

[Practice](#)[Compete](#)[Jobs](#)[Rank](#)[Leaderboard](#)[Dashboard](#) > [Java](#) > [Advanced](#) > [Java Varargs - Simple Addition](#)

Points: 658 Rank: 3431

Java Varargs - Simple Addition

by [aa1992](#)

Problem

Submissions

Leaderboard

Discussions

Editorial

You are given a class *Solution* and its *main* method in the editor.

Your task is to create the class *Add* and the required methods so that the code prints the *sum of the numbers* passed to the function *add*.

Note: Your *add* method in the *Add* class must print the *sum* as given in the *Sample Output*

Input Format

There are six lines of input, each containing an integer.

Output Format

There will be only four lines of output. Each line contains the sum of the *integers* passed as the parameters to *add* in the *main* method.

Sample Input

```
1
2
3
4
5
6
```

Sample Output

```
1+2=3
1+2+3=6
1+2+3+4+5=15
1+2+3+4+5+6=21
```

[f](#) [t](#) [in](#)Submissions: [15549](#)

Max Score: 15

Difficulty: Easy

Rate This Challenge:

Download problem statement

Download sample test cases



Suggest Edits

[Collapse](#)



```
1 import java.io.*;
2 import java.lang.reflect.*;
3 import java.util.*;
4 import java.text.*;
5 import java.math.*;
6 import java.util.regex.*;
7
8 class Add {
9     void add(int... nums) {
10         int sum = 0, i = 0;
11         for (int n : nums) {
12             sum = sum + n;
13             if (i++ != nums.length - 1)
14                 System.out.print(n + "+");
15             else
16                 System.out.print(n + "=");
17         }
18         System.out.println(sum);
19     }
20 }
21
22
23
24
25 public class Solution {
26
27     public static void main(String[] args) {
28         try{
29             BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
30             int n1=Integer.parseInt(br.readLine());
31             int n2=Integer.parseInt(br.readLine());
32             int n3=Integer.parseInt(br.readLine());
33             int n4=Integer.parseInt(br.readLine());
34             int n5=Integer.parseInt(br.readLine());
35             int n6=Integer.parseInt(br.readLine());
36             Add ob=new Add();
37             ob.add(n1,n2);
38             ob.add(n1,n2,n3);
39             ob.add(n1,n2,n3,n4,n5);
40             ob.add(n1,n2,n3,n4,n5,n6);
41             Method[] methods=Add.class.getDeclaredMethods();
42             Set<String> set=new HashSet<>();
43             boolean overload=false;
44             for(int i=0;i<methods.length;i++)
45             {
46                 if(set.contains(methods[i].getName()))
47                 {
48                     overload=true;
49                     break;
50                 }
51                 set.add(methods[i].getName());
52             }
53             if(overload)
54             {
55                 throw new Exception("Overloading not allowed");
56             }
57             catch(Exception e)
58             {
59                 e.printStackTrace();
60             }
61         }
62     }
63 }
64
65
66
67
68 }
69
```

 [Upload Code as File](#)☐ Test against custom input[Run Code](#)[Submit Code](#)

Processing

 Test Case #0 Test Case #3 Test Case #1 Test Case #4 Test Case #2

Join us on IRC at [#hackerrank](#) on freenode for hugs or bugs.

[Contest Calendar](#) | [Blog](#) | [Scoring](#) | [Environment](#) | [FAQ](#) | [About Us](#) | [Support](#) | [Careers](#) | [Terms Of Service](#) | [Privacy Policy](#) | [Request a Feature](#)