Session;

import java.util.Arrays;

public class ArrayBubbleSort {

public static void main(String[] args) {

int[] arr= new int[] {2,6,4,3,7,10,8,0};

System.*out*.println("Before Sorting : -" + Arrays.*toString*(arr));

boolean swap;

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

{

swap=false;

for(int j=0;j<arr.length-1;j++)

{

if(arr[j]>arr[j+1])

{

int temp=arr[j];

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

arr[j+1]=temp;

swap=true;

}

}

if(!swap)

{

break;

}

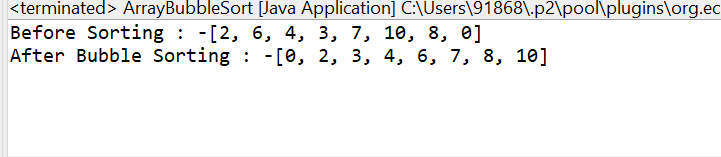
}

System.*out*.println("After Bubble Sorting : -" + Arrays.*toString*(arr));

}

}

**Output:-**

****

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

package Session;

import java.util.Scanner;

import java.util.Arrays;

public class ArrayCubePrime {

public static boolean Prime(int num) {

if (num <= 1) {

return false;

}

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

if (num % i == 0) {

return false;

}

}

return true;

}

public static void main(String[] args) {

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

int[] arr = new int[10];

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

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

arr[i] = s.nextInt();

}

System.*out*.println("Array: " + Arrays.*toString*(arr));

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

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

if (*Prime*(arr[i])) {

int cube = arr[i] \* arr[i] \* arr[i];

System.*out*.println("Cube of " + arr[i] + " is " + cube);

}

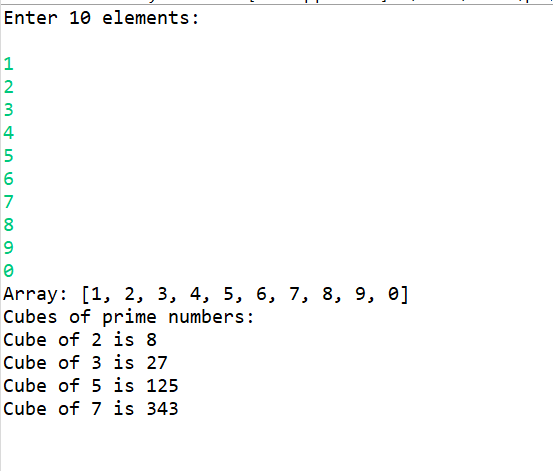
}

s.close();

}

}

**Output:-**

****

**4. Write a java program to implement integer wrapper class methods.(any 3 methods)**

package Session;

public class IntegerWrapperClassMethod {

public static void main(String[] args) {

String number= "123";

int num = Integer.*parseInt*(number);

System.*out*.println("Parsed integer from string '" + number + "': " + num);

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

System.*out*.println("Integer object from string '" + number + "': " + intObj);

int Value = 42;

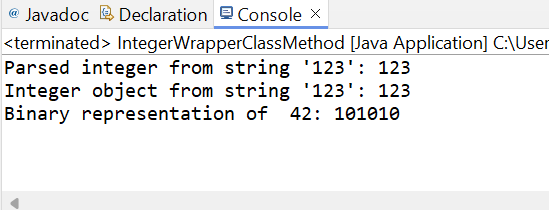
String binary = Integer.*toBinaryString*(Value);

System.*out*.println("Binary representation of " + Value + ": " + binary);

}

}

**Output:-**

****

**5. Write a java program to implement double wrapper class methods.(any 3 methods)**

package Session;

public class IntegerDoubleWrapperClass {

public static void main(String[] args) {

Double d1 = 123.50;

Double d2 = 456.78;

double Value = d1.doubleValue();

System.*out*.println("doubleValue(): " + Value);

int comparison = Double.*compare*(d1, d2);

System.*out*.println("compare(double x, double y): " + (comparison < 0 ? "123.50 is less than 456.78" : "123.50 is greater than or equal to 456.78"));

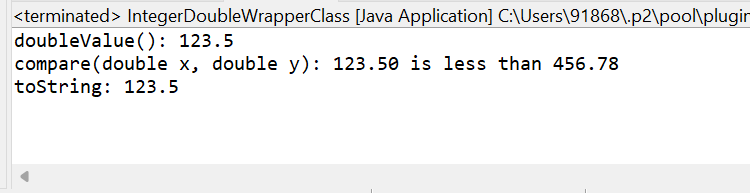
String String = Double.*toString*(d1);

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

}

}

**Output:-**

****

**6. Write a java program to implement float wrapper class methods.(any 3 methods)**

package Session;

public class FloatWrapperClassMethod {

public static void main(String[] args) {

Float f1 = 123.45f;

Float f2 = 456.78f;

String fla = Float.*toString*(f1);

System.*out*.println("toString(float f): " + fla);

System.*out*.println("MIN\_VALUE: " + Float.*MIN\_VALUE*);

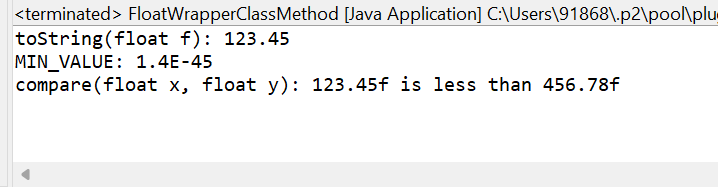
int comparison = Float.*compare*(f1, f2);

System.*out*.println("compare(float x, float y): " + (comparison < 0 ? "123.45f is less than 456.78f" : "123.45f is greater than or equal to 456.78f"));

}

}

**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 Session;

import java.util.Scanner;

import java.util.regex.Pattern;

import java.util.regex.Matcher;

public class EmailRegularExpression {

private static final String *EMAIL\_REGEX* = "^[a-zA-Z0-9.\_-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}$";

public static void main(String[] args)

{

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

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

String email = s.nextLine();

if (*isValidEmail*(email)) {

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

} else {

System.*out*.println("The email address is invalid.");

}

s.close();

}

public static boolean isValidEmail(String email) {

Pattern pattern = Pattern.*compile*(*EMAIL\_REGEX*);

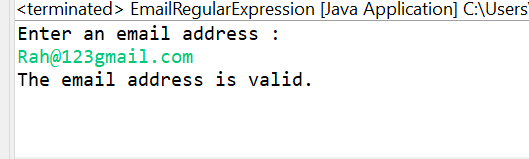
Matcher matcher = pattern.matcher(email);

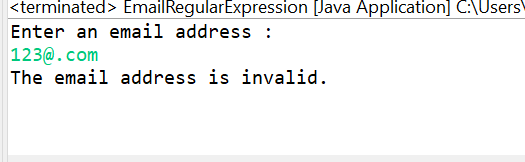
return matcher.matches();

}

}

**Output:-**

****

****

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

package Session;

import java.util.Scanner;

import java.util.regex.Pattern;

import java.util.regex.Matcher;

public class PhoneNumberRegularExpression {

private static final String *Phone\_REGEX* = "^\\(\\d{3}\\) \\d{3}-\\d{4}$";

public static void main(String[] args)

{

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

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

String phone = s.nextLine();

if (*isValidPhone*(phone)) {

System.*out*.println("The phone number is valid.");

} else {

System.*out*.println("The phone number is invalid.");

}

s.close();

}

public static boolean isValidPhone(String phone) {

Pattern pattern = Pattern.*compile*(*Phone\_REGEX*);

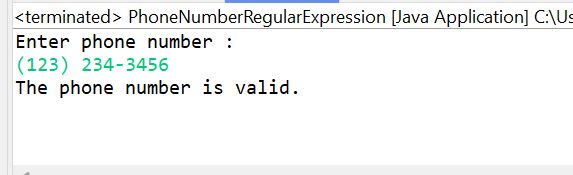
Matcher matcher = pattern.matcher(phone);

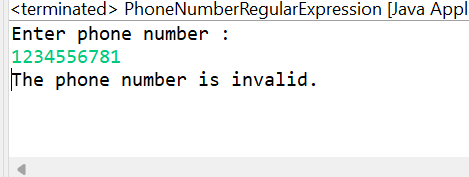
return matcher.matches();

}

}

**Output:-**

****

****