Prepare > Java > Object Oriented Programming > Java Method Overriding    Exit Full Screen View

Problem
Submissions
Leaderboard
Discussions
Editorial

When a subclass inherits from a superclass, it also inherits its methods; however, it can also override the superclass methods (as well as declare and implement new ones). Consider the following Sports class:

```
class Sports{
    String getName(){
        return "Generic Sports";
    }
    void getNumberOfTeamMembers(){
        System.out.println( "Each team ha
    }
}
```

Next, we create a Soccer class that inherits from the Sports class. We can override the getName method and return a different, subclass-specific string:

```
class Soccer extends Sports{
    @Override
    String getName(){
        return "Soccer Class";
    }
}
```

**Note:** When overriding a method, you should precede it with the `@Override` annotation. The parameter(s) and return type of an overridden method must be exactly the same as those of the method inherited from the supertype.

**Task**

Complete the code in your editor by writing an overridden getNumberOfTeamMembers method that prints the same statement as the superclass' getNumberOfTeamMembers method, except that it replaces $n$ with $11$ (the number of players on a Soccer team).

**Output Format**

```
}

public class Solution{

    public static void main(String []args){
        Sports c1 = new Sports();
        Soccer c2 = new Soccer();
        System.out.println(c1.getName());
```

Line: 23 Col: 2

Upload Code as File    Run Code    Submit Code

□ Test against custom input

You have earned 10.00 points!

You are now 97 points away from the gold level for your java badge.

3%    153/250

Java ★★★★

**Congratulations**

You solved this challenge. Would you like to challenge your friends?    Next Challenge

✓ Test case 0

Compiler Message

```
Success
```

Expected Output    Download

```
Generic Sports
Each team has n players in Generic Sports
Soccer Class
```

Prepare > Java > Object Oriented Programming > Java Interface

Exit Full Screen View

Problem

Submissions

Leaderboard

Discussions

Editorial

A Java interface can only contain method signatures and fields. The interface can be used to achieve polymorphism. In this problem, you will practice your knowledge on interfaces.

You are given an interface AdvancedArithmetic which contains a method signature int divisor_sum(int n). You need to write a class called MyCalculator which implements the interface.

divisorSum function just takes an integer as input and return the sum of all its divisors. For example divisors of 6 are 1, 2, 3 and 6, so divisor_sum should return 12. The value of n will be at most 1000.

Read the partially completed code in the editor and complete it. You just need to write the MyCalculator class only. Your class shouldn't be public.

**Sample Input**

```
6
```

**Sample Output**

```
I implemented: AdvancedArithmetic
12
```

**Explanation**

Divisors of 6 are 1,2,3 and 6. 1+2+3+6=12.

```
            Class[] theInterfaces = o.getClass().getI
            for (int i = 0; i < theInterfaces.length;
                String interfaceName = theInterfaces[
                System.out.println(interfaceName);
            }
        }
    }


```

Line: 20 Col: 2

Upload Code as File

Run Code

Submit Code

□ Test against custom input

You have earned 10.00 points!

You are now 7 points away from the 4th star for your java badge.

**90%** 143/150

Java ★★★

## Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

✓ **Test case 0**

✓ Test case 1

✓ Test case 2

✓ Test case 3 🔒

Compiler Message

```
Success
```

Input (stdin) Download

```
1   6
```

Expected Output Download

Prepare > Java > Data Structures > Java BitSet Exit Full Screen View

Problem

Submissions

Leaderboard

Discussions

Editorial

Java's BitSet class implements a vector of bit values (i.e.: $false$ ($0$) or $true$ ($1$)) that grows as needed, allowing us to easily manipulate bits while optimizing space (when compared to other collections). Any element having a bit value of $1$ is called a set bit.

Given $2$ BitSets, $B_1$ and $B_2$, of size $N$ where all bits in both BitSets are initialized to $0$, perform a series of $M$ operations. After each operation, print the number of set bits in the respective BitSets as two space-separated integers on a new line.

**Input Format**

The first line contains $2$ space-separated integers, $N$ (the length of both BitSets $B_1$ and $B_2$) and $M$ (the number of operations to perform), respectively.
The $M$ subsequent lines each contain an operation in one of the following forms:

- AND $<\text{set}> <\text{set}>$
- OR $<\text{set}> <\text{set}>$
- XOR $<\text{set}> <\text{set}>$
- FLIP $<\text{set}> <\text{index}>$
- SET $<\text{set}> <\text{index}>$

In the list above, $<\text{set}>$ is the integer $1$ or $2$, where $1$ denotes $B_1$ and $2$ denotes $B_2$.
$<\text{index}>$ is an integer denoting a bit's index in the BitSet corresponding to $<\text{set}>$.

For the binary operations $AND$, $OR$, and $XOR$, operands are read from left to right and the BitSet resulting from the operation replaces the contents of the first operand. For example:

```
AND 2 1
```

$B_2$ is the left operand, and $B_1$ is the right operand. This operation should assign the result of $B_2 \wedge B_1$ to $B_2$.

```
            }
            if(q.equals("OR")) {
                b[left].or(b[right]);
}
            if(q.equals("XOR")) {
                b[left].xor(b[right]);
}
            if(q.equals("FLIP")) {
                b[left].flip(N - right - 1);
            }
            if(q.equals("SET")) {
                b[left].set(N - right - 1);
            }
            System.out.println(b[0].cardinality()
.cardinality());
        }
    }
}

```

Line: 35 Col: 4

Upload Code as File

Run Code

Submit Code

□ Test against custom input

✓ **Test case 0**

✓ Test case 1 🔒

✓ Test case 2 🔒

✓ Test case 3 🔒

✓ Test case 4 🔒

✓ Test case 5 🔒

✓ Test case 6 🔒

Compiler Message

```
Success
```

Input (stdin) Download

```
5 4
AND 1 2
SET 1 4
FLIP 2 2
OR 2 1
```

Expected Output Download

```
0 0
```

Prepare > Java > Object Oriented Programming > Java Abstract Class

Exit Full Screen View

Problem

Submissions

Leaderboard

Discussions

Editorial

A Java abstract class is a class that can't be instantiated. That means you cannot create new instances of an abstract class. It works as a base for subclasses. You should learn about Java Inheritance before attempting this challenge.

Following is an example of abstract class:

```
abstract class Book{
    String title;
    abstract void setTitle(String s);
    String getTitle(){
        return title;
    }
}
```

If you try to create an instance of this class like the following line you will get an error:

```
Book new_novel=new Book();
```

You have to create another class that extends the abstract class. Then you can create an instance of the new class.

Notice that setTitle method is abstract too and has no body. That means you must implement the body of that method in the child class.

In the editor, we have provided the abstract Book class and a Main class. In the Main class, we created an instance of a class called MyBook. Your task is to write just the MyBook class.

Your class mustn't be public.

**Sample Input**

```
A tale of two cities
```

**Sample Output**

Change Theme

Language Java 8

```
    @Override
    void setTitle(String s){
        this.title = s;
    }

}
public class Main{

    public static void main(String []args){
        //Book new_novel=new Book(); This line pr
Book is abstract; cannot be instantiated
        Scanner sc=new Scanner(System.in);
        String title=sc.nextLine();
        MyBook new_novel=new MyBook();
        new_novel.setTitle(title);
        System.out.println("The title is: "+new_r
        sc.close();

    }
}
```

Line: 19 Col: 2

Upload Code as File

Run Code

Submit Code

□ Test against custom input

You have earned 10.00 points!

You are now 17 points away from the 4th star for your java badge.

**76%** 133/150

Java ★★★

## Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

Prepare > Java > Data Structures > Java Dequeue

Exit Full Screen View

Problem

Submissions

Leaderboard

Discussions

Editorial

In computer science, a double-ended queue (dequeue, often abbreviated to deque, pronounced deck) is an abstract data type that generalizes a queue, for which elements can be added to or removed from either the front (head) or back (tail).

Deque interfaces can be implemented using various types of collections such as `LinkedList` or `ArrayDeque` classes. For example, deque can be declared as:

```
Deque deque = new LinkedList<>();
or
Deque deque = new ArrayDeque<>();
```

You can find more details about Deque here.

In this problem, you are given $N$ integers. You need to find the maximum number of unique integers among all the possible contiguous subarrays of size $M$.

Note: Time limit is $3$ second for this problem.

**Input Format**

The first line of input contains two integers $N$ and $M$: representing the total number of integers and the size of the subarray, respectively. The next line contains $N$ space separated integers.

**Constraints**

$1 \leq N \leq 100000$

$1 \leq M \leq 100000$

$M \leq N$

The numbers in the array will range between $[0, 10000000]$.

**Output Format**

Print the maximum number of unique integers among all possible contiguous subarrays of size $M$.

**Sample Input**

Change Theme Language Java 8

```
    import java.util.*;
    public class test {
        public static void main(String[] args) {
            Scanner in = new Scanner(System.in);
            final Deque<Integer> deque = new Arra
              final Map<Integer, Integer> map = r
Integer>();
            final int n = in.nextInt();
            final int m = in.nextInt();
            int res =0;

            for (int i = 0; i < n; i++) {
                int num = in.nextInt();
                  deque.addLast(num);
            if (map.containsKey(num)) {
                map.put(num, map.get(num).intValu
            } else {
                map.put(num, 1);
            }
```

Line: 8 Col: 19

Upload Code as File

Run Code

Submit Code

□ Test against custom input

You have earned 20.00 points!

You are now 47 points away from the 4th star for your java badge.

33% 103/150

Java ★★★

## Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

# Solve Me First ★

9 more points to get your first star!

Rank: **2656246** | Points: **21/30**

**Your Solve Me First submission got 1.00 points.** Share Tweet

You are now 9 points away from the 1st star for your problem solving badge.

**Try the next challenge** | **Try a Random Challenge**

Problem Submissions Leaderboard Editorial

Complete the function solveMeFirst to compute the sum of two integers.

**Example**

$a = 7$

$b = 3$

Return $10$.

**Function Description**

Complete the solveMeFirst function in the editor below.

solveMeFirst has the following parameters:

- int a: the first value
- int b: the second value

Returns

- int: the sum of $a$ and $b$

**Constraints**

$1 \leq a, b \leq 1000$

**Sample Input**

```
a = 2
b = 3
```

**Sample Output**

```
5
```

**Explanation**

$2 + 3 = 5$.

Change Theme Language Java 8

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {


    static int solveMeFirst(int a, int b) {
```

```
    static int solveMeFirst(int a, int b) {
        // Hint: Type return a+b; below
           return a+b;

   }

public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int _a = in.nextInt();
        int _b = in.nextInt();
        int sum = solveMeFirst(_a, _b);
        System.out.println(sum);
   }
```

Line: 12 Col: 11

Upload Code as File □ Test against custom input Run Code Submit Code

You have earned 1.00 point!

You are now 9 points away from the 1st star for your problem solving badge.

**70%** 21/30

Problem Solving

## Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

✓ **Test case 0**

✓ Test case 1

Compiler Message

```
Success
```

Input (stdin) Download

```
2
3
```

Expected Output Download

```
5
```

Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy