### Parenthesis Checker

Given an expression string **x**. Examine whether the pairs and the orders of {,},(,),[,] are correct in exp.  
For example, the function should return 'true' for exp = [()]{}{[()()]()} and 'false' for exp = [(]).

**Note:**The drive code prints "balanced" if function return true, otherwise it prints "not balanced".

**Example 1:**

**Input**:

{([])}

**Output**:

true

**Explanation**:

{ ( [ ] ) }. Same colored brackets can form

balanced pairs, with 0 number of

unbalanced bracket.

**Example 2:**

**Input**:

()

**Output**:

true

**Explanation**:

(). Same bracket can form balanced pairs,

and here only 1 type of bracket is

present and in balanced way.

**Example 3:**

**Input**:

([]

**Output**:

false

**Explanation**:

([]. Here square bracket is balanced but

the small bracket is not balanced and

Hence , the output will be unbalanced.

### JAVA SOLUTION

import java.util.\*;

import java.io.\*;

import java.lang.\*;

class Driverclass

{

public static void main(String args[])

{

Scanner sc = new Scanner(System.in);

//Reading total number of testcases

int t= sc.nextInt();

while(t-- >0)

{

//reading the string

String st = sc.next();

//calling ispar method of Paranthesis class

//and printing "balanced" if it returns true

//else printing "not balanced"

if(new Solution().ispar(st) == true)

System.out.println("balanced");

else

System.out.println("not balanced");

}

}

}

class Solution

{

//Function to check if brackets are balanced or not.

static boolean ispar(String s)

{

int c=0;

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

if(s.charAt(i)=='('){

c++;

}

else if(s.charAt(i)==')'){

c--;

}

else if(s.charAt(i)=='{')

c++;

else if(s.charAt(i)=='}')

c--;

else if(s.charAt(i)=='[')

c++;

else if(s.charAt(i)==']')

c--;

if(c<0)

return false;

}

if(c==0)

return true;

else

return false;

}

}