Division

Saturday, August 13, 2022 9:57 PM

An integer can be converted into a floating point number by placing a type-casting operation `(double)` before it:

```
int first = 3;
int second = 2;

double result1 = (double) first / second;
System.out.println(result1); // prints 1.5

double result2 = first / (double) second;
System.out.println(result2); // prints 1.5

double result3 = (double) (first / second);
System.out.println(result3); // prints 1.0
```

Sample output

```
1.5
1.5
1.0
```

The last example produces an incorrectly rounded result, because the integer division is executed before the type casting.

If the result of a division is assigned to an integer-type variable, the result is automatically an integer.

```
int integer = 3.0 / 2;
System.out.println(integer);
```

Sample output

```
1
```

The next example prints "1.5"; the dividend is converted into a floating-point number by multiplying it with a floating-point number prior to executing the division.

```
int dividend = 3;
int divisor = 2;

double result = 1.0 * dividend / divisor;
System.out.println(result);
```

Sample output

```
1.5
```

I/O

Thursday, August 18, 2022 4:14 PM

Note: If you use the *nextLine()* method immediately following the *nextInt()* method, recall that *nextInt()* reads integer tokens; because of this, the last newline character for that line of integer input is still queued in the input buffer and the next *nextLine()* will be reading the remainder of the integer line (which is empty).

From <https://www.hackerrank.com/challenges/java-stdin-stdout/problem?isFullScreen=true&h_r=next-challenge&h_v=zen&h_r=next-challenge&h_v=zen&h_r=next-challenge&h_v=zen>

```
System.out.print("Insert a number: ");
int number = input.nextInt();
input.nextLine(); // This line you have to add (It consumes the \n character)
System.out.print("Text1: ");
String text1 = input.nextLine();
```

next() can read the input only till the space. It can't read two words separated by space. Also, next() places the cursor in the same line after reading the input. nextLine() reads input including space between the words (that is, it reads till the end of line \n).

From <https://www.google.com/search?q=java+next+vs+nextline&oq=java+next+vs+&aqs=chrome.0.0i512j69i57j0i22i30l8.4446j0j4&sourceid=chrome&ie=UTF-8>

Array

Thursday, August 18, 2022 4:56 PM

// Simultaneously declare and initialize an array of strings
String[] fruits={"apple","banana","pear","kiwi"};

From <https://www.caveofprogramming.com/java-video/java-for-complete-beginners-video-part-11-string-arrays.html>

```
// 2D array (grid or table)
int[][] grid = {
    {3, 5, 2343},
    {2, 4},
    {1, 2, 3, 4}
};
```

```
// The last array index is optional.
String[][] words = new String[2][];

// Each sub-array is null.
System.out.println(words[0]);

// We can create the subarrays 'manually'.
words[0] = new String[3];
```

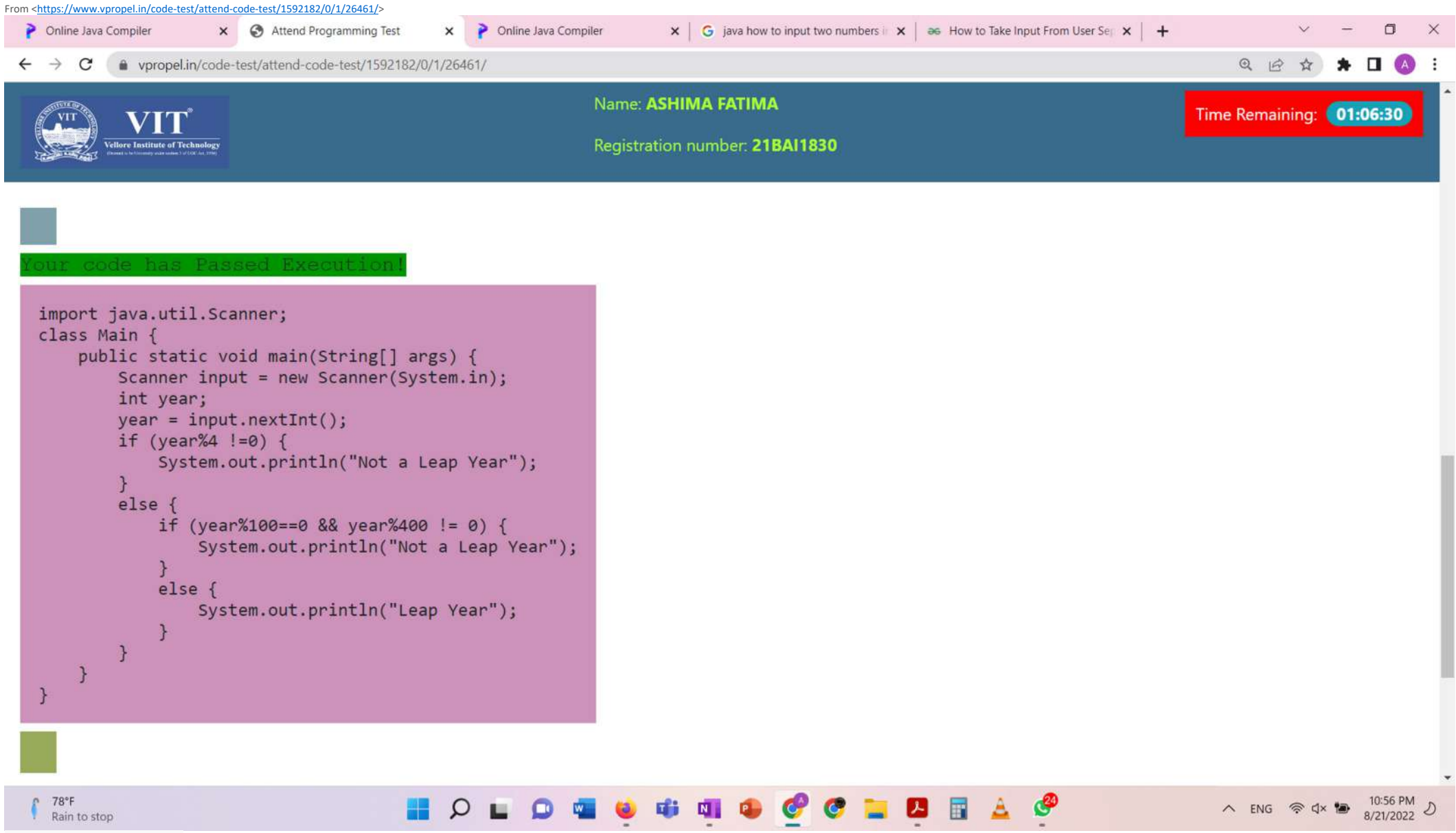
IPS3 Command Structures

Sunday, August 21, 2022 10:55 PM

Leap Year using Java

Write a Java Program to check whether the given year is leap year or not.

From <https://www.vpropel.in/code-test/attend-code-test/1592182/0/1/26461/>



Factors of number using Java

Write a Java program to print the factors of a given number.

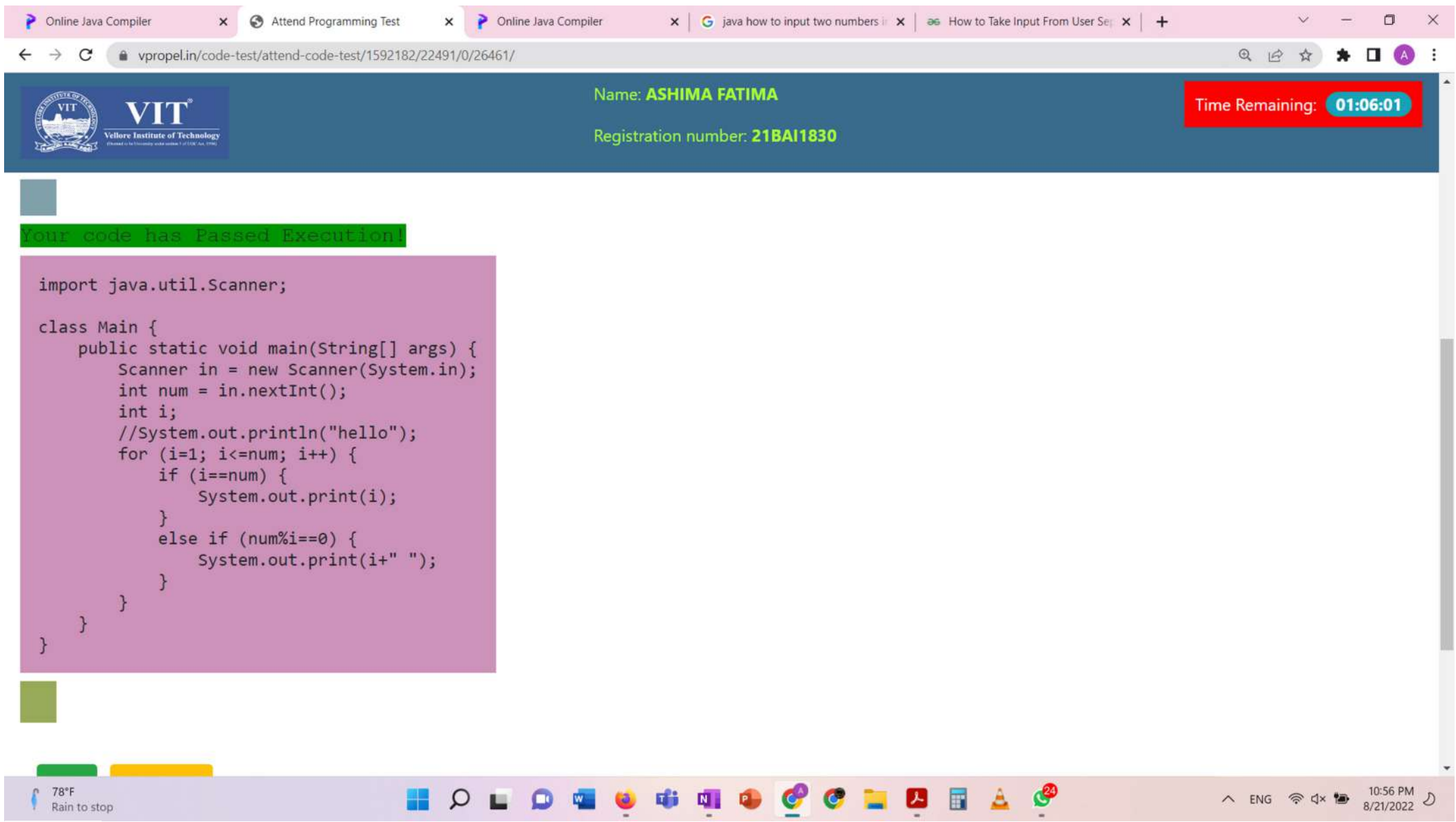
Sample Input:

144

Sample Output:

1 2 3 4 6 8 9 12 16 18 24 36 48 72 144

From <https://www.vpropel.in/code-test/attend-code-test/1592182/22491/0/26461/>



Square Integers

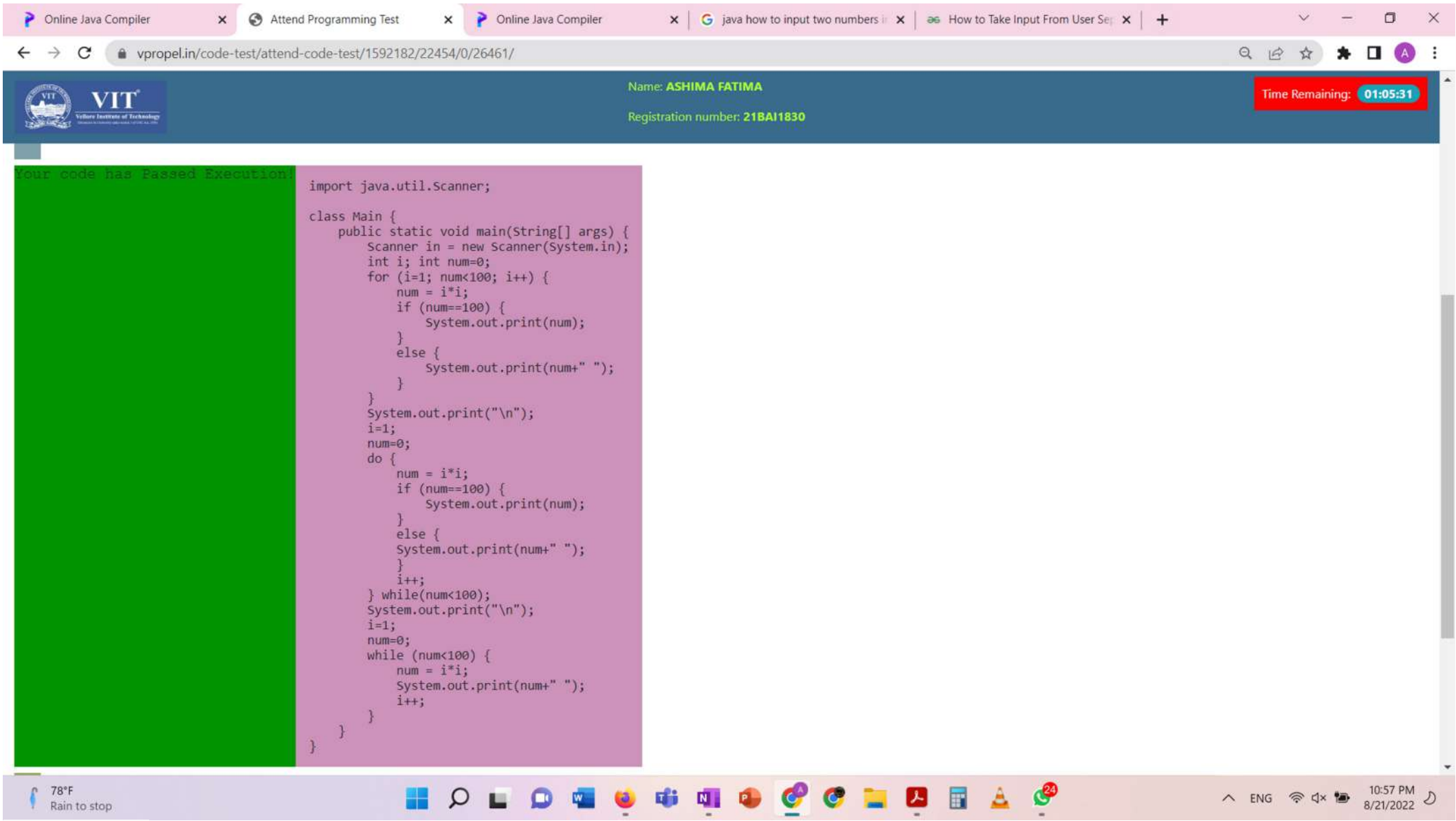
Print the square of integers from 1 to 100 using three different loop statements (For, while and Do while)

1 4 9 16 25 36 49 64 81 100

1 4 9 16 25 36 49 64 81 100

1 4 9 16 25 36 49 64 81 100

From <https://www.vpropel.in/code-test/attend-code-test/1592182/22454/0/26461/>



Surprise Chocolate Plan

Kids Play school is planning for a slow walk race for 'n' children. The kid who touches the finishing line first gets 0 chocolate. The second kid gets 1 chocolate. The numer of chocolates the third kid gets is the sum of chocolates given to first and second kid. Fourth kid gets sum of chocolates of second and third kids and it goes on. Print the number of chocolates got by each kid.

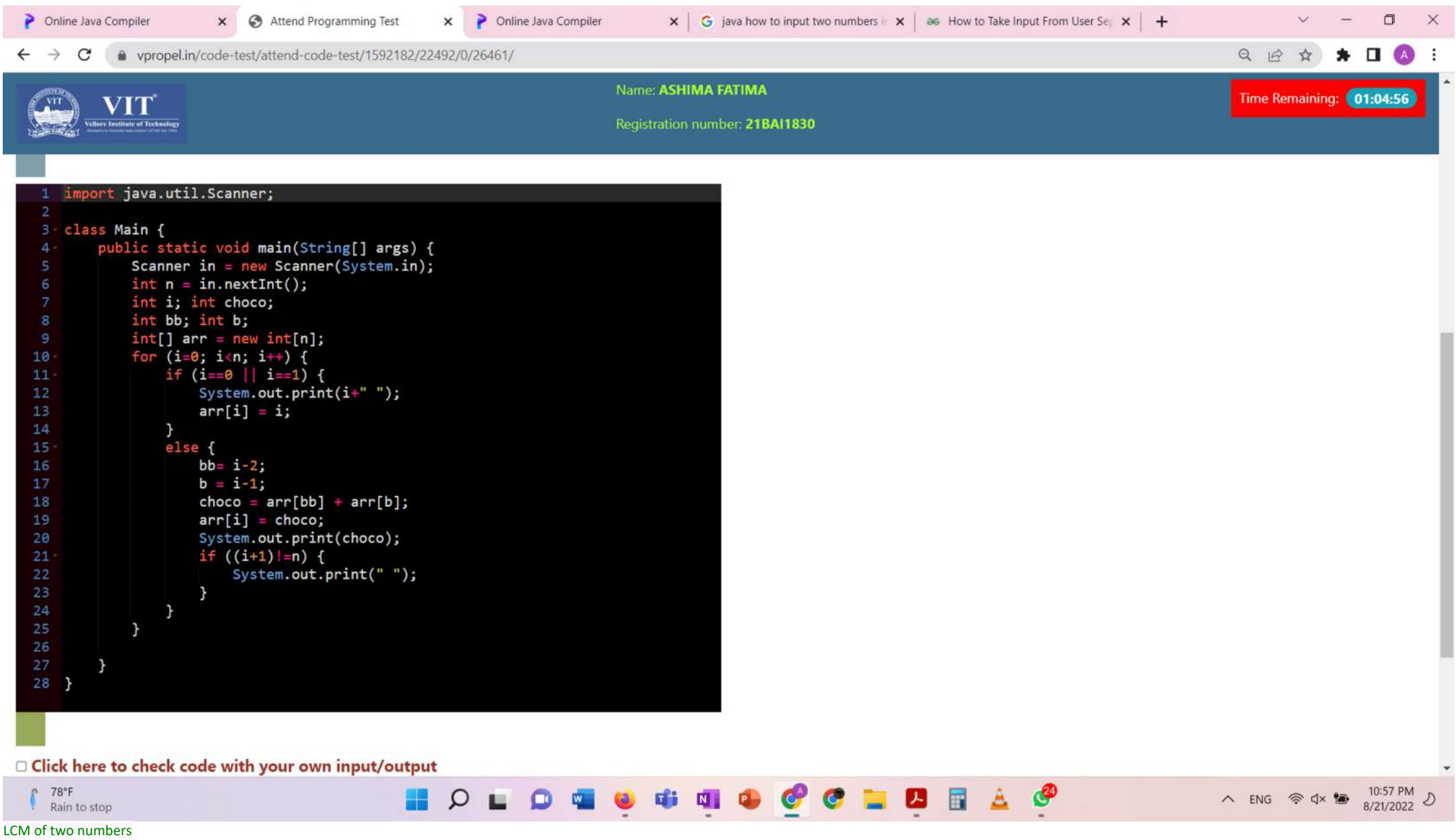
Sample Input:

5 - Number of Children - Integer

Sample Output:

0 1 1 2 3

From <https://www.vpropel.in/code-test/attend-code-test/1592182/22492/0/26461/>



LCM of two numbers

Write a Java program to find the LCM of two numbers.

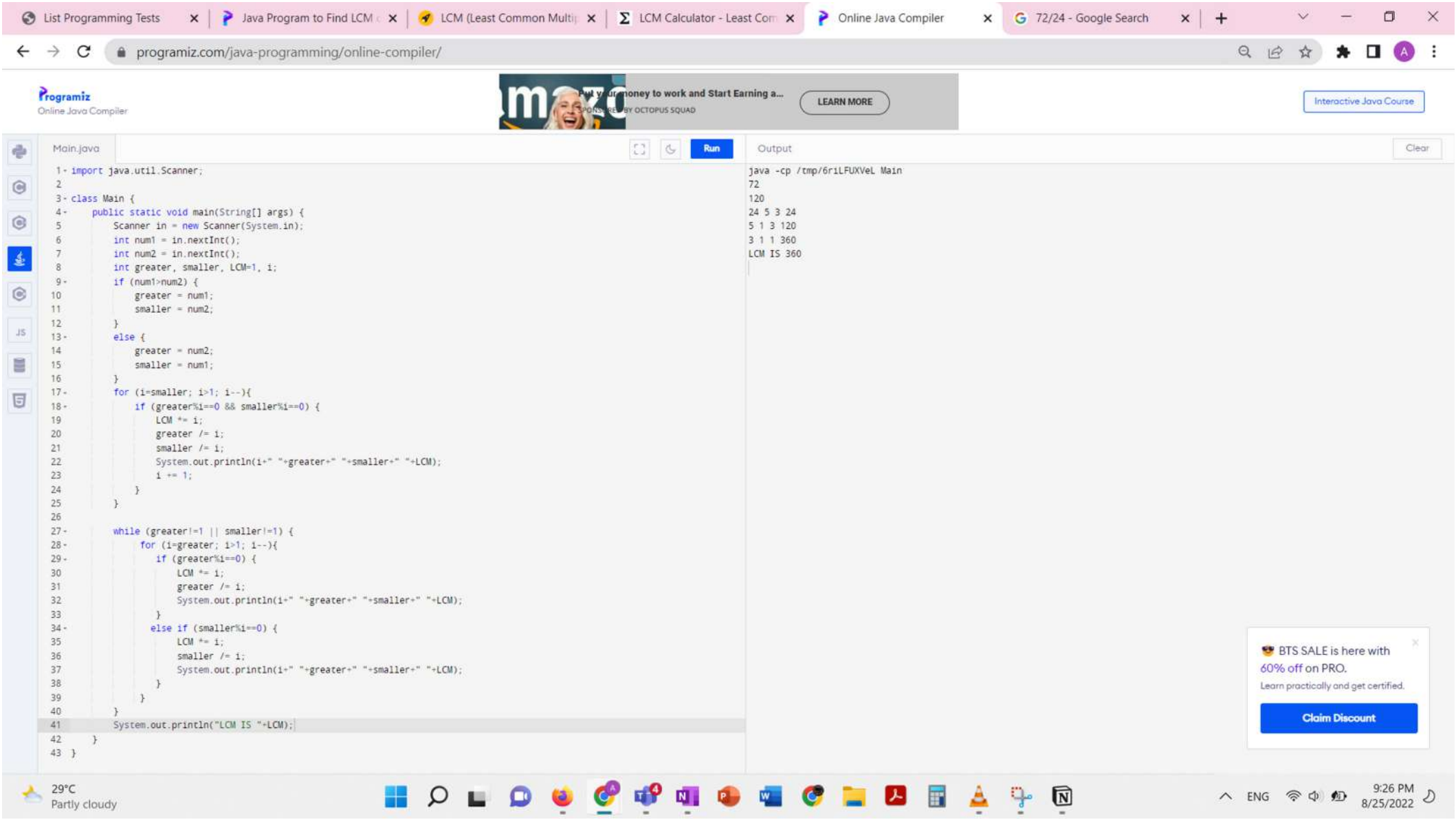
Sample Input:

72 120

Sample Output

360

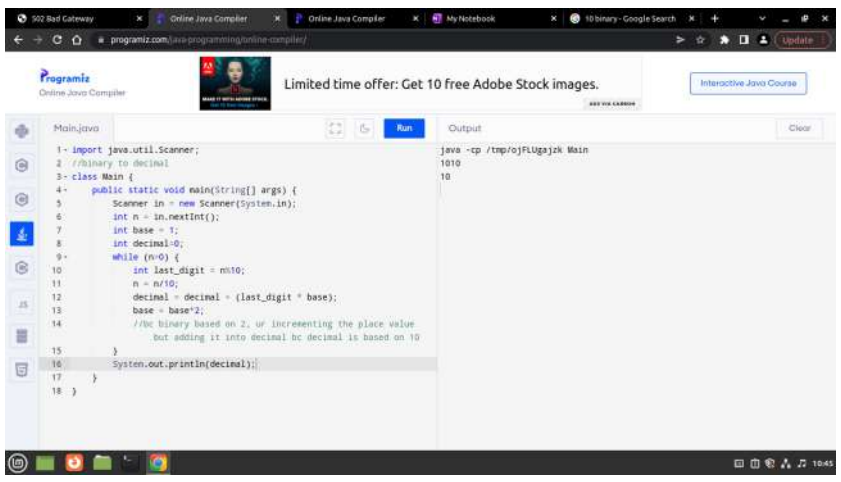
From <https://www.vpropel.in/code-test/attend-code-test/1592182/22494/0/26461/>


List Programming Tests
Java Program to Find LCM
LCM (Least Common Multi
LCM Calculator - Least Com
Online Java Compiler
72/24 - Google Search
programiz.com/java-programming/online-compiler/
Programiz
Online Java Compiler
Put your money to work and Start Earning a...
SPONSORED BY OCTOPUS SQUAD
LEARN MORE
Interactive Java Course
Main.java
Run
Output
Clear
1 import java.util.Scanner;
2
3 class Main {
4 public static void main(String[] args) {
5 Scanner in = new Scanner(System.in);
6 int num1 = in.nextInt();
7 int num2 = in.nextInt();
8 int greater, smaller, LCM=1, i;
9 if (num1>num2) {
10 greater = num1;
11 smaller = num2;
12 }
13 else {
14 greater = num2;
15 smaller = num1;
16 }
17 for (i=smaller; i>1; i--){
18 if (greater%i==0 && smaller%i==0) {
19 LCM *= i;
20 greater /= i;
21 smaller /= i;
22 System.out.println(i+" "+greater+" "+smaller+" "+LCM);
23 i += 1;
24 }
25 }
26
27 while (greater!=1 || smaller!=1) {
28 for (i=greater; i>1; i--){
29 if (greater%i==0) {
30 LCM *= i;
31 greater /= i;
32 System.out.println(i+" "+greater+" "+smaller+" "+LCM);
33 }
34 else if (smaller%i==0) {
35 LCM *= i;
36 smaller /= i;
37 System.out.println(i+" "+greater+" "+smaller+" "+LCM);
38 }
39 }
40 }
41 System.out.println("LCM IS "+LCM);
42 }
43 }
java -cp /tmp/6riLFUXVeL Main
72
120
24 5 3 24
5 1 3 120
3 1 1 360
LCM IS 360
BTS SALE is here with
60% off on PRO.
Learn practically and get certified.
Claim Discount
29°C
Partly cloudy
ENG
9:26 PM
8/25/2022

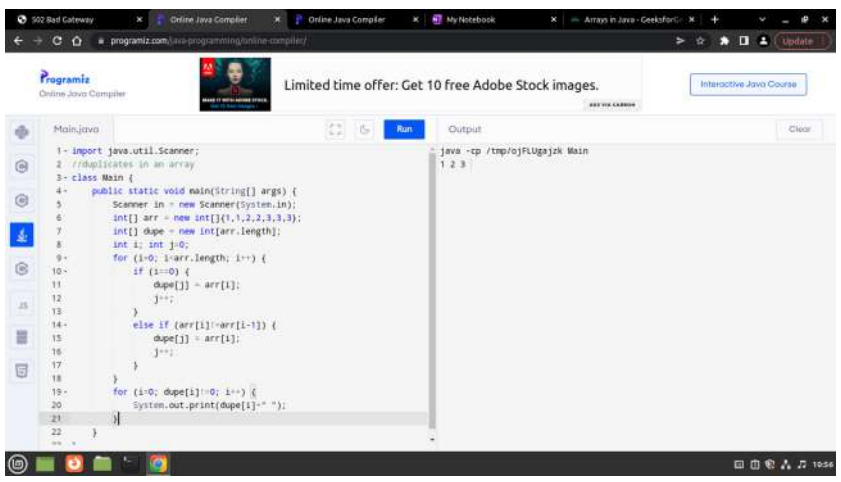
27/8 Int array problems

Monday, August 22, 2022 10:22 AM

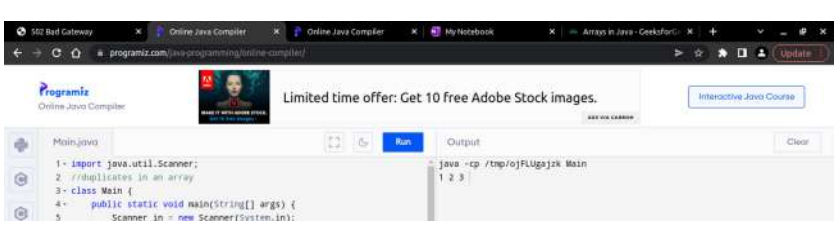

```
import java.util.Scanner;
//decimal to binary
class Main {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      int n = in.nextInt();
      int i;
      int[] arr = new int[50];
      for (i=0; n>0; i++) {
         arr[i] = n%2;
         n = n/2;
      }
      for (int j= i-1; j>=0; j--) {
         System.out.print(arr[j]);
      }
   }
}
```



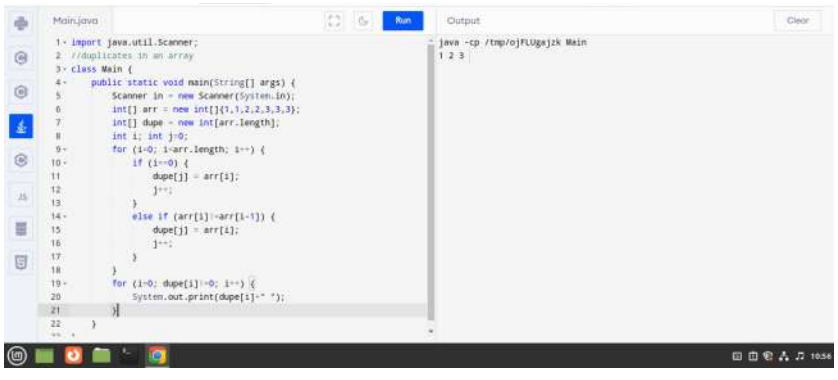

```
import java.util.Scanner;
//binary to decimal
class Main {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      int n = in.nextInt();
      int base = 1;
      int decimal=0;
      while (n>0) {
         int last_digit = n%10;
         n = n/10;
         decimal = decimal + (last_digit * base);
         base = base*2;
         //bc binary based on 2, ur incrementing the place value but adding it into decimal bc decimal is
based on 10
      }
      System.out.println(decimal);
   }
}
```




```
import java.util.Scanner;
//duplicates in an array
class Main {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      int[] arr = new int[]{1,1,2,2,3,3,3};
      int[] dupe = new int[arr.length];
      int i; int j=0;
      for (i=0; i<arr.length; i++) {
         if (i==0) {
            dupe[j] = arr[i];
            j++;
         }
         else if (arr[i]!=arr[i-1]) {
            dupe[j] = arr[i];
            j++;
         }
      }
      for (i=0; dupe[i]!=0; i++) {
         System.out.print(dupe[i]+" ");
      }
   }
}
```

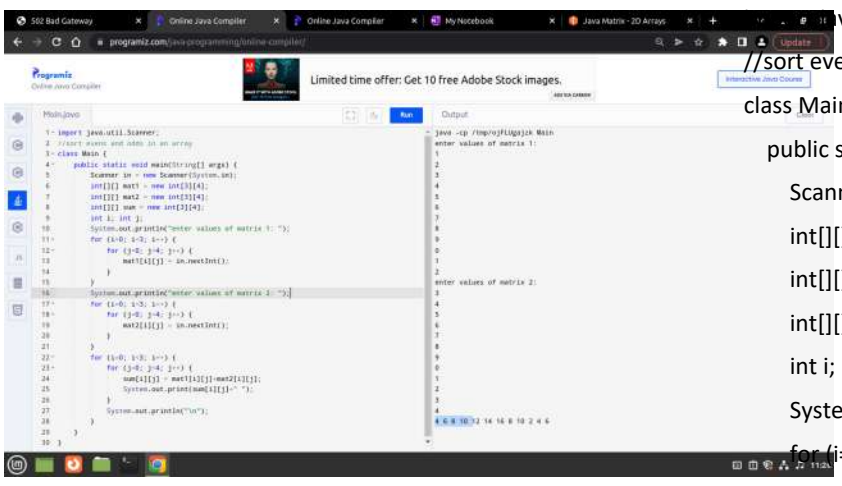



```
import java.util.Scanner;
//sort evens and odds in an array
class Main {
   public static void main(String[] args) {
```




```
class Main {
  public static void main(String[] args) {
    Scanner in = new Scanner(System.in);
    System.out.println("Enter number of values:");
    int n = in.nextInt();
    int i;
    int[] arr = new int[n];
    int[] even = new int[n];
    int[] odd = new int[n];
    for (i=0; i<n; i++) {
      arr[i] = in.nextInt();
      even[i] = -1;
      odd[i] = -1;
    }
    int j=0; int k =0;
    for (i=0; i<n; i++) {
      if (arr[i]%2==0) {
        even[j] = arr[i];
        j++;
      }
      else {
        odd[k] = arr[i];
        k++;
      }
    }
    System.out.println("Even values are: ");
    for (i=0; even[i]!=-1; i++) {
      System.out.println(even[i]+" ");
    }
    System.out.println("Odd values are: ");
    for (i=0; odd[i]!=-1; i++) {
      System.out.println(odd[i]+" ");
    }
  }
}
```

wrapper class in java
Matrix addition/ subtraction of 2 3x4 matrix
A 2d array in C is diff from in java
    Can leave 2nd dimension blank and it isn't fixed in java
    Therefore, number of columns for each row can be diff.
Multidimensional arrays in java called arrays of arrays




```
va.util.Scanner;
//sort evens and odds in an array
class Main {
public static void main(String[] args) {
  Scanner in = new Scanner(System.in);
  int[][] mat1 = new int[3][4];
  int[][] mat2 = new int[3][4];
  int[][] sum = new int[3][4];
  int i; int j;
  System.out.println("enter values of matrix 1: ");
  for (i=0; i<3; i++) {
    for (j=0; j<4; j++) {
      mat1[i][j] = in.nextInt();
    }
  }
  System.out.println("enter values of matrix 2: ");
  for (i=0; i<3; i++) {
    for (j=0; j<4; j++) {
      mat2[i][j] = in.nextInt();
    }
  }
  for (i=0; i<3; i++) {
    for (j=0; j<4; j++) {
      sum[i][j] = mat1[i][j]+mat2[i][j];
      System.out.print(sum[i][j]+" ");
```

```
            }
            System.out.println("\n");
        }
    }
}
```

IPS 4

Tuesday, August 23, 2022 11:50 PM

Multiplication table

Read an integer and print its multiplication table.

Sample I/O:

2

2*1=2

2*2=4

2*3=6

.

.

.

2*10=20

Font Size: 18

Language: Select a Theme

Editor Theme

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        for (int i=1; i<11; i++) {
            System.out.println(n+"*"+i+"="+(n*i));
        }
    }
}
```

Remove Duplicated from a sorted array

Write a Java program to remove duplicate elements from a sorted array.

Sample I/O

7

1 1 2 2 2 3 3

1 2 3

Font Size

18

Language

Editor Theme

Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
//duplicates in an array
class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        int[] arr = new int[n];
        int i; int j=0;
        for (i=0; i<n; i++) {
            arr[i] = in.nextInt();
        }
        int[] dupe = new int[arr.length];
        for (i=0; i<arr.length; i++) {
            if (i==0) {
                dupe[j] = arr[i];
                j++;
            }
            else if (arr[i]!=arr[i-1]) {
                dupe[j] = arr[i];
                j++;
            }
        }
        for (i=0; dupe[i]!=0; i++) {
            System.out.print(dupe[i]+" ");
        }
    }
}
```

Sort the given array

Write a Java program to sort the given array.

Sample I/O:

6

12 2 3 5 78 21

2 3 5 12 21 78

Font Size

18

Language

Editor Theme

Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        int i; int j=0; int min; int temp;
        int[] arr = new int[n];
        for (i=0; i<n; i++) {
            arr[i] = in.nextInt();
        }
        for (i=0; i<n-1; i++) {
            min = arr[i];
            for (j=i+1; j<n; j++) {
                if (arr[j]<min) {
                    temp = arr[i];
                    arr[i] = arr[j];
                    arr[j] = temp;
                    break;
                }
            }
        }
        for (i=0; i<n; i++) {
            System.out.print(arr[i]+" ");
        }
    }
}
```

Binary to Decimal Conversion

Write a Java Program to convert the given binary number into decimal.

Sample Input:

1010

Output:

10

Font Size

18

Language

Editor Theme

Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
//binary to decimal
class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        int base = 1;
        int decimal=0;
        while (n>0) {
            int last_digit = n%10;
            n = n/10;
            decimal = decimal + (last_digit * base);
            base = base*2;
            //bc binary based on 2, ur incrementing the place value but adding it into decimal bc decimal is based on 10
        }
        System.out.println(decimal);
    }
}
```

Decimal to Binary Conversion

Write a java program to convert the given decimal number to binary.

Sample Input:

7

Sample Output:

111

Font Size

18

Language

Editor Theme

Select a Theme

Your code has Passed Execution!

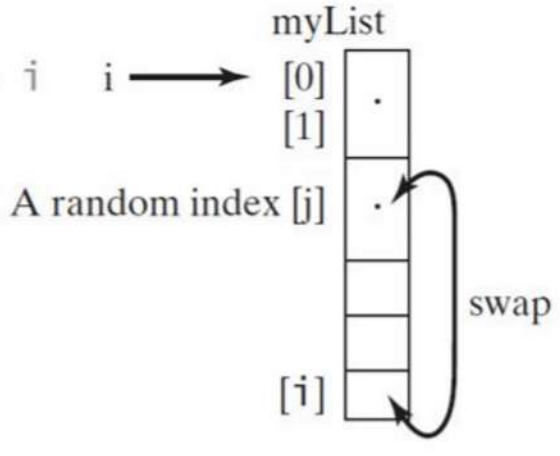
```
import java.util.Scanner;
//decimal to binary
class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        int i;
        int[] arr = new int[50];
        for (i=0; n>0; i++) {
            arr[i] = n%2;
            n = n/2;
        }
        for (int j= i-1; j>=0; j--) {
            System.out.print(arr[j]);
        }
    }
}
```

Arrays

Saturday, August 27, 2022 9:02 PM

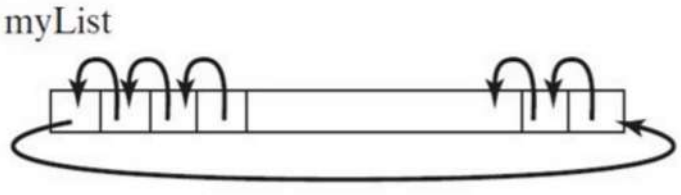
Random shuffling

```
for (int i = myList.length - 1; i > 0; i--) {
  // Generate an index j randomly with 0 <= j <= i
  int j = (int)(Math.random()
    * (i + 1));

  // Swap myList[i] with myList[j]
  double temp = myList[i];
  myList[i] = myList[j];
  myList[j] = temp;
}
```



Shifting Elements

```
double temp = myList[0]; // Retain the first element

// Shift elements left
for (int i = 1; i < myList.length; i++) {
  myList[i - 1] = myList[i];
}

// Move the first element to fill in the last position
myList[myList.length - 1] = temp;
```



IPS 5 Single D Arrays

Saturday, August 27, 2022 9:04 PM

Merge Sorted Arrays

Given two sorted arrays of different size, merge these arrays into a single sorted array.

Sample I/O:

5

12 18 26 27 32

8

8 10 15 28 36 45 78 96

Output:

8 10 12 15 18 26 27 28 32 36 45 78 96

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n1; int n2; int i;
        n1 = in.nextInt();
        int[] arr1 = new int[n1];
        for (i=0; i<n1; i++) {
            arr1[i] = in.nextInt();
        }
        n2 = in.nextInt();
        int[] arr2 = new int[n2];
        for (i=0; i<n2; i++) {
            arr2[i] = in.nextInt();
        }
        int n3 = n1+n2;
        int[] merged = new int[n3];
        for (i=0; i<n3; i++) {
            if (i<n2) {
                merged[i] = arr2[i];
            }
            else {
                merged[i] = arr1[i-n2];
            }
        }
        int min; int j;
        for (i=0; i<(n3-1); i++) {
            min = merged[i];
            for (j=i+1; j<n3; j++) {
                if (merged[j] < min) {
                    min = merged[j];
                    merged[j] = merged[i];
                    merged[i] = min;
                }
            }
        }
        for (i=0; i<n3; i++) {
            System.out.print(merged[i]+" ");
```

Insert Element in an Array

Write a Java program to insert a new element in the particular position. (Create array of size - n+1)

Sample I/O:

6 - n

12 23 25 28 45 68 - elements

3 - position

15 - new element

Output:

12 23 15 25 28 45 68

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        int[] arr = new int[n+1];
        int i;
        for (i=0; i<n; i++) {
            arr[i] = in.nextInt();
        }
        int pos = in.nextInt();
        int ele = in.nextInt();
        i=n;
        while(i>(pos-1)) {
            arr[i] = arr[i-1];
            i--;
        }
        arr[pos-1] = ele;
        for (i=0; i<(n+1); i++) {
            System.out.print(arr[i]+" ");
        }
    }
}
```

Shift Zeroes to beginning

Write a java program to push the zeroes to the beginning of the array.

Sample I/O:

11

12 25 0 0 2 0 6 8 0 18 0

0 0 0 0 0 12 25 2 6 8 18

Font Size: 18

Language

Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        int[] arr = new int[n];
        int i; int count=0 ;
        for (i=0; i<n; i++) {
            arr[i] = in.nextInt();
            if (arr[i]==0) {
                count += 1;
            }
        }
        int[] shift = new int[n];
        for (i=0; i<count; i++) {
            shift[i] = 0;
        }
        for (i=0; i<n; i++) {
        }
        int j=count;
        for (i=0; i<n; i++) {
            if (arr[i] != 0) {
                shift[j] = arr[i];
                j++;
            }
        }
        for (i=0; i<n; i++) {
            System.out.print(shift[i]+" ");
        }
    }
}
```

IPS 6 2D Array

Saturday, August 27, 2022 10:52 PM

Matrix Addition

Write a java program to add two matrices

From <https://www.vpropel.in/code-test/attend-code-test/1605774/22754/0/26664/>

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int m = in.nextInt();
        int n = in.nextInt();
        int i; int j;
        int[][] mat1 = new int[m][n];
        int[][] mat2 = new int[m][n];
        for (i=0; i<m; i++) {
            for (j=0; j<m; j++) {
                mat1[i][j] = in.nextInt();
            }
        }
        for (i=0; i<m; i++) {
            for (j=0; j<m; j++) {
                mat2[i][j] = in.nextInt();
            }
        }
        int[][] sum = new int[m][n];
        for (i=0; i<m; i++) {
            for (j=0; j<m; j++) {
                sum[i][j] = mat1[i][j]+mat2[i][j];
            }
        }
        for (i=0; i<m; i++) {
            for (j=0; j<m; j++) {
                if (j==(m-1)) {
                  System.out.print(sum[i][j]);
                }
                else {
                    System.out.print(sum[i][j]+" ");
                }
            }
            System.out.println();
        }
    }
}
```

Matrix Subtraction

Write a java program to print difference of two m X n matrices

From <https://www.vpropel.in/code-test/attend-code-test/1605774/22755/0/26664/>

```java
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int m = in.nextInt();
        int n = in.nextInt();
        int[][] mat1 = new int[m][n];
        int[][] mat2 = new int[m][n];
        int i; int j;
        for (i=0; i<m; i++) {
            for (j=0; j<n; j++) {
                mat1[i][j] = in.nextInt();
            }
        }
        for (i=0; i<m; i++) {
            for (j=0; j<n; j++) {
                mat2[i][j] = in.nextInt();
            }
        }
        int[][] diff = new int[m][n];
        for (i=0; i<m; i++) {
            for (j=0; j<n; j++) {
                diff[i][j] = mat1[i][j]-mat2[i][j];
            }
        }
        for (i=0; i<m; i++) {
            for (j=0; j<n; j++) {
                if (j==n-1) {
                    System.out.print(diff[i][j]);
                }
                else {
                    System.out.print(diff[i][j]+" ");
                }
            }
            System.out.println();
        }
    }
}
```

Matrix Multiplication

Input
m n
A matrix of m X n size
p q
A matrix of p X q size
Output:
Product matrix of m X p size
If n and p are not equal, print matrix multiplication not possible

From <https://www.vpropel.in/code-test/attend-code-test/1605774/22756/0/26664/>

```
class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int m1 = in.nextInt();
        int n1 = in.nextInt();
        int i; int j;
        int[][] mat1 = new int[m1][n1];
        for (i=0; i<m1; i++) {
            for (j=0; j<n1; j++) {
                mat1[i][j] = in.nextInt();
            }
        }
        int m2 = in.nextInt();
        int n2 = in.nextInt();
        int[][] mat2 = new int[m2][n2];
        for (i=0; i<m2; i++) {
            for (j=0; j<n2; j++) {
                mat2[i][j] = in.nextInt();
            }
        }
        if (m1==n2) {
            int[][] product = new int[m1][n2];
            //matrix multiplication has rows of 1st matrix and colums of 2nd matrix

            for (i=0; i<m1; i++) {
                for (j=0; j<n2; j++) {
                    for (int k=0; k<n1; k++) {
                        //i and j go thru product matrixes but bc m1 = n2, need to go thru for every column in n1 too so use k
                        // matrix multiplication is [m][n]*[n][m] and then their sum
                        product[i][j] += mat1[i][k] * mat2[k][i];
                    }
                }
            }
            for (i=0; i<m1; i++) {
                for (j=0; j<n2; j++) {
                    if (j==n2-1) {
                        System.out.print(product[i][j]);
                    }
                    else {
                        System.out.print(product[i][j] + " ");
                    }
                }
                System.out.println();
            }
        }
        else {
            System.out.println("Multiplication not possible");
        }
    }
}
```

3/9 String Objects

Saturday, September 3, 2022 9:56 AM

Interned String:

```
import java.util.Scanner;

class String {
public static void main(String[] args) {
Scanner in = new Scanner(System.in);
String s1 = "Bunny"; // stored in local memory; called interned strings
String s2 = new String("Bunny"); // stored in heap memory;
//if (s1.equals(s2)) { // takes the content to check
      if (s1 == s2) {
System.out.println("Equal");
}
else {
System.out.println("Not Equal");
// the strings are not equal
}
}
}
```

Char b[] = toCharArray(s1); - string converted to sequence of characters and stored in the array

IPS Q4; s1.CompareTo(s2) method to compare strings

Anagram = array.sort() and then compare arrays

IPS 7

Saturday, September 3, 2022 11:14 AM

Sorting characters in a String

Read a string, sort the characters present in the string in alphabetical order.

Sample I/O:

apple

aelpp

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        char arr[] = s1.toCharArray();
        Arrays.sort(arr);
        for (int i=0; i<arr.length; i++) {
            System.out.print(arr[i]);
        }
    }
}
```

Anagram or not

Check whether two strings are anagrams to each other using a java program.

Two *strings* are said to be *anagram* if we can form one *string* by arranging the characters of another *string*.

Example:

silent and listen are anagrams

triangle and integral are anagrams

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        String s2 = in.nextLine();
        char arr1[] = s1.toCharArray();
        char arr2[] = s2.toCharArray();
        Arrays.sort(arr1);
        Arrays.sort(arr2);
        int i; int ana=1;
        if (arr1.length != arr2.length) {
            System.out.println("Two strings are not Anagrams");
        }
        else {
            for (i=0; i<arr1.length; i++) {
                if (arr1[i] != arr2[i]) {
                    System.out.println("Two strings are not Anagrams");
                    ana =0;
                    break;
                }
            }
            if (ana==1) {
                System.out.println("Two strings are Anagrams");
            }

        }
    }
}
```

Frequency of Characters in a String

Read a string, print the number of times each character is appearing in the string using Java.

Sample I/O:

intellectual ability

i: 3

n: 1

t: 3

e: 2

l: 4

c: 1

u: 1

a: 2

b: 1

y: 1

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        char arr[] = s1.toCharArray();
        int count[] = new int[arr.length];
        int i; int j;
        for (i=0; i<arr.length; i++) {
            if (arr[i] != '*' && arr[i] != ' ') {
                count[i] += 1;
            for (j=(i+1); j<arr.length; j++) {
                if (arr[i]==arr[j]) {
                    arr[j] = '*';
                    count[i] +=1;
                }
            }
            }
        }
        for (i=0; i<arr.length; i++) {
            if (arr[i] != '*' && arr[i] != ' ') {
                System.out.println(arr[i]+": "+count[i]);
            }
        }
    }
}
```

counting number of vowels, consonants, spaaces and special characters

Sample I/O:

India is my country!

Output:

Vowels: 6

Consonants:10

Special Character: 1

Spaces: 3

```
class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        char arr[] = s1.toCharArray();
        char vowel[] = new char[] {'A', 'E', 'I', 'O', 'U','a','e','i','o','u'};
        char spc[] = new char[] {'~', '!', '@', '#', '$', '%', '^', '&', '*', '(', ')','+'};
        int v_count=0; int c_count=0; int space=0; int spc_count=0;
        for (int i=0; i<arr.length; i++) {
            int consta=0; int j;
            if (arr[i] == ' ') {
                space++;
            }
            else {
                for (j=0; j<vowel.length; j++) {
                    if (arr[i] == vowel[j]) {
                        v_count++;
                        consta = 1;
                    }
                }
                for (j=0; j<spc.length; j++) {
                    if (arr[i] == spc[j]) {
                        spc_count++;
                        consta = 1;
                    }
                }
                if (consta==0) {
                        c_count++;
                    }
            }
        }
        System.out.println("Vowels: "+v_count);
        System.out.println("Consonants: "+c_count);
        System.out.println("Special Character: "+spc_count);
        System.out.println("Spaces: "+space);
    }
}
```

5/9 OOP concepts

Monday, September 5, 2022 10:03 AM

Object – represents an entity in the real world that can be distincly identified (e.g. book)

- State of an object (a.k.a properties/attributes)
- An object has a state and a behaviour
- The state defines an object and behaviour is what the object does

Public, private, protected

- Main function should be put inside a separate public class w name same as file name

Public vs default

- If not public then belongs to default

Class, child class

- Classes are constructs that define objects of the same type
- Class provides special type of methods known as constructors; invoked to construct objects from the class
- When child class called, the call goes to the child class from which it goes to the parent class
- If no default constrcutor is created and an object is created, the complier automatically create defalut constrcutor w empty body
- Destructor deallocates memory allocated to constructor
- In java deallocation (garbage allocation) done automatically by jvm (?), no explicit destructors in java
- Do not specify data type for return in constructor

Every object has a property called data member

Package

The place where you define methods matters for security

Data abstraction

There are default values for variables inside a class

- Java assigns no default value to a local variable inside a method

To split a STRING - Str.split(" ")
To replace a word – str.replaceALL(" "," ");

IPS 8

Monday, September 5, 2022 11:03 AM

```
String s1="java string split method by javatpoint";
String[] words=s1.split("\\s");//splits the string based on whitespace
//using java foreach loop to print elements of string array
for(String w:words){
System.out.println(w);
}
```

To check whether words in the given string is palindrome or not

Sample I/O:

madam is teaching ada language

madam

ada

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        String[] words = s1.split("\\s");
        int i; int j;

        for (i=0; i<words.length; i++) {
           int same =0;
           //loop for one word alone
           char[] letter = words[i].toCharArray();
           char[] r_letter = new char[letter.length];
           int k=0;
           //make an array of letters in reverse
           for (j=letter.length-1; j>=0; j--) {
               r_letter[k]=letter[j];
               k++;
           }

           //now have 2 arrays, compare the arrays
           for (j=0; j<letter.length; j++) {
               if (r_letter[j] != letter[j]) {
                   same = 1;
                   break;
               }
           }
           if (same==0) {
               System.out.println(words[i]);
           }
        }
    }
}
```

```
1  /*Sample I/O:
2  madam is teaching ada language
3  madam
4  ada*/
5
6  import java.util.*;
7  public class Main
8  {
9      public static void main(String ... args)
10     {
11         Scanner sob=new Scanner(System.in);
12         String s1=sob.nextLine();
13         String a[]=s1.split(" ");
14         for(String s:a)
15         {
16             s=s.toLowerCase();
17             String r="";
18             for(int i=s.length()-1;i>=0;i--)
19             {
20                 r=r+s.charAt(i);
21             }
22             if(r.equals(s))
23             {
24                 System.out.println(s);
25             }
26
27         }
28     }
29 }
```

Remove word from the sentence

Sample I/O:

The VIT Quick VIT Brown Fox VIT jumps VIT over VIT the mountain.

The Quick Brown Fox jumps over the mountain.

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        String split[] = s1.split("VIT ");
        for (int i=0; i<split.length; i++) {
            System.out.print(split[i]);
        }
    }
}
```

```java
import java.util.Scanner;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = "The VIT Quick VIT Brown Fox VIT jumps VIT over VIT the
            mountain.";
        s1 = s1.replace("VIT ","");
        System.out.println(s1);
    }
}
```

```
java -cp /tmp/zCET9VirHN Main
The Quick Brown Fox jumps over the mountain.
```

Move capitals to end

Sample I/O:

InDiAnGOVernmenT

ninernmenIDAGOVT

Font Size: 18

Language

Editor Theme: Select a Theme

Your code has Passed Execution!

```java
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        char c1[] = s1.toCharArray();
        int i; int n = c1.length;
        for (i=0; i<n; i++) {
            if (Character.isLowerCase(c1[i])) {
                System.out.print(c1[i]);
            }
        }
        for (i=0; i<n; i++) {
            if (Character.isUpperCase(c1[i])) {
                System.out.print(c1[i]);
            }
        }
    }
}
```

```java
/*Sample I/O:
InDiAnGOVernmenT
ninernmenIDAGOVT*/

import java.util.*;
public class Main
{
    public static void main(String...args)
    {
        Scanner sob=new Scanner(System.in);
        String s=sob.next();
        String small="",big="";
        char c[]=s.toCharArray();
        for(int i=0;i<s.length();i++)
        {
            if(c[i]>=65 && c[i]<=90)
            {
                big=big+c[i];
            }
            else
            {
                small=small+c[i];
            }
        }
        System.out.print(small+big);
    }
}
```

- Character.isUpperCase()
- Character.isLowerCase()

Toggle String

Sample I/O:

Queen ViCToRiA

qUEEN vIctOrIa

Font Size: 18

Language

Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        char c1[] = s1.toCharArray();
        String k;
        for (int i=0; i<c1.length; i++) {
            k = Character.toString(c1[i]);
            if (Character.isUpperCase(c1[i])) {
                k= k.toLowerCase();
                System.out.print(k);
            }
            else {
                k= k.toUpperCase();
                System.out.print(k);
            }
        }
    }
}
```

The default value of a char data type **'\u0000'**. The character values are enclosed with a single quote. Its default size is 2 bytes.

https://www.javatpoint.com › character-array-in-java

Character Array in Java - Javatpoint

Java String **toUpperCase() Method**

The toUpperCase() method converts a string to upper case letters. Note: The toLowerCase() method converts a string to lower case letters.

Convert char to String Java

CharToStringJava.java

```
package com.journaldev.string;

public class CharToStringJava {

    public static void main(String[] args) {

        // char to string
        char c = 'a';
        String str = String.valueOf(c);

        // using Character class
        str = Character.toString(c);

        // another way
        str = new Character(c).toString();
        // string concatenation - worst performance
        str = "" + c;

        // char array to string
        char[] ca = { 'a', 'b', 'c' };
        str = String.valueOf(ca);
        // another way
        str = new String(ca);

    }
```

Splitting words from a sentence

(There may be multiple spaces in a word)

Sample I/O:

India is my country

India

is

my

country

Font Size: 18

Language

Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        String[] s2 = s1.split("\\s+");
        for (String w:s2) {
            if (w!= " ") {
                System.out.println(w);
            }
        }
    }
}
```

Signature

There are two signature for split() method in java string.

> **public** String split(String regex)
> and,
> **public** String split(String regex, **int** limit)

Parameter

regex : regular expression to be applied on string.

limit : limit for the number of strings in array. If it is zero, it will returns all the strings matching regex.

You can use `Quantifiers` to specify the number of spaces you want to split on: -

```
    `+` - Represents 1 or more
    `*` - Represents 0 or more
    `?` - Represents 0 or 1
`{n,m}` - Represents n to m
```

So, `\\s+` will split your string on `one or more` spaces

```
String[] words = yourString.split("\\s+");
```

Also, if you want to specify some specific numbers you can give your range between `{}`:

```
yourString.split("\\s{3,6}"); // Split String on 3 to 6 spaces
```

Palindrome Check

Sample I/O:

Madam

Palindrome

India

Not Palindrome

Font Size: 18

Language

Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        char[] letter = s1.toCharArray();
        char[] r_letter = new char[letter.length];
        int i; int j=0;
        int same=0;
        for (i=letter.length-1; i>-1; i--) {
            r_letter[j] = letter[i];
            j++;
        }
        for (i=0; i<letter.length; i++) {
            if (r_letter[i]!=letter[i]) {
                same = 1;
                break;
            }
        }
        if (same==0) {
            System.out.println("Palindrome");
        }
        else {
            System.out.println("Not Palindrome");
        }
    }
}
```

Sorting set of Strings

Read n - number of strings, set of 'n' strings. Display the sorted list of strings using Java.

Sample I/O:

5

India

america

japan

mexico

switzerland

Output:

america

india

japan

mexico

switzerland

Font Size: 18

Language

Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        in.nextLine();
        int i;
        String[] a = new String[n];
        for (i=0; i<n; i++) {
            a[i] = in.nextLine();
        }
        Arrays.sort(a);
        for (i=0; i<n; i++) {
            System.out.println(a[i]);
        }
    }
}
```

```
// Read the integer
int var = sc.nextInt();

// Read the leftover new line
sc.nextLine();
```

```
//defining an array of type string
String[] countries = {"Wood apple", "Blackberry", "Date", "Naseb
//sorts string array in alphabetical order or ascending order
Arrays.sort(countries);
//prints the sorted string array in ascending order
System.out.println(Arrays.toString(countries));
```

Count Special Characters

Count the special characters in the given string (excluding spaces) using java program

Sample I/O:

India is&&& my*** Country!!!

9

Font Size 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.Scanner;
import java.util.Arrays;

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        String s1 = in.nextLine();
        char[] c1 = s1.toCharArray();
        int i; int count=0;
        for (i=0; i<c1.length; i++) {
            if (Character.isLetter(c1[i])==false) {
                if (Character.isWhitespace(c1[i])==false) {
                    count++;
                }
            }
        }
        System.out.println(count);
    }
```

```
String myStr1 = "Hello";
String myStr2 = "Hello";
String myStr3 = "Another String";
System.out.println(myStr1.equals(myStr2)); // Returns true because they are equal
System.out.println(myStr1.equals(myStr3)); // false
```

Try it Yourself »

Definition and Usage

The `equals()` method compares two strings, and returns true if the strings are equal, and false if not.

Tip: Use the compareTo() method to compare two strings lexicographically.

```
System.out.println(myStr1.compareTo(myStr2));// Returns 0 because they
are equal
```

Tip: Use compareToIgnoreCase() to compare two strings lexicographyically, ignoring lower case and upper case differences.

Tip: Use the equals() method to compare two strings without consideration of Unicode values.

Returns:	An `int` value: 0 if the string is equal to the other string. < 0 if the string is lexicographically less than the other string > 0 if the string is lexicographically greater than the other string (more characters)

From <https://www.w3schools.com/java/ref_string_compareto.asp>

```java
package org.kodejava.lang;

public class CharacterIsLetterExample {
    public static void main(String[] args) {
        String name = "Kode Java 123";

        // Determines if the specified character is a letter
        if (Character.isLetter(name.charAt(5))) {
            System.out.println("The fifth character (" +
                    name.charAt(5) + ") is an alphabet!");
        }

        // Iterates all characters in the string to see if it is
        // a letter or not.
        for (char c : name.toCharArray()) {
            if (Character.isLetter(c)) {
                System.out.println(c + " is a letter.");
            } else {
                System.out.println(c + " not a letter.");
            }
        }
    }
}
```

```java
public class JavaCharacterCompareExample2{
   public static void main(String[] args) {
char firstValue = '1';
char secondValue = '2';
// compare the first char to the second
   int comp = Character.compare(firstValue, secondValue);
   if (comp< 0) {
   System.err.println("Value 1 is greater than the value 2.");
   }
   else {
   System.err.println("Value 1 is less than the second value2.");
      }
     }
}
```

Java Character isWhitespace() Method - Javatpoint

The **isWhitespace**(**char** ch) method of **Character** class determines **whether** the given **character**(Unicode code point) is a **whitespace character** or not.

Strings

Wednesday, September 7, 2022 8:31 AM

https://www.w3schools.com/java/java_ref_string.asp

charAt()	Returns the character at the specified index (position)

compareTo()	Compares two strings lexicographically	int
compareToIgnoreCase()	Compares two strings lexicographically, ignoring case differences	int
concat()	Appends a string to the end of another string	String
contains()	Checks whether a string contains a sequence of characters	boolean
contentEquals()	Checks whether a string contains the exact same sequence of characters of the specified CharSequence or StringBuffer	boolean
copyValueOf()	Returns a String that represents the characters of the character array	String
endsWith()	Checks whether a string ends with the specified character(s)	boolean

CLASS&OBJECTS [MY NOTES]

Monday, October 3, 2022 9:43 PM

- Classes contain data and behavior
- Use return inside a method if you want to get a value/smt in the main class and use the value/smt instead of just printing it
    - You can put the value returned into a variable in the main class and use it as u want
- If you have a private variable in the class, you can't "get" the variable from the main class i.e. you can only "set" the variable by passing a value to the method
- Local variables mask the instance variables if they have the same name. like if you pass name parameter into a method that only says "name" and it'll pass the parameter into name instead of whatever was already declared
    - If you want to reference to the instance variable tho use the this.x keyword
    - Then x itself will automatically refer to the local variable passed to the method
    - The this keyword is essentially a reference to the object you're in
- To run the constructors in a class alone all you need to do is: new className() in the main class
- Can also pass stuff to constructors e.g. Class c1 = new Class("passed"); would pass the string to a constructor with parameters
- Can use this(parameters) to call a constructor inside a constructor too, also just this() if no need for parameters
- Static variable (a.k.a class variables) belong to the class so they exist as only one copy
    - The opposite, instance variables exist in multiple copies as different instances of each object
- Static methods also don’t need object instansiations
    - Call directly as className.methodName();
    - Static methods can access static data bc both belong to the class;
    - Static methods can't access instance variables
    - Instance methods can access static data tho
    - Use static methods when it only deals w static data
- Final is a keyword used to note that smt is a constant and can't be reassigned
    - Static variables are automatically final constants
    - Static variables are therefore really useful as counters; increments for each object
- If variables doesn't hold anything then automatically it hold 'null' for string, 0 for int, false for boolean and 'u/0000' for char
    - But java doesn't assign default values for local variables inside a method

- Modifiers are keywords that fine-tune access to our class, its member, their scope and behavior in certain situations
    - 2 types:
        1. Access: public, private, protected, default
        2. Non-access: static, final, abstract

- Primitive vs Object data types
    - A variable of primitive data type tells compiler it can hold value of primitive data type while a variable of reference type tells compiler a reference to where an object is located.

- Array of Objects
    - Stores references of objects and not the objects themselves
    - Ppt 79

- Date Class
    - Ppt 45
- Random Class
    - Use Math.random to obtain random double value bw 0.0 and 1.0 excluding 1.0
    - Else use Random() from java.util.Random class

- Destructor
    - Opp. of constructor, used to delete object to free up space in the memory
    - There is no concept of destructor in java
    - Instead java provides a 'garbage destructor' that works the same way
    - Garbage collector is a program that automatically deletes unused objects and frees up memory so that programmer has no need to manage memory manually

- Private constructors
    - Won't be able to create an object of the class outside it
    - Doesn't allow a class to be subclassed
    - Can't access private constructor outside class
    - Used if all methods inside it are static

- Data Field Encapsulation
    - Ppt 67

- Immutable Objects
    - Ppt 83

7/9 Constructors

Wednesday, September 7, 2022 8:31 AM

- Static vs instance
- Private constructors
- This keyword to call constructors (?)
  - Constructor chaining
- Packages used to group classes
  - Only public and default classes can be accessed from another class
  - Only public classes can be accessed from another package

Notes:

- this is a reference variable used to point at the current object
  resolves the problem of ambiguity if there is ambiguity between the instance variables and parameters

```
Student(int rollno,String name,float fee){
this.rollno=rollno;
this.name=name;
this.fee=fee;
}
```

parameters (formal arguments) and instance variables are same. So, we are using this keyword to distinguish local variable and instance variable.

- **Static vs instance methods**

  Instance method are methods which require an object of its class to be created before it can be called.

  Instance methods are not stored on a per-instance basis, even with virtual methods. They're stored in a single memory location, and they only "know" which object they belong to because this pointer is passed when you call them.

  Static methods are the methods in Java that can be called without creating an object of class. They are referenced by the **class name itself** or reference to the Object of that class. i.e **ClassName.methodName(args)**.

  They are designed with the aim to be shared among all objects created from the same class.

  When to use static methods?

  - When you have code that can be shared across all instances of the same class, put that portion of code into static method.

  Instance method vs Static method

  - Instance method can access the instance methods and instance variables directly.
  - Instance method can access static variables and static methods directly.

  - Static methods can access the static variables and static methods directly.
  - Static methods can't access instance methods and instance variables directly. They must use reference to object. And static method can't use this keyword as there is no instance for 'this' to refer to.

- **Private Constructors**

  If constructor private, won't be able to create objects of that class
  - It does not allow a class to be sub-classed.
  - It does not allow to create an object outside the class.
  - If a class has a private constructor and when we try to extend the class, a compile-time error occurs.
  - We cannot access a private constructor from any other class.
  - If all the constant methods are there in our class, we can use a private constructor.
  - If all the methods are static then we can use a private constructor.
  - We can use a public function to call the private constructor if an object is not initialized.
  - We can return only the instance of that object if an object is already initialized.

  main purpose of using a private constructor is **to restrict object creation**

- **Constructor Chaining**
  - a constructor is called from another constructor in the same class through inheritence
  - When we create an instance of a derived class, all the constructors of the inherited class (base class) are first invoked, after that the constructor of the calling class (derived class) is invoked.
  - Constructor chaining is done through two ways:
    1. Within same class: "this" keyword is sued
    2. From base class: if constructors belong to different classes (parent and child class), "super" keyword is used to call constructor from base class.
  - Changing the order of the constructors won't affect the output
  - Rules of Constructor Chaining
    - An expression that uses **this** keyword must be the first line of the constructor.
    - **Order** does not matter in constructor chaining.
    - There must exist at least one constructor that does not use **this**
  - https://www.javatpoint.com/what-is-constructor-chaining-in-java

How to call a private method from another class

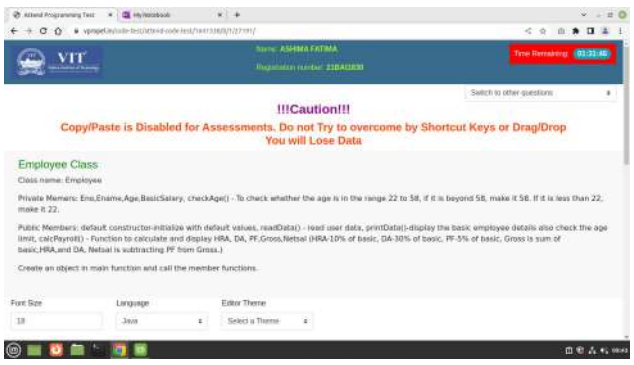
https://www.geeksforgeeks.org/static-methods-vs-instance-methods-java/#:~:text=Instance%20method%20vs%20Static%20method&text=Static%20methods%20can%20access%20the,'this'%20to%20refer%20to.

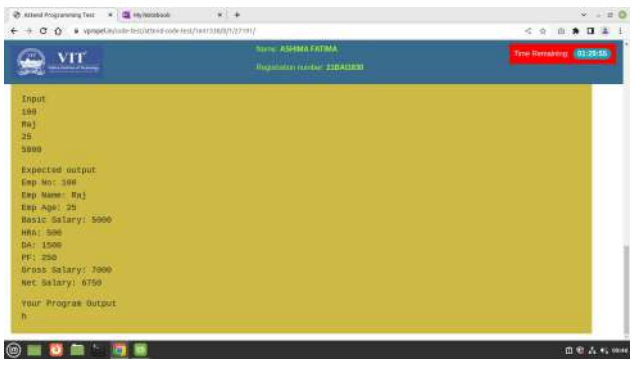
```
mport java.io.*;

class Geek {

   public static String geekName = "";

   public static void geek(String name)
   {
      geekName = name;
   }
}

class GFG {
   public static void main(String[] args)
   {

      // Accessing the static method geek()
      // and field by class name itself.
      Geek.geek("vaibhav");
      System.out.println(Geek.geekName);

      // Accessing the static method geek()
      // by using Object's reference.
      Geek obj = new Geek();
      obj.geek("mohit");
      System.out.println(obj.geekName);
   }
}
```

https://www.javatpoint.com/private-constructor-in-java

IPS 9

Wednesday, September 21, 2022 8:44 AM





- Use default constructor

https://www.javatpoint.com/java-employee-details-program

- Have to use 2 diff. Methods to get a value and to set a value

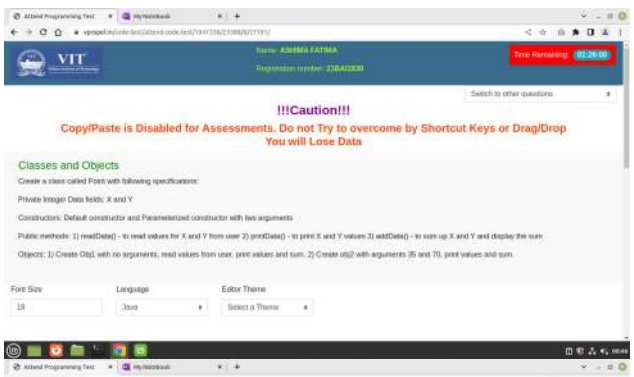
https://www.thejavaprogrammer.com/java-program-for-employee-details-using-class-and-object/

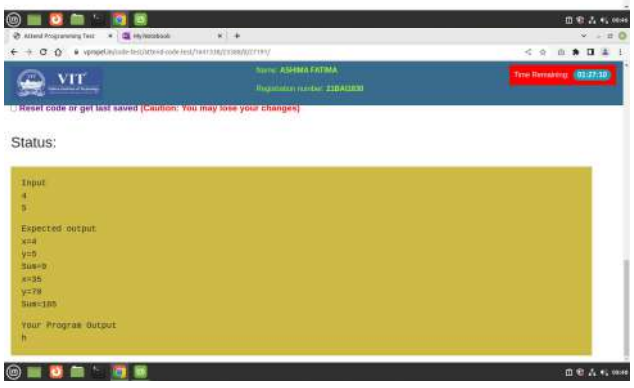
Input

100
Raj
25
5000
Expected output

Emp No: 100
Emp Name: Raj
Emp Age: 25
Basic Salary: 5000
HRA: 500
DA: 1500
PF: 250
Gross Salary: 7000
Net Salary: 6750

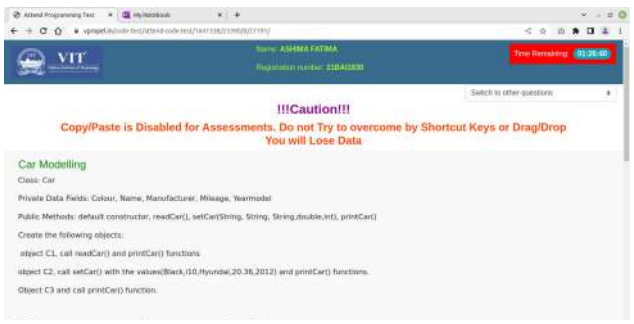
```
import java.util.Scanner;

class Employee {
   private int Eno;
   private String Ename;
   private int Age;
   private double Salary;
   private double HDR;
   private double DA;
   private double PF;
   private double G_salary;
   private double N_salary;

   public void readData(int Eno, String Ename, int Age, double Salary) {
      this.Eno = Eno;
      this.Ename = Ename;
      this.Age = Age;
      this.Salary = Salary;
   }
   private int checkAge(int Age) {
      if (Age>58) {
         Age = 58;
      }
      else if (Age<22) {
         Age = 22;
      }
      return Age;
   }
   public void calcPayroll(double Salary) {
      HDR = 0.1*Salary;
      DA = 0.3*Salary;
      PF = 0.05*Salary;
      G_salary = Salary+HDR+DA;
      N_salary = G_salary - PF;
   }
   public void printData() {
      checkAge(Age);
      calcPayroll(Salary);
      System.out.println("Emp No: "+Eno);
      System.out.println("\nEmp Name: "+Ename);
      System.out.println("\nEmp Age: "+Age);
      int iSalary = (int)Salary;
      int iHDR = (int)HDR;
      int iDA = (int)DA;
      int iPF = (int)PF;
      int iG_salary = (int)G_salary;
      int iN_salary = (int)N_salary;
      System.out.println("\nBasic Salary: "+iSalary);
      System.out.println("\nHDR: "+iHDR);
      System.out.println("\nDA: "+iDA);
      System.out.println("\nPF: "+iPF);
      System.out.println("\nGross Salary: "+iG_salary);
      System.out.println("\nNet Salary: "+iN_salary);
   }
}

class Main{
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      Employee Obj = new Employee();
      int no = in.nextInt();
      in.nextLine();
      String name = in.nextLine();
      int age = in.nextInt();
      double sal = in.nextDouble();
      Obj.readData(no, name, age, sal);
      Obj.printData();
   }
}
```






```
import java.util.Scanner;

class Point {
   private int x;
   private int y;
   private int sum;
   //default constructor means just initialise the given variables
   Point () {
      x=0; y=0;
   }
   public void readData() {
      Scanner in = new Scanner(System.in);
      x = in.nextInt();
      y = in.nextInt();
   }
   public void setData(int x, int y) {
      this.x = x;
      this.y = y;
   }
   public int addData() {
      sum = x+y;
      return sum;
   }
   public void printData() {
      System.out.print("\nx="+x);
      System.out.print("\ny="+y);
      System.out.println("\nsum="+sum);
   }
}

class Main {
   public static void main(String[] args) {
         Point Obj1 = new Point();
         Obj1.readData();
         Obj1.addData();
         Obj1.printData();
         Point Obj2 = new Point();
         Obj2.setData(35,70);
         Obj2.addData();
         Obj2.printData();
   }
}
```

https://www.includehelp.com/java-programs/java-program-to-find-area-and-perimeter-of-circle-using-class.aspx

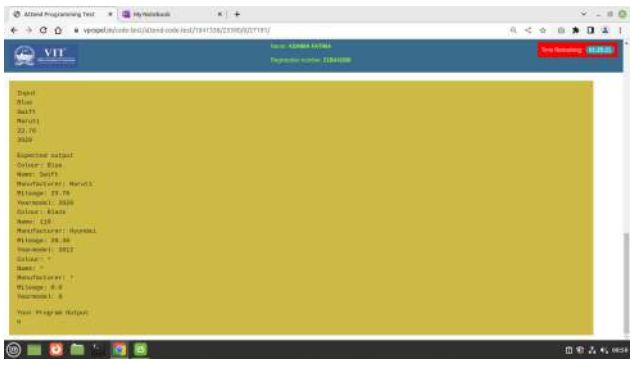


Input

Blue
Swift
Maruti
23.76
2020

Expected output

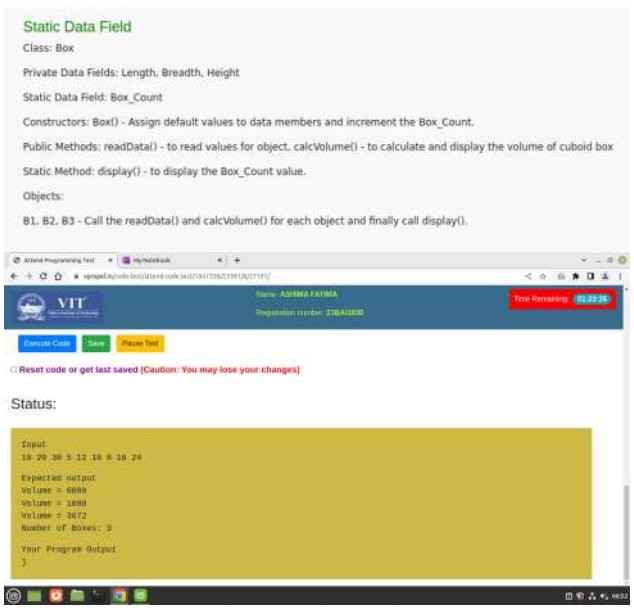
```
import java.util.Scanner;

class Car {
   private String color;
   private String name;
   private String manuf;
   private double mileage;
   private int yearModel;

   Car() {
      color = "*";
      name = "*";
      manuf = "*";
```



23.76
2020

Expected output
Colour: Blue
Name: Swift
Manufacturer: Maruti
Mileage: 23.76
Yearmodel: 2020
Colour: Black
Name: i10
Manufacturer: Hyundai
Mileage: 20.36
Yearmodel: 2012
Colour: *
Name: *
Manufacturer: *
Mileage: 0.0
Yearmodel: 0

```
  Car() {
    color = "*";
    name = "*";
    manuf = "*";
   mileage = 0.0;
   yearModel = 0;
  }
  public void readCar() {
    Scanner in = new Scanner(System.in);
    color = in.nextLine();
    name = in.nextLine();
    manuf = in.nextLine();
    mileage = in.nextDouble();
    yearModel = in.nextInt();
  }
  public void setCar(String color, String name, String manuf,
double mileage, int yearModel) {
    this.color = color;
    this.name = name;
    this.manuf = manuf;
    this.mileage = mileage;
    this.yearModel = yearModel;
  }
  public void printCar() {
    System.out.println("Color: "+color);
    System.out.println("Name: "+name);
    System.out.println("Manufacturer: "+manuf);
    System.out.println("Mileage: "+mileage);
    System.out.println("Yearmodel: "+yearModel);
  }
}

class Main {
  public static void main(String[] args) {
    Car C1 = new Car();
    C1.readCar();
    C1.printCar();
    Car C2 = new Car();
    C2.setCar("Black","i10","Hyundai",20.36,2012);
    C2.printCar();
    Car C3 = new Car();
    C3.printCar();
  }
}
```




```
import java.util.Scanner;

class Box {
  private int length;
  private int breadth;
  private int height;
  private int vol;
  static int Box_Count;

  Box() {
    length=0;
    breadth=0;
    height=0;
    Box_Count++;
  }
  public void readData() {
    Scanner in = new Scanner(System.in);
    length = in.nextInt();
    breadth = in.nextInt();
    height = in.nextInt();
  }
  public void calcVolume() {
    vol = length*breadth*height;
    System.out.println("Volume: "+ vol);
  }
  public void display() {
    System.out.println("Number of boxes: "+Box_Count);
  }
}

class Main {
  public static void main(String[] args) {
    Box B1 = new Box();
    Box B2 = new Box();
    Box B3 = new Box();
    B1.readData();
    B1.calcVolume();
    B2.readData();
    B2.calcVolume();
    B3.readData();
    B3.calcVolume();
    B3.display();
  }
}
```

Input

10 20 30 5 12 18 8 16 24
Expected output

Volume = 6000
Volume = 1080
Volume = 3072
Number of Boxes: 3

INHERITANCE [MY NOTES]

Monday, October 3, 2022 10:46 PM

- Superclass (parent class, base class), Subclass(child class, derived class)
- Class Car extends Machine means that class Car is the child class of class Machine and inherits all the methods of Machine class
    - A class can only extend one parent class
- If same method in parents and child class, method in child class will override the method in parent class
    - Only non-private methods can be overridden bc private methods are not visible in the subclass
    - Cannot override static methods
    - Cannot override fields
- Private variables can't be inherited
- Protected variables can be accessed anywhere in the package and can be inherited by the child class
- Access modifiers are the keyword used to control the visibility of fields, methods and constructors in a class
    - 4 access modifiers in java:
        Default (no keyword), public, private and protected
    - Default has package-lvl visibility
    - Default vs protected: default can only be accessed if they are in the same package and protected can be accessed within the same package and outside the package if it is inherited by a child class
- Can't have a private class bc access modifiers only apply to things within a class not classes themselves
- Super keyword is a reference variable used to refer to immediate parent class object
    - Use super.variable to refer to immediate parent class instance variable or super.method() for method
        - Super.method() is only needed when you override the method
        - Can pass parameters to it
    - Use super() to invoke immediate parent class constructor
        - Can also pass parameters to it
        - This is bc constructors don't get inherited to the subclass but the subclass's constrcutor automatically class the superclass's 0 parameter constructor before it executes any code in its own constructor
            - To prevent superclass's constructor being called must explicitly invoke one of the superclass's other constructors using super keyword and the call must be in the first line of code in the subclass's constructor
            - This is true even if superclass as no 0 parameter constructors
- Indirect inheritance is possible in java
- Polymorphism: if you have a child class, you can also use the child class where ever you would use a parent class bc everything is anyways inherited.
    - https://www.udemy.com/course/java-tutorial/learn/lecture/149501#notes
- Method Overloading is when a class has multiple methods w same name but diff parameters

SUMMARY

- Public: anywhere
- Private: within class
- Protected: within class, subclass and package
- Default: within package

- Abstraction
  - Focus on commonalities among objects in system
- "is-a" vs. "has-a"
  - "is-a"
    - Inheritance
    - subclass object treated as superclass object
    - Example: Car *is a* vehicle
      - Vehicle properties/behaviors also car properties/behaviors
  - "has-a"
    - Composition
    - Object contains one or more objects of other classes as members
    - Example: Car *has a* steering wheel

21/9 Inheritance

Wednesday, September 21, 2022 8:52 AM

- Inheritance
- Is-a, has-a relationships
- Specilization, generalisation, composition
- Inheritance – extens
- interface – infers (?)
- Polymorphism – dynamic …
- Indirect inheritance
- Abstract class and inheritance

IPS 10

Monday, September 26, 2022 9:55 AM

Dynamic Polymorphism

Create three classes *Person*, *Professor* and *Student*. The class *Person* should have data members name and age. The classes *Professor* and *Student* should inherit from the class *Person*.

The class *Professor* should have two integer members: *publications* and Empid. There will be two instance methods: *getdata* and *putdata*. The function *getdata* should get the input from the user: the *name*, *age* and *publications* of the professor. The function *putdata* should print the *name*, *age*, *publications* and the Empid of the professor.

The class *Student* should have two data fields: *marks*, which is an array of size 3 and studID. It has two instance methods: *getdata* and *putdata*. The function *getdata* should get the input from the user: the *name*, *age*, and the *marks* of the student in 3 subjects. The function *putdata* should print the *name*, *age*, *sum* of the marks and the studID of the student.

For each object being created of the *Professor* or the *Student* class, sequential id's should be assigned to them starting from 1 .

Solve this problem using dynamic polymorphism, constructors and static variables. Create two objects each for both Professor Class and Student Class.

Sample Input:

Walter 50 98

Jessie 25 15

White 18 89 96 96

Pinkman 19 54 52 45

Sample Output:

Name:Walter

Age:50

Publications:98

Professor ID:1

Name:Jessie

Age:25

Publications:15

Professor ID:2

Name:White

Age:18

Mark1:89

Mark2:96

Mark3:96

Student ID:1

Name:Pinkman

Age:19

Mark1:54

Mark2:52

Mark3:45

Student ID:2

```
Input
ABC 25 100
Sergio 30 53
Allen 18 23 25 26
Andrew 19 45 45 49

Expected output
ABC 25 100 ID:1
Sergio 30 53 ID:2
Allen 18 23 25 26 ID:1
Andrew 19 45 45 49 ID:2

Your Program Output
h
```

```
Input
Walter 50 98
Jessie 25 15
White 18 89 96 96
Pinkman 19 54 52 45

Expected output
Walter 50 98 ID:1
Jessie 25 15 ID:2
White 18 89 96 96 ID:1
Pinkman 19 54 52 45 ID:2

Your Program Output
ABC 25 100 ID:1
Sergio 30 53 ID:2
Allen 18 23 25 26 ID:1
Andrew 19 45 45 49 ID:2
```

```
import java.util.*;

class Person {
   public String name;
   public int age;

   Person() {
      name = "*";
      age = 0;
   }
}

class Proffesor extends Person{
   public int publications;
   static int Empid=0;

   Proffesor() {
      publications = 0;
      Empid++;
   }

   public void getdata(Scanner in) {
      name = in.next();
      age = in.nextInt();
      publications = in.nextInt();
   }
   public void putdata() {
      System.out.println(name+" "+age+" "+publications+" ID:"+Empid);
   }
}

class Student extends Person{
   public int[] marks = new int[3];
   static int stuID=0;
   static int i=0;

   Student() {
      for (i=0; i<3; i++) {
         marks[i]=0;
      }
      stuID++;
   }

   public void getdata(Scanner in) {
      name = in.next();
      age = in.nextInt();
      for (i=0; i<3; i++) {
         marks[i] = in.nextInt();
      }
   }
   public void putdata() {
      System.out.print(name+" "+age);
      for (i=0; i<3; i++) {
         System.out.print(" "+marks[i]);
      }
      System.out.print(" ID:"+stuID+"\n");
   }
}

public class Main {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      Proffesor prof1 = new Proffesor();
      prof1.getdata(in);
      prof1.putdata();
      Proffesor prof2 = new Proffesor();
      prof2.getdata(in);
      prof2.putdata();
      Student stu1 = new Student();
      stu1.getdata(in);
      stu1.putdata();
      Student stu2 = new Student();
      stu2.getdata(in);
      stu2.putdata();
   }
}
```

```
import java.util.Scanner;
class Person{
public String name;
public int age;
Person(){
name = "*";
age = 0;
}
public void getdata(Scanner sc){
name = sc.next();
age = sc.nextInt();
}
public void putdata(){
System.out.println(name + " " + age);
}
}
class Professor extends Person{
public int publications;
static int Empid = 0;
Professor(){
publications = 0;
Empid++;
}
public void getdata(Scanner sc){
name = sc.next();
age = sc.nextInt();
publications = sc.nextInt();
}
public void putdata(){
System.out.println(name + " " + age + " " + publications + " ID:" + Empid);
}
}
class Student extends Person{
int [] m = new int[3];
static int studID = 0;
Student(){
studID++;
}
public void getdata(Scanner sc){
name = sc.next();
age = sc.nextInt();
for (int i = 0; i < 3; i++)
m[i] = sc.nextInt();
}
public void putdata(){
System.out.println(name + " " + age + " " + m[0] + " " + m[1] + " " + m[2] + " ID:" + studID);
}
}
public class Main{
public static void main(String [] args){
Scanner sc = new Scanner(System.in);
Person p1 = new Professor();
p1.getdata(sc);
p1.putdata();
p1 = new Professor();
p1.getdata(sc);
p1.putdata();
p1 = new Student();
p1.getdata(sc);
p1.putdata();
p1 = new Student();
p1.getdata(sc);
p1.putdata();
}
```

Interface Demo

Create an Interface named Shape with common properties like

color -String, border-int, void fillColor() , void drawBorder(), void calcArea()

The colour of all shapes is Black and Border thickness of all shapes is 2.

Create the following classes which implements the Shape Interface. Include additional methods /constructors in the class to read the required parameters.

Circle

Square

Cylinder - Additionally calculate Volume also.

```
Input
5
4
5
7

Expected output
Colour of Circle is Black
Border of Circle is 2
radius = 5
Area = 78.5
Colour of Square is Black
Border of Square is 2
Side = 4
Area = 16
Colour of Cylinder is Black
Border of Cylinder is 2
radius = 5
height = 7
Total Surface Area of Cylinder = 376.8
Volume of Cylinder = 549.5
```

```
import java.util.*;

interface Shape {
   String shape;
   String color;
   int border;
   int Area;

   Shape() {
      shape = "*";
      color = "*";
      border = 0;
      Area = 0;
   }

   public void getShape(String shape) {
      this.shape = shape;
   }
   public void fillColor() {
      color = "Black";
      System.out.println("Colour of "+shape+" is "+color);
   }
   public void drawBorder() {
      border = 2;
   }
   public void calcArea(String shape) {
      if (shape.equals("Circle")) {
         public void circle(int radius)
      }
      else if (shape.equals("Square")) {

      }
      else if (shape.equals('Cylinder')) {

      }
   }
}

class Circle implements Shape {
   String shape
   int radius;

   Circle() {
      radius = 0;
   }
}

class Square implements Shape {

}

class Cylinder implements Shape {

}

class Main {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);

   }
}
```

26/9 Interface

Monday, September 26, 2022 9:55 AM

- collection of abstract methods and classes (?)
- Interface can't be instantiated
- Methods in it are public by default
- Interface vs abstract class: interface only has, by default abstarct methods; abstract classes can have normal and abstract methods
- Abstract class: keyword abstract; ment for inheritance, never instanciated, don't create objects for it, a general class from which sub classes are creates
    - Abstract classs Classname {
      }
      Abstract void methodname () { }
        - Abstract method doesn't have a body; only header
- Inherit interface: implements
- Normal class inheritance: extends
-

PACKAGES [MY NOTES]

Tuesday, October 4, 2022 1:17 PM

- Package is a folder that contains java files and all the classes inside it
- To declare it, as first line use "package packageName"
- If u wanna use classes from another package use "import packageName.fileName" or "import packageName.*" to import everything from that package
- Packages are hierarchical in java so can have packages within packages
    - Import packageName.subpackgaeName.fileName

Package Assignment

Saturday, October 8, 2022 9:22 AM

```
package Shape;

import java.util.*;
import java.lang.Math;

abstract class GeometricObject() {
   private String color;
   private boolean filled;
   protected double area;
   protected double perimeter;

   protected GeometricObject() {
      color = "White";
      filled = false;
      area = 0.0;
      perimeter = 0.0;
   }
   protected GeometricObject(String color, boolean filled) {
      //this.color = "White";
      //this.filled = false;
      area = 0.0;
      perimeter = 0.0;
   }
   String getColor(in) {
      color = in.next();
      return color;
   }
   void setColor(String color) {
      this.color = color;
   }
   boolean isFilled(in) {
      filled = in.nextBoolean();
      return filled;
   }
   void setFilled(boolean filled) {
      this.filled = filled;
   }
   abstract double getArea();
   abstract double getPerimeter();
}

class Circle extends GeometricObject {
   private double radius;
   private double diamter;

   Circle() {
      radius = 0.0;
      GeometricObject();
   }
   Circle(double radius) {
      GeometricObject();
   }
   Circle(double radius, String color, boolean filled) {
      GeometricObject(color, filled);
   }
```

```
double getRadius(in) {
   radius = in.nextInt()
   return radius;
}
void setRadius(double radius) {
   this.radius = radius;
}
double getDiameter() {
   diameter = radius*2;
   return diameter;
}
double getArea() {
   area = Math.PI*radius*radius;
   System.out.println(area);
   return area;
}
double getPerimeter() {
   perimeter = 2*Math.PI*radius;
   System.out.println(perimeter);
   return perimeter;
}

}

class Rectangle extends GeometricObject {
   private double width;
   private double height;

   Rectangle() {
      width = 0;
      height = 0;
   }
   Rectangle(double width, double height) {
      GeometricObject();
   }
   Rectangle(double width, double height, String color, boolean filled) {
      GeometricObject(color, filled);
   }
   double getWidth(in) {
      width = in.next();
      return width;
   }
   void setWidth() {
      this.width = width;
   }
   double getHeight(in) {
      height = in.next();
      return height;
   }
   void setHeight() {
      this.height = height;
   }
   double getArea() {
      area = width*height;
      System.out.println(area);
      return area;
   }
   double getPerimeter() {
      perimeter = 2*(width+height);
```

```
        System.out.println(perimeter);
        return perimeter;
    }
}

class Main() {
    public static main void(String[] args) {
        Scanner in = new Scanner(System.in);
        Circle c1 = new Circle();
        c1.getRadius();
        c1.getArea();
        c2.getPerimeter();
        Circle c2 = new Circle(2.0);
        c2.setRadius(2.0);
        c2.getArea();
        c2.getPerimeter();
        Circle c3 = new Circle(3.0, "Black", true);
        c3.setRadius(3.0, "Black", true);
        c3.getArea();
        c3.getPerimeter();
        Rectangle r1 = new Rectangle();
        r1.getWidth();
        r1.getHight();
        r1.getArea();
        r1.getPerimeter();
        Rectangle r2 = new Rectangle(5, 10);
        r2.setWidth(5);

    }
```

Create Rectangle objects as specified - r1(), r2(5,10), r3(10,15,"Red",true)

```
}
```

Packages

Monday, October 3, 2022 11:25 AM

Comparable and cloneable interfaces
StringBuilder, StringTokenizer, StringBuffer – Final Classes

- **Packages**
  - a group of similar types of classes, interfaces and sub-packages.
  - Advantage of Java Package

    1) Java package is used to categorize the classes and interfaces so that they can be easily maintained.

    2) Java package provides access protection.

    3) Java package removes naming collision.
  - Access it using packagename.*

https://www.javatpoint.com/package#:~:text=A%20java%20package%20is%20a,io%2C%20util%2C%20sql%20etc.

Exception Handling

Tuesday, October 11, 2022 10:58 PM

- ArithmeticException - divide by 0
- NullPointerException - have a null value in any variable
- NumberFormatException - can't make a string into int
- ArrayIndexOutOfBoundsException

KeyWords
- Throw
- Throws
- Finally
- Catch

Import java.io

For each try block, can have multiple catch blocks but only one finally block

IPS&PAT5 Exception

Sunday, November 13, 2022 8:37 PM

```java
import java.util.*;
import java.io.*;

class InvalidValue extends Exception {
    InvalidValue(String str) {
        super(str);
    }
}

class Main {
    public static void main(String args[]) {
        try {
            Scanner in = new Scanner(System.in);
            int value = in.nextInt();
            if (value<30) {
                throw new InvalidValue("Value too small");
            }
            else {
                System.out.println("Value: "+value);
            }
        }
        catch(InvalidValue e) {
            System.out.println(e);
        }
        finally {
            System.out.println("VIT University");
        }
    }
}
```

```
29
InvalidValue: Value too small
VIT University
```

Main.java

```java
import java.util.*;
import java.io.*;

class Main {
    public static void main(String args[]) {
        try {
            Scanner in = new Scanner(System.in);
            int value = in.nextInt();
            if (value<30) {
                throw new Exception("Value too small");
            }
            else {
                System.out.println("Value: "+value);
            }
        }
        catch(Exception e) {
            System.out.println(e.getMessage());
        }
        finally {
            System.out.println("VIT University");
        }
    }
}
```

```
29
Value too small
VIT University
```

```java
/*Read an integer from the user and print it. If any input other than integer is entered,raise an exception to user saying
"Invalid Input, Integer required" and continue to read from the user until he enters a valid integer.*/

import java.util.*;
import java.io.*;

class Main {
    public static void main(String args[]) {
        boolean valid = false;
        while (!valid) {
            try {
                Scanner in = new Scanner(System.in);
                int value = in.nextInt();
                valid = true;
                System.out.println(value);
            }
            catch(InputMismatchException e) {
                System.out.println(e+": Invalid Input, Integer required");
            }
        }
    }
}
```

Main.java

```java
/*Define a class to store student's register number, name, and an integer array to store five subject's marks.

Define methods to read, print the student's details. Read method must throw "InvalidMarksException", if the marks are
less than 0 or greater than 30.*/

import java.util.*;
import java.io.*;

class InvalidMarksException extends Exception {
    InvalidMarksException(String smarks) {
        super(smarks+" is less than 0 or greater than 30");
    }
}

class Student {
    private String no;
    private String name;
    private int[] marks = new int[5];
    int i;

    Student() {
        no ="*";
        name = "*";
        for (i=0; i<5; i++) {
            marks[i] = 0;
        }
    }

    void getData(Scanner in) throws InvalidMarksException{
        no = in.next();
        name = in.next();
        for (i=0; i<5; i++) {
            marks[i] = in.nextInt();
            if (marks[i]<0 || marks[i]>30) {
                String smarks = String.valueOf(marks[i]);
                throw new InvalidMarksException(smarks);
            }
        }
    }

    void printData() {
        System.out.println("Registration no.: "+no);
        System.out.println("Name: "+name);
        for (i=0; i<5; i++) {
            System.out.println("Mark"+i+": "+marks[i]);
        }
    }
}

class Main {
    public static void main(String args[]) {
        Scanner in = new Scanner(System.in);
        Student s1 = new Student();
        try {
            s1.getData(in);
        }
        catch(InvalidMarksException e) {
            System.out.println(e);
        }
        finally {
            s1.printData();
        }
    }
}
```

input

```
InvalidMarksException: 32 is less than 0 or greater than 30
Registration no.: bai
Name: ashima
Mark0: 24
Mark1: 32
Mark2: 0
Mark3: 0
Mark4: 0
```

Main.java

```java
/*XYZ Shop announces exclusive offer sale for three products 1. Shoes 2. Perfume 3. Chocolate.
Implement readData(Scanner) -  to read the Product name, qty and price from each user, and calculate the amount,
printData()- print the bill. The customer is restricted to choose the products into shopping cart based on the following conditions.
1) Customer can't buy more than one Shoe. 2) Bill amount can't exceed 1500 while buying Perfumes
3) Customer can't buymore than 20 Chocalates.Create InvalidChoiceException class.
Raise InvalidChoiceExcepion if the user is voilating the restrictions while choosing products.
(Assume customer buys only one product at a time) Hint: Raise all the exceptions inside readData() method*/

import java.util.*;
import java.io.*;

class InvalidChoiceException extends Exception {
    InvalidChoiceException(String msg) {
        super(msg);
    }

}

class Shop {
    private String name;
    private int qty;
    private double price;
    private double bill;
    int i;

    Shop() {
        name ="*";
        qty = 0;
        price = 0.0;
        bill =0.0;
    }

    void readData(Scanner in) throws InvalidChoiceException{
        name = in.next();
        qty = in.nextInt();
        if (name.equals("Shoe") && qty>1) {
            throw new InvalidChoiceException("Customer can't buy more than one Shoe");
        }
        else if (name.equals("Chocolates") && qty>20) {
            throw new InvalidChoiceException("Bill amount can't exceed 1500 while buying Perfumes");
        }
        else {
            price = in.nextDouble();
            bill = (qty*price);
            if (name.equals("Perfume") && bill>1500) {
                throw new InvalidChoiceException("Customer can't buymore than 20 Chocalates");
            }
        }
    }

    void printData() {
        System.out.println("Bill: "+bill);
    }
}

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        try {
            Shop s1 = new Shop();
            s1.readData(in);
        }
        catch(InvalidChoiceException e) {
            System.out.println(e);
        }
    }
}
```

Generic IPS13

Tuesday, October 11, 2022 10:58 PM

https://www.geeksforgeeks.org/how-to-implement-stack-in-java-using-array-and-generics/


Generic Queue


Generic Stack

Generic Sort

Implement a generic method to sort an array of n generic elements in ascending order.

Sample Input

5

20 14 65 78 25

6

12.14 21.10 245.24 8.2 7.2 69.2

4

Son Hen Den Que

Sample Output:

14 20 25 65 78

7.2 8.2 12.14 21.10 69.2 245.24

Den Hen Que Son

From <https://www.vpropel.in/code-test/attend-code-test/1692446/24313/0/27978/>

10/12/22, 12:20 AM Attend Programming Test

Switch to other questions

!!!Caution!!!

Copy/Paste is Disabled for Assessments. Do not Try to overcome by Shortcut Keys or Drag/Drop
You will Lose Data

Generic Queue using ArrayList

Implement a Generic Queue using ArrayList

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Pasaed Execution!

```
import java.util.*;
import java.io.*;

class Queue<I> {
    private int front;
    private int rear;
    ArrayList<I> queue;

    Queue() {
        front =-1;
        rear = -1;
        this.queue = new ArrayList<>();
    }

    void enqueue(I item) {
        if (front==-1) {
            front++;
        }
        rear++;
        queue.add(item);
    }

    void dequeue() {
        if (front!= -1 && rear != -1) {
            System.out.println("Queue is
deleted: "+queue.get(front));
            front++;
        }
    }

    void display() {
        if (front!= -1 && rear != -1) {
            int i=front;
            while (i!=rear+1) {
                System.out.print("
"+queue.get(i));
                i++;
            }
            System.out.print("\n");
        }
    }
}

class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int i;
```

https://www.vpropel.in/code-test/attend-code-test/1692446/24325/0/27978/ 1/2

```
        Queue<Integer> q1 = new Queue<>();

        for (i=0; i<6; i++) {
            q1.enqueue(in.nextInt());
        }
        System.out.print("Queue Contents:");
        q1.display();
        System.out.println("Queue insert done");
        q1.enqueue(in.nextInt());
        System.out.print("Queue is:");
        q1.display();
        q1.dequeue();
        q1.dequeue();
        System.out.print("Queue is:");
        q1.display();
        Queue<String> q2 = new Queue<>();
        for (i=0; i<6; i++) {
            q2.enqueue(in.next());
        }
        System.out.print("Queue is:");
        q2.display();
        System.out.println("Queue insert done");
        q2.enqueue(in.next());
        System.out.print("Queue is:");
        q2.display();
        q2.dequeue();
        q2.dequeue();
        System.out.print("Queue is:");
        q2.display();
        Queue<Double> q3 = new Queue<>();
        for (i=0; i<6; i++) {
            q3.enqueue(in.nextDouble());
        }
        System.out.print("Queue is:");
        q3.display();
        System.out.println("Queue insert done");
        q3.enqueue(in.nextDouble());
        System.out.print("Queue is:");
        q3.display();
        q3.dequeue();
        q3.dequeue();
        System.out.print("Queue is:");
        q3.display();
    }
}
```

Save Pause Test

Submit Code

Status:

IPS13 Generic...

Switch to other questions

!!!Caution!!!

Copy/Paste is Disabled for Assessments. Do not Try to overcome by Shortcut Keys or Drag/Drop
You will Lose Data

Implementation of Generic Stack using ArrayList

Create a GenericStack that can hold 1) Integers 2) Doubles 3) Strings. Implement push(element), pop(), peek(), isEmpty(), size() methods to operate the stack. Call methods in sequence as per the test case.

Font Size: 18 | Language | Editor Theme: Select a Theme

Your code has Passed Execution!

```
import java.util.*;
import java.io.*;

class Stack <I>{
    private int top;
    ArrayList<I> stackList;
    Stack () {
        int top =-1;
        this.stackList =  new ArrayList<I>();

    }

    void push(I item) {
        top++;
        stackList.add(item);
    }

    void pop() {
        top--;
        System.out.println("Stack is Popped:
"+stackList.get(top));
        stackList.remove(top);
    }

    void display() {
        int i=0;
        System.out.print("Stack Contents:");
        while (i!=top) {
            System.out.print("
"+stackList.get(i));
            i++;
        }
        System.out.print("\n");
    }
 }


class Main {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int i;
        Stack<Integer> s1 = new Stack<>();
        for (i=0; i<6; i++) {
            s1.push(in.nextInt());
        }
        s1.display();
```

```
        System.out.println("Stack Push done");

        s1.push(in.nextInt());
        s1.display();
        s1.pop();
        s1.pop();
        s1.display();
        Stack<String> s2 = new Stack<>();
        for (i=0; i<6; i++) {
            s2.push(in.next());
        }
        s2.display();
        System.out.println("Stack Push done");
        s2.push(in.next());
        s2.display();
        s2.pop();
        s2.pop();
        s2.display();
        Stack<Double> s3 = new Stack<>();
        for (i=0; i<6; i++) {
            s3.push(in.nextDouble());
        }
        s3.display();
        System.out.println("Stack Push done");
        s3.push(in.nextDouble());
        s3.display();
        s3.pop();
        s3.pop();
        s3.display();

    }
}
```

Save Pause Test

Submit Code

Status:

Implement a generic method to sort an array of n generic elements in ascending order.
Sample Input
5
20 14 65 78 25
6
12.14 21.10 245.24 8.2 7.2 69.2
4
Son Hen Den Que
Sample Output
14 20 25 65 78
7.2 8.2 12.14 21.10 69.2 245.24
Den Hen Que Son
**********/

```
import java.util.*;
public class GENERICSort{
public static <E extends Comparable<? super E>> void sort(E [] a){
for (int i = 0; i < a.length;i++){
for (int j = 0; j < a.length - i - 1;j++){
if (a[j+1].compareTo(a[j])<0){
E temp = a[j];
a[j] = a[j + 1];
a[j + 1] = temp;
}
}
}
}
public static <E> void print(E [] list){
for (int i = 0; i < list.length; i++){
if (i != list.length - 1)
System.out.print(list[i] + " ");
else
System.out.println(list[i]);
}
}
public static void main(String [] args){
Scanner sc = new Scanner(System.in);
int n = sc.nextInt();
Integer [] iObj = new Integer[n];
for (int i = 0; i<n;i++){
iObj[i] = sc.nextInt();
}
sort(iObj);
print(iObj);
n = sc.nextInt();
Double [] dObj = new Double[n];
for (int i = 0; i<n;i++){
dObj[i] = sc.nextDouble();
}
sort(dObj);
print(dObj);
n = sc.nextInt();
String [] sObj = new String[n];
for (int i = 0; i<n;i++){
sObj[i] = sc.next();
}
sort(sObj);
print(sObj);
}
}
```

https://cs.brynmawr.edu/Courses/cs206/fall2013/slides/08_Sorting_GenericMethod.pdf

Generic

Wednesday, November 2, 2022 9:21 AM

```
import java.util.*;

class Sales{
   String name;
   int sales;

   public String getname(){
      return name;
   }
   public void setname(String name){
      this.name=name;
   }
   public int getsales(){
      return sales;
   }
   public void setsales(int sales){
      this.sales=sales;
   }
}

class SalesComparator implements Comparator <Sales>, java.io.Serializable{
   public int compare(Sales s1, Sales s2){
      int sales1 = s1.getsales();
      int sales2 = s2.getsales();
      if(sales1<sales2)
         return -1;
      else if(sales1>sales2)
         return 1;
      else
         return 0;
   }
}

class Main{
   public static void main(String[] args){
      Sales s1 = new Sales();
      Sales s2 = new Sales();
      s1.setname("Andrew");
      s1.setsales(100);
      s2.setname("Sam");
      s2.setsales(98);

      Sales s = max(s1, s2, new SalesComparator());
      System.out.println("The good Sales are done by: "+s.getname());
   }
   public static Sales max(Sales s1,Sales s2, Comparator<Sales> c) {
      if (c.compare(s1, s2) > 0)
         return s1;
      else
         return s2;
   }
}
```

Java Map

Monday, October 31, 2022 10:38 AM

1. Comparator for salesperson class
2. Occurrence of words using maps

Contains method

- There are two interfaces for implementing Map in java: Map and SortedMap, and three classes: HashMap, LinkedHashMap, and TreeMap.
- A Map doesn't allow duplicate keys, but you can have duplicate values.
- HashMap and LinkedHashMap allow null keys and values, but TreeMap doesn't allow any null key or value.
- Hashmap and Linkedhashmap maintain insertion order but TreeMap maintains ascending order

METHODS:

- V put(Object key, Object value)
- V putIfAbsent(K key, V value)
- V remove(Object key)
- void clear() - used to reset the map
- boolean containsValue(Object value) - returns true if some value equal to the value exists within the map, else return false.
- boolean containsKey(Object key) - returns true if some key equal to the key exists within the map, else return false.
- boolean equals(Object o) - used to compare the specified Object with the Map.
- V get(Object key)
- V replace(K key, V value)
- boolean replace(K key, V oldValue, V newValue)
- int size() - returns the number of entries in the map.

1830 java

```
import java.util.*;
import java.io.*;
//The house was white and white and was very white
class Main {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      System.out.println("Enter line\n");
      String line = "The house was white and white and was very white";
      //String line = in.nextLine();
      String[] words = line.split("\\s");
      Map<String ,Integer> dict = new HashMap<String ,Integer>();
      int n = words.length;
      int i, count=0;
      for (i=0; i<n; i++) {
         boolean found = dict.containsKey(words[i]);
         //dict.put(words[i], 0);
         System.out.println(found);
         if (found = false) {
            dict.put(words[i], 0);
         }
         System.out.println(dict.getValue(words[i]));
         /*else {
            System.out.println(dict.get(words[i]));
            count = dict.get(words[i]) + 1;
            dict.replace(words[i], count);
         }*/
      }
   }
}
```

Java Files

Wednesday, November 9, 2022 8:50 AM

PAT1

Monday, November 14, 2022 8:08 AM

```
for (int i = 2; i <= num / 2; ++i) {
  // condition for nonprime number
  if (num % i == 0) {
    flag = true;
    break;
  }
}
```

Here, note that we are looping from 2 to num/2. It is because a number is not divisible by more than its half.

From <https://www.programiz.com/java-programming/examples/prime-number>

```
/*Read an array of 'n' integers, check whether each integer is a prime number or not. If any given
number is 0 or 1,
display "It is neither prime nor composite". Also print the numbers which are perfect numbers in the
given list.
Also print the sum of even numbers and odd numbers from the given list.
Perfect Number - a number that is equal to the sum of its proper divisors.
Prime Number - a number that can be divided exactly only by itself and 1*/

import java.util.*;

class Main {
   public static void main(String[] args) {
      Scanner in = new Scanner(System.in);
      int i, j;
      int n = in.nextInt();
      int[] arr = new int[n];
      for (i=0; i<n; i++) {
         arr[i] = in.nextInt();
      }
      for (i=0; i<n; i++) {
         int notprime=0;
         if (arr[i]==0|| arr[i]==1) {
            System.out.println(arr[i]+" is neither prime nor composite");
         }
         else {
            for (j=0; j<arr[i]/2; j++) {
               if (arr[i]%j==0) {
                  notprime=1;
                  break;
               }
            }
            if (notprime==0) {
               System.out.println(arr[i]+" is prime");
            }
            else {
               System.out.println(arr[i]+" is not prime");
            }
         }

      }
   }
}
```

FAT java learnings

Friday, November 18, 2022 12:26 PM

- Don't forget to return in getData methods
- in.nextBoolean()
- abstract class
    - Public abstract method int xx();
    - Extend the abstract class when u inherit
    - Only initialize new variables in class constructor
    - If need to use abstract class constructor with parameters use super(x,x) //super() calls the parent constructor
    - Call the setData() methods inside the constructor
    - Abstract methods don't have any content inside
    - Abstract class is just a given framework
    - Call abstract methods exactly as they are in the abstract class e.g. public double getData()
- Math.PI to multiply w pi
- Math.pow for exponents
- Use printf("%2f", x) to determine no. of decimal point
- Don't forget to pass in and Scanner in to the methods
- Just return sometimes for simple calculation don't need a variable for those

- To sort char arrays use: Arrays.sort(array_name, start_index, end_index);

- To convert from binary to decimal:

```
int num = in.nextInt();
      int dec = 0, base=1, rem=0;
      while (num>0) {
         rem = num % 10;
         dec = dec+(rem*base);
         num /= 10;
         base *= 2;
      }
```

  Decimal to binary:

  Step 1: Divide the number by 2 through % (modulus operator) and store the remainder in array
  Step 2: Divide the number by 2 through / (division operator)
  Step 3: Repeat the step 2 until number is less than 0

- To compare char: Character.compare(char x, char y) ==0 if equal

- Strings if char >=65 && <= 90 -----> upper case
    - >=97 && <122 -------> lower case
    - == 32 --------> a space

- Can only calc b/w int and int or double and double
    - To convert b/w them e.g. double-> int: int v_name = (int) double_name