**Java Loop Controls**

There may be a situation when we need to execute a block of code several number of times, and is often referred to as a loop.

Java has very flexible three looping mechanisms. You can use one of the following three loops:

* while Loop
* do...while Loop
* for Loop

As of Java 5, the *enhanced for loop* was introduced. This is mainly used for Arrays.

* Enhanced for Loop

The while Loop:

A while loop is a control structure that allows you to repeat a task a certain number of times.

Syntax:

The syntax of a while loop is:

while(Boolean\_expression)

{

//Statements

}

When executing, if the *boolean\_expression* result is true, then the actions inside the loop will be executed. This will continue as long as the expression result is true.

Here, key point of the *while* loop is that the loop might not ever run. When the expression is tested and the result is false, the loop body will be skipped and the first statement after the while loop will be executed.

Example:

publicclassTest{

publicstaticvoid main(String args[]){

int x =10;

while( x <20){

System.out.print("value of x : "+ x );

x++;

System.out.print("\n");

}

}

}

This would produce the following result:

value of x :10

value of x :11

value of x :12

value of x :13

value of x :14

value of x :15

value of x :16

value of x :17

value of x :18

value of x :19

The do...while Loop:

A do...while loop is similar to a while loop, except that a do...while loop is guaranteed to execute at least one time.

Syntax:

The syntax of a do...while loop is:

do

{

//Statements

}while(Boolean\_expression);

Notice that the Boolean expression appears at the end of the loop, so the statements in the loop execute once before the Boolean is tested.

If the Boolean expression is true, the flow of control jumps back up to do, and the statements in the loop execute again. This process repeats until the Boolean expression is false.

Example:

publicclassTest{

publicstaticvoid main(String args[]){

int x =10;

do{

System.out.print("value of x : "+ x );

x++;

System.out.print("\n");

}while( x <20);

}

}

This would produce the following result:

value of x :10

value of x :11

value of x :12

value of x :13

value of x :14

value of x :15

value of x :16

value of x :17

value of x :18

value of x :19

The for Loop:

A for loop is a repetition control structure that allows you to efficiently write a loop that needs to execute a specific number of times.

A for loop is useful when you know how many times a task is to be repeated.

Syntax:

The syntax of a for loop is:

for(initialization;Boolean\_expression; update)

{

//Statements

}

Here is the flow of control in a for loop:

* The initialization step is executed first, and only once. This step allows you to declare and initialize any loop control variables. You are not required to put a statement here, as long as a semicolon appears.
* Next, the Boolean expression is evaluated. If it is true, the body of the loop is executed. If it is false, the body of the loop does not execute and flow of control jumps to the next statement past the for loop.
* After the body of the for loop executes, the flow of control jumps back up to the update statement. This statement allows you to update any loop control variables. This statement can be left blank, as long as a semicolon appears after the Boolean expression.
* The Boolean expression is now evaluated again. If it is true, the loop executes and the process repeats itself (body of loop, then update step, then Boolean expression). After the Boolean expression is false, the for loop terminates.

Example:

publicclassTest{

publicstaticvoid main(String args[]){

for(int x =10; x <20; x = x+1){

System.out.print("value of x : "+ x );

System.out.print("\n");

}

}

}

This would produce the following result:

value of x :10

value of x :11

value of x :12

value of x :13

value of x :14

value of x :15

value of x :16

value of x :17

value of x :18

value of x :19

Enhanced for loop in Java:

As of Java 5, the enhanced for loop was introduced. This is mainly used for Arrays.

Syntax:

The syntax of enhanced for loop is:

for(declaration : expression)

{

//Statements

}

* **Declaration:** The newly declared block variable, which is of a type compatible with the elements of the array you are accessing. The variable will be available within the for block and its value would be the same as the current array element.
* **Expression:** This evaluates to the array you need to loop through. The expression can be an array variable or method call that returns an array.

Example:

publicclassTest{

publicstaticvoid main(String args[]){

int[] numbers ={10,20,30,40,50};

for(int x : numbers ){

System.out.print( x );

System.out.print(",");

}

System.out.print("\n");

String[] names ={"James","Larry","Tom","Lacy"};

for(String name : names ){

System.out.print( name );

System.out.print(",");

}

}

}

This would produce the following result:

10,20,30,40,50,

James,Larry,Tom,Lacy,

The break Keyword:

The *break* keyword is used to stop the entire loop. The break keyword must be used inside any loop or a switch statement.

The break keyword will stop the execution of the innermost loop and start executing the next line of code after the block.

Syntax:

The syntax of a break is a single statement inside any loop:

break;

Example:

publicclassTest{

publicstaticvoid main(String args[]){

int[] numbers ={10,20,30,40,50};

for(int x : numbers ){

if( x ==30){

break;

}

System.out.print( x );

System.out.print("\n");

}

}

}

This would produce the following result:

10

20

The continue Keyword:

The *continue* keyword can be used in any of the loop control structures. It causes the loop to immediately jump to the next iteration of the loop.

* In a for loop, the continue keyword causes flow of control to immediately jump to the update statement.
* In a while loop or do/while loop, flow of control immediately jumps to the Boolean expression.

Syntax:

The syntax of a continue is a single statement inside any loop:

continue;

Example:

publicclassTest{

publicstaticvoid main(String args[]){

int[] numbers ={10,20,30,40,50};

for(int x : numbers ){

if( x ==30){

continue;

}

System.out.print( x );

System.out.print("\n");

}

}

}

This would produce the following result:

10

20

40

50

Example Programs:

* [Calculate Average value of Array elements using Java Example](http://www.java-examples.com/calculate-average-value-array-elements-using-java-example)
* [Declare multiple variables in for loop Example](http://www.java-examples.com/declare-multiple-variables-loop-example)
* [Fibonacci Series Java Example](http://www.java-examples.com/fibonacci-series-java-example)
* [Generate Pyramid For a Given Number Example](http://www.java-examples.com/generate-pyramid-given-number-example)
* [Infinite For loop Example](http://www.java-examples.com/infinite-loop-example)
* [Java Palindrome Number Example](http://www.java-examples.com/java-palindrome-number-example)
* [Java Pyramid 1 Example](http://www.java-examples.com/java-pyramid-1-example)
* [Java Pyramid 2 Example](http://www.java-examples.com/java-pyramid-2-example)
* [Java Pyramid 3 Example](http://www.java-examples.com/java-pyramid-3-example)
* [Java Pyramid 4 Example](http://www.java-examples.com/java-pyramid-4-example)
* [Java Pyramid 5 Example](http://www.java-examples.com/java-pyramid-5-example)
* [Java Pyramid 6 Example](http://www.java-examples.com/java-pyramid-6-example)
* [List Even Numbers Java Example](http://www.java-examples.com/list-even-numbers-java-example)
* [List Odd Numbers Java Example](http://www.java-examples.com/list-odd-numbers-java-example)
* [Prime Numbers Java Example](http://www.java-examples.com/prime-numbers-java-example)
* [Read Number from Console and Check if it is a Palindrome Number](http://www.java-examples.com/read-number-console-and-check-if-it-palindrome-number)
* [Simple For loop Example](http://www.java-examples.com/simple-loop-example)

# Calculate Average value of Array elements using Java Example

* /\*
* Calculate Average value of Array elements using Java Example
* This Java Example shows how to calculate average value of array
* elements.
* \*/
* public class CalculateArrayAverageExample {
* public static void main(String[] args) {
* //define an array
* int[] numbers = new int[]{10,20,15,25,16,60,100};
* /\*
* \* Average value of array elements would be
* \* sum of all elements/total number of elements
* \*/
* //calculate sum of all array elements
* int sum = 0;
* for(int i=0; i < numbers.length ; i++)
* sum = sum + numbers[i];
* //calculate average value
* double average = sum / numbers.length;
* System.out.println("Average value of array elements is : " + average);
* }
* }
* /\*
* Output of Calculate Average value of Array elements using Java Example would be
* Average value of array elements is : 35.0
* \*/

# Declare multiple variables in for loop Example

* /\*
* Declare multiple variables in for loop Example
* This Java Example shows how to declare multiple variables in Java For loop using
* declaration block.
* \*/
* public class DeclaringMultipleVariablesInForLoopExample {
* public static void main(String[] args) {
* /\*
* \* Multiple variables can be declared in declaration block of for loop.
* \*/
* for(int i=0, j=1, k=2 ; i<5 ; i++)
* System.out.println("I : " + i + ",j : "+ j + ", k : " + k);
* /\*
* \* Please note that the variables which are declared, should be of same type
* \* as in this example int.
* \*/
* //THIS WILL NOT COMPILE
* //for(int i=0, float j; i < 5; i++);
* }
* }

# Fibonacci Series Java Example

* /\*
* Fibonacci Series Java Example
* This Fibonacci Series Java Example shows how to create and print
* Fibonacci Series using Java.
* \*/
* public class JavaFibonacciSeriesExample {
* public static void main(String[] args) {
* //number of elements to generate in a series
* int limit = 20;
* long[] series = new long[limit];
* //create first 2 series elements
* series[0] = 0;
* series[1] = 1;
* //create the Fibonacci series and store it in an array
* for(int i=2; i < limit; i++){
* series[i] = series[i-1] + series[i-2];
* }
* //print the Fibonacci series numbers
* System.out.println("Fibonacci Series upto " + limit);
* for(int i=0; i< limit; i++){
* System.out.print(series[i] + " ");
* }
* }
* }
* /\*
* Output of the Fibonacci Series Java Example would be
* Fibonacci Series upto 20
* 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377 610 987 1597 2584 4181
* \*/

# Generate Pyramid For a Given Number Example

* */\**
* *Generate Pyramid For a Given Number Example*
* *This Java example shows how to generate a pyramid of numbers for given*
* *number using for loop example.*
* *\*/*
* **import** java.io.BufferedReader;
* **import** java.io.InputStreamReader;
* **public** **class** GeneratePyramidExample {
* **public** **static** **void** main (String[] args) **throws** Exception{
* BufferedReader keyboard = **new** BufferedReader (**new** InputStreamReader(System.in));
* System.out.println("Enter Number:");
* **int** as= Integer.parseInt (keyboard.readLine());
* System.out.println("Enter X:");
* **int** x=   Integer.parseInt (keyboard.readLine());
* **int** y = 0;
* **for**(**int** i=0; i<= as ;i++){
* **for**(**int** j=1; j <= i ; j++){
* System.out.print(y + "**\t**");
* y = y + x;
* }
* System.out.println("");
* }
* }
* }
* */\**
* *Output of this example would be*
* *Enter Number:*
* *5*
* *Enter X:*
* *1*
* *0*
* *1       2*
* *3       4       5*
* *6       7       8       9*
* *10      11      12      13      14*
* *----------------------------------------------*
* *Enter Number:*
* *5*
* *Enter X:*
* *2*
* *0*
* *2       4*
* *6       8       10*
* *12      14      16      18*
* *20      22      24      26      28*
* *----------------------------------------------*
* *Enter Number:*
* *5*
* *Enter X:*
* *3*
* *0*
* *3       6*
* *9       12      15*
* *18      21      24      27*
* *30      33      36      39      42*
* *\*/*

# Infinite For loop Example

* */\**
* *Infinite For loop Example*
* *This Java Example shows how to create a for loop that runs infinite times*
* *in Java program. It happens when the loop condition is always evaluated as true.*
* *\*/*
* **public** **class** InfiniteForLoop {
* **public** **static** **void** main(String[] args) {
* */\**
* *\* Its perfectely legal to skip any of the 3 parts of the for loop.*
* *\* Below given for loop will run infinite times.*
* *\*/*
* **for**(;;)
* System.out.println("Hello");
* */\**
* *\* To terminate this program press ctrl + c in the console.*
* *\*/*
* }
* }
* */\**
* *Output would be*
* *Hello*
* *Hello*
* *Hello*
* *Hello*
* *..*
* *..*
* *\*/*

# Java Palindrome Number Example

* */\**
* *Java Palindrome Number Example*
* *This Java Palindrome Number Example shows how to find if the given*
* *number is palindrome number or not.*
* *\*/*

* **public** **class** JavaPalindromeNumberExample {
* **public** **static** **void** main(String[] args) {
* *//array of numbers to be checked*
* **int** numbers[] = **new** **int**[]{121,13,34,11,22,54};
* *//iterate through the numbers*
* **for**(**int** i=0; i < numbers.length; i++){
* **int** number = numbers[i];
* **int** reversedNumber  = 0;
* **int** temp=0;
* */\**
* *\* If the number is equal to it's reversed number, then*
* *\* the given number is a palindrome number.*
* *\**
* *\* For example, 121 is a palindrome number while 12 is not.*
* *\*/*
* *//reverse the number*
* **while**(number > 0){
* temp = number % 10;
* number = number / 10;
* reversedNumber = reversedNumber \* 10 + temp;
* }
* **if**(numbers[i] == reversedNumber)
* System.out.println(numbers[i] + " is a palindrome number");
* **else**
* System.out.println(numbers[i] + " is not a palindrome number");
* }
* }
* }
* */\**
* *Output of Java Palindrome Number Example would be*
* *121 is a palindrome number*
* *13 is not a palindrome number*
* *34 is not a palindrome number*
* *11 is a palindrome number*
* *22 is a palindrome number*
* *54 is not a palindrome number*
* *\*/*

# Java Pyramid 1 Example

* */\**
* *Java Pyramid 1 Example*
* *This Java Pyramid example shows how to generate pyramid or triangle like*
* *given below using for loop.*
* *\**
* *\*\**
* *\*\*\**
* *\*\*\*\**
* *\*\*\*\*\**
* *\*/*
* **public** **class** JavaPyramid1 {
* **public** **static** **void** main(String[] args) {
* **for**(**int** i=1; i<= 5 ;i++){
* **for**(**int** j=0; j < i; j++){
* System.out.print("\*");
* }
* *//generate a new line*
* System.out.println("");
* }
* }
* }
* */\**
* *Output of the above program would be*
* *\**
* *\*\**
* *\*\*\**
* *\*\*\*\**
* *\*\*\*\*\**
* *\*/*

# Java Pyramid 2 Example

* */\**
* *Java Pyramid 2 Example*
* *This Java Pyramid example shows how to generate pyramid or triangle like*
* *given below using for loop.*
* *\*\*\*\*\**
* *\*\*\*\**
* *\*\*\**
* *\*\**
* *\**
* *\*/*
* **public** **class** JavaPyramid2 {
* **public** **static** **void** main(String[] args) {
* **for**(**int** i=5; i>0 ;i--){
* **for**(**int** j=0; j < i; j++){
* System.out.print("\*");
* }
* *//generate a new line*
* System.out.println("");
* }
* }
* }
* */\**
* *Output of the example would be*
* *\*\*\*\*\**
* *\*\*\*\**
* *\*\*\**
* *\*\**
* *\**
* *\*/*

# Java Pyramid 3 Example

* */\**
* *Java Pyramid 3 Example*
* *This Java Pyramid example shows how to generate pyramid or triangle like*
* *given below using for loop.*
* *\**
* *\*\**
* *\*\*\**
* *\*\*\*\**
* *\*\*\*\*\**
* *\*\*\*\*\**
* *\*\*\*\**
* *\*\*\**
* *\*\**
* *\**
* *\*/*
* **public** **class** JavaPyramid3 {
* **public** **static** **void** main(String[] args) {
* **for**(**int** i=1; i<= 5 ;i++){
* **for**(**int** j=0; j < i; j++){
* System.out.print("\*");
* }
* *//generate a new line*
* System.out.println("");
* }
* *//create second half of pyramid*
* **for**(**int** i=5; i>0 ;i--){
* **for**(**int** j=0; j < i; j++){
* System.out.print("\*");
* }
* *//generate a new line*
* System.out.println("");
* }
* }
* }
* */\**
* *Output of the example would be*
* *\**
* *\*\**
* *\*\*\**
* *\*\*\*\**
* *\*\*\*\*\**
* *\*\*\*\*\**
* *\*\*\*\**
* *\*\*\**
* *\*\**
* *\**
* *\*/*

# Java Pyramid 4 Example

* */\**
* *Java Pyramid 4 Example*
* *This Java Pyramid example shows how to generate pyramid or triangle like*
* *given below using for loop.*
* *1*
* *12*
* *123*
* *1234*
* *12345*
* *\*/*
* **public** **class** JavaPyramid4 {
* **public** **static** **void** main(String[] args) {
* **for**(**int** i=1; i<= 5 ;i++){
* **for**(**int** j=0; j < i; j++){
* System.out.print(j+1);
* }
* System.out.println("");
* }
* }
* }
* */\**
* *Output of the example would be*
* *1*
* *12*
* *123*
* *1234*
* *12345*
* *\*/*

# Java Pyramid 5 Example

* */\**
* *Java Pyramid 5 Example*
* *This Java Pyramid example shows how to generate pyramid or triangle like*
* *given below using for loop.*
* *12345*
* *1234*
* *123*
* *12*
* *1*
* *\*/*
* **public** **class** JavaPyramid5 {
* **public** **static** **void** main(String[] args) {
* **for**(**int** i=5; i>0 ;i--){
* **for**(**int** j=0; j < i; j++){
* System.out.print(j+1);
* }
* System.out.println("");
* }
* }
* }
* */\**
* *Output of the example would be*
* *12345*
* *1234*
* *123*
* *12*
* *1*
* *\*/*

# Java Pyramid 6 Example

* */\**
* *Java Pyramid 6 Example*
* *This Java Pyramid example shows how to generate pyramid or triangle like*
* *given below using for loop.*
* *\*\*\*\*\**
* *\*\*\*\**
* *\*\*\**
* *\*\**
* *\**
* *\**
* *\*\**
* *\*\*\**
* *\*\*\*\**
* *\*\*\*\*\**
* *\*/*
* **public** **class** JavaPyramid6 {
* **public** **static** **void** main(String[] args) {
* *//generate upper half of the pyramid*
* **for**(**int** i=5; i>0 ;i--){
* **for**(**int** j=0; j < i; j++){
* System.out.print("\*");
* }
* *//create a new line*
* System.out.println("");
* }
* *//generate bottom half of the pyramid*
* **for**(**int** i=1; i<= 5 ;i++){
* **for**(**int** j=0; j < i; j++){
* System.out.print("\*");
* }
* *//create a new line*
* System.out.println("");
* }
* }
* }
* */\**
* *Output of the example would be*
* *\*\*\*\*\**
* *\*\*\*\**
* *\*\*\**
* *\*\**
* *\**
* *\**
* *\*\**
* *\*\*\**
* *\*\*\*\**
* *\*\*\*\*\**
* *\*/*

# List Even Numbers Java Example

* */\**
* *List Even Numbers Java Example*
* *This List Even Numbers Java Example shows how to find and list even*
* *numbers between 1 and any given number.*
* *\*/*
* **public** **class** ListEvenNumbers {
* **public** **static** **void** main(String[] args) {
* *//define limit*
* **int** limit = 50;
* System.out.println("Printing Even numbers between 1 and " + limit);
* **for**(**int** i=1; i <= limit; i++){
* *// if the number is divisible by 2 then it is even*
* **if**( i % 2 == 0){
* System.out.print(i + " ");
* }
* }
* }
* }
* */\**
* *Output of List Even Numbers Java Example would be*
* *Printing Even numbers between 1 and 50*
* *2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44 46 48 50*
* *\*/*

# List Odd Numbers Java Example

* */\**
* *List Odd Numbers Java Example*
* *This List Odd Numbers Java Example shows how to find and list odd*
* *numbers between 1 and any given number.*
* *\*/*
* **public** **class** ListOddNumbers {
* **public** **static** **void** main(String[] args) {
* *//define the limit*
* **int** limit = 50;
* System.out.println("Printing Odd numbers between 1 and " + limit);
* **for**(**int** i=1; i <= limit; i++){
* *//if the number is not divisible by 2 then it is odd*
* **if**( i % 2 != 0){
* System.out.print(i + " ");
* }
* }
* }
* }
* */\**
* *Output of List Odd Numbers Java Example would be*
* *Printing Odd numbers between 1 and 50*
* *1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49*
* *\*/*

# Prime Numbers Java Example

* */\**
* *Prime Numbers Java Example*
* *This Prime Numbers Java example shows how to generate prime numbers*
* *between 1 and given number using for loop.*
* *\*/*
* **public** **class** GeneratePrimeNumbersExample {
* **public** **static** **void** main(String[] args) {
* *//define limit*
* **int** limit = 100;
* System.out.println("Prime numbers between 1 and " + limit);
* *//loop through the numbers one by one*
* **for**(**int** i=1; i < 100; i++){
* **boolean** isPrime = **true**;
* *//check to see if the number is prime*
* **for**(**int** j=2; j < i ; j++){
* **if**(i % j == 0){
* isPrime = **false**;
* **break**;
* }
* }
* *// print the number*
* **if**(isPrime)
* System.out.print(i + " ");
* }
* }
* }
* */\**
* *Output of Prime Numbers example would be*
* *Prime numbers between 1 and 100*
* *1 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97*
* *\*/*

# Read Number from Console and Check if it is a Palindrome Number

* */\**
* *Read Number from Console and Check if it is a Palindrome Number*
* *This Java example shows how to input the number from console and*
* *check if the number is a palindrome number or not.*
* *\*/*
* **import** java.io.BufferedReader;
* **import** java.io.IOException;
* **import** java.io.InputStreamReader;
* **public** **class** InputPalindromeNumberExample {
* **public** **static** **void** main(String[] args) {
* System.out.println("Enter the number to check..");
* **int** number = 0;
* **try**
* {
* *//take input from console*
* BufferedReader br = **new** BufferedReader(**new**InputStreamReader(System.in));
* *//parse the line into int*
* number = Integer.parseInt(br.readLine());
* }
* **catch**(NumberFormatException ne)
* {
* System.out.println("Invalid input: " + ne);
* System.exit(0);
* }
* **catch**(IOException ioe)
* {
* System.out.println("I/O Error: " + ioe);
* System.exit(0);
* }
* System.out.println("Number is " + number);
* **int** n = number;
* **int** reversedNumber  = 0;
* **int** temp=0;
* *//reverse the number*
* **while**(n > 0){
* temp = n % 10;
* n = n / 10;
* reversedNumber = reversedNumber \* 10 + temp;
* }
* */\**
* *\* if the number and it's reversed number are same, the number is a*
* *\* palindrome number*
* *\*/*
* **if**(number == reversedNumber)
* System.out.println(number + " is a palindrome number");
* **else**
* System.out.println(number + " is not a palindrome number");
* }
* }
* */\**
* *Output of the program would be*
* *Enter the number to check..*
* *121*
* *Number is 121*
* *121 is a palindrome number*
* *\*/*

# Simple For loop Example

* */\**
* *Simple For loop Example*
* *This Java Example shows how to use for loop to iterate in Java program.*
* *\*/*
* **public** **class** SimpleForLoopExample {
* **public** **static** **void** main(String[] args) {
* */\* Syntax of for loop is*
* *\**
* *\* for(<initialization> ; <condition> ; <expression> )*
* *\*   <loop body>*
* *\**
* *\* where initialization usually declares a loop variable, condition is a*
* *\* boolean expression such that if the condition is true, loop body will be*
* *\* executed and after each iteration of loop body, expression is executed which*
* *\* usually increase or decrease loop variable.*
* *\**
* *\* Initialization is executed only once.*
* *\*/*
* **for**(**int** index = 0; index < 5 ; index++)
* System.out.println("Index is : " + index);
* */\**
* *\* Loop body may contains more than one statement. In that case they should*
* *\* be in the block.*
* *\*/*
* **for**(**int** index=0; index < 5 ; index++)
* {
* System.out.println("Index is : " + index);
* index++;
* }
* */\**
* *\* Please note that in above loop, index is a local variable whose scope*
* *\* is limited to the loop. It can not be referenced from outside the loop.*
* *\*/*
* }
* }
* */\**
* *Output would be*
* *Index is : 0*
* *Index is : 1*
* *Index is : 2*
* *Index is : 3*
* *Index is : 4*
* *Index is : 0*
* *Index is : 2*
* *Index is : 4*
* *\*/*

Assignments:

Write programs to print following patterns on console.

Enter Your Name: GOOGLE  
G  
GO  
GOO  
GOOG  
GOOGL  
GOOGLE

1  
12  
123  
1234  
12345  
123456

12345  
1234  
123  
12  
1

\*  
\*\*  
\*\*\*  
\*\*\*\*  
\*\*\*\*\*  
\*\*\*\*\*\*  
\*\*\*\*\*\*\*

12345654321  
12345 54321  
1234 4321  
123 321  
12 21  
1 1

543210  
43210  
3210  
210  
10  
0

5 4 3 2 1  
5 4 3 2  
5 4 3  
5 4  
5

999999999  
7777777  
55555  
333  
1