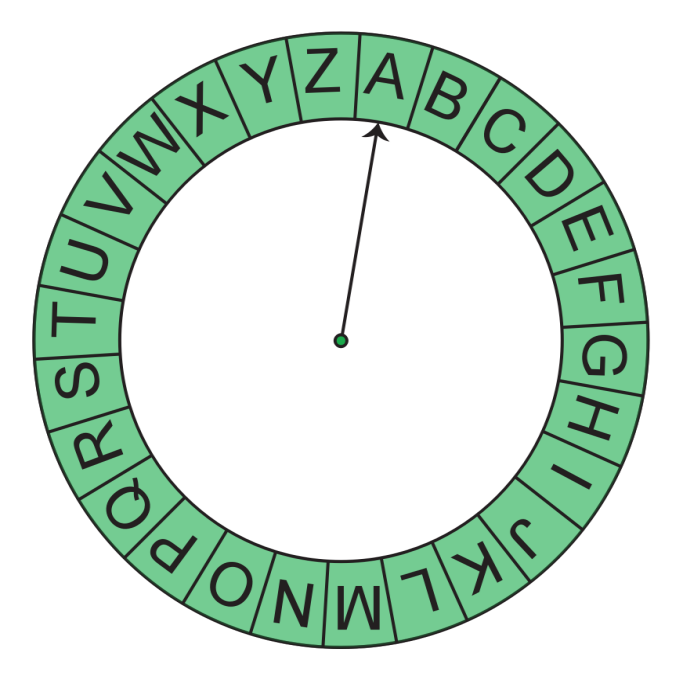
Circular Printer

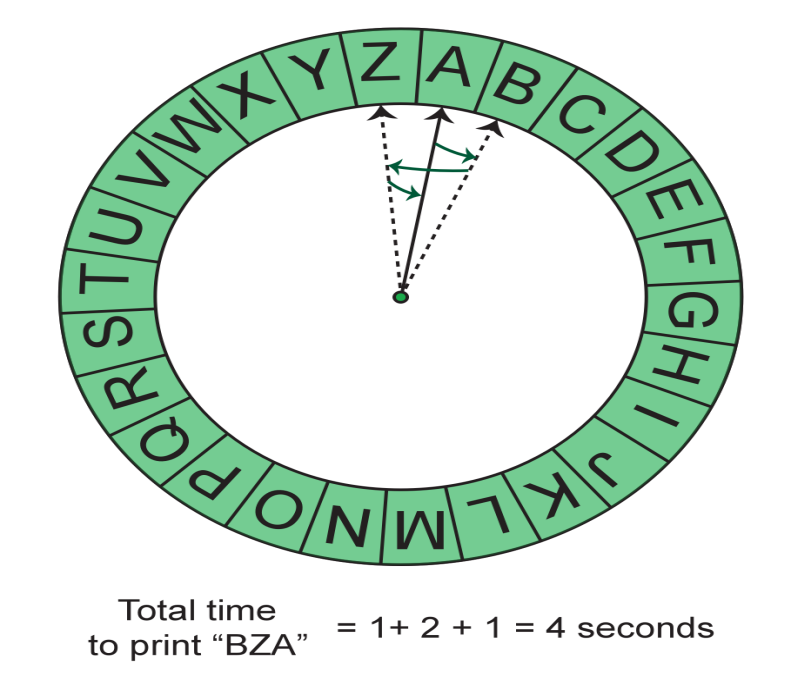
A company has invented a new type of printing technology—a circular printer that looks like this:



It is a circular printer wheel with the letters A through Z in sequence.  It wraps so A and Z are adjacent. The printer has a pointer that is initially at 'A'. Moving from any character to any adjacent character takes 1 second. It can move in either direction. Given a string of letters, what is the minimum time needed to print the string? (Note: Assume that printing does not take any time. Only consider the time it takes for the pointer to move.)

****Example****

*s = "BZA"*



First, move the pointer from 'A' to 'B' (1 second), then from 'B' to 'Z' (2 seconds), and finally from 'Z' to 'A' (1 second). So the minimum time needed to print "BZA" is 4 seconds.

**Function Description**

Complete the function *getTime* in the editor below.

getTime has the following parameter:

    string *s:*  the string of characters that need to be printed

Returns:

    int: the minimum number of seconds needed to print *s*

**Constraints**

1 ≤ length of *s* ≤ 10^5

**Input Format For Custom Testing**

The first line contains a string, *s*, the string to be printed.

**Sample Case 0**

**Sample Input For Custom Testing**

STDIN     Function

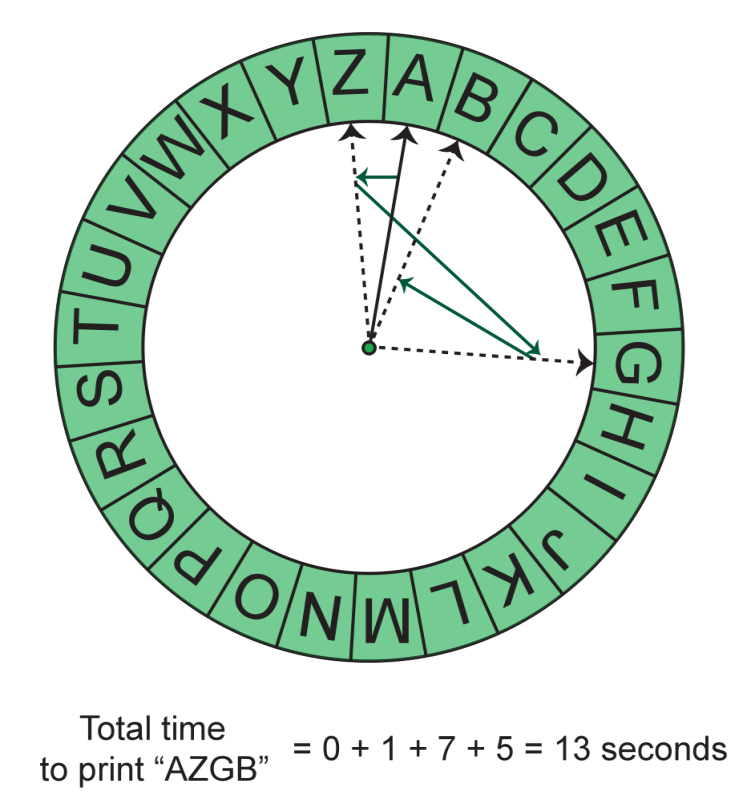
-----     --------

AZGB => s = "AZGB"

**Sample Output**

13

**Explanation**



Initially, the pointer is at 'A', so there is no need to move it for the first character. First, move the pointer from 'A' to 'Z' (1 second), then from 'Z' to 'G' (7 seconds), and finally from 'G' to 'B' (5 seconds). Therefore, the total time is 1 + 7 + 5 = 13.

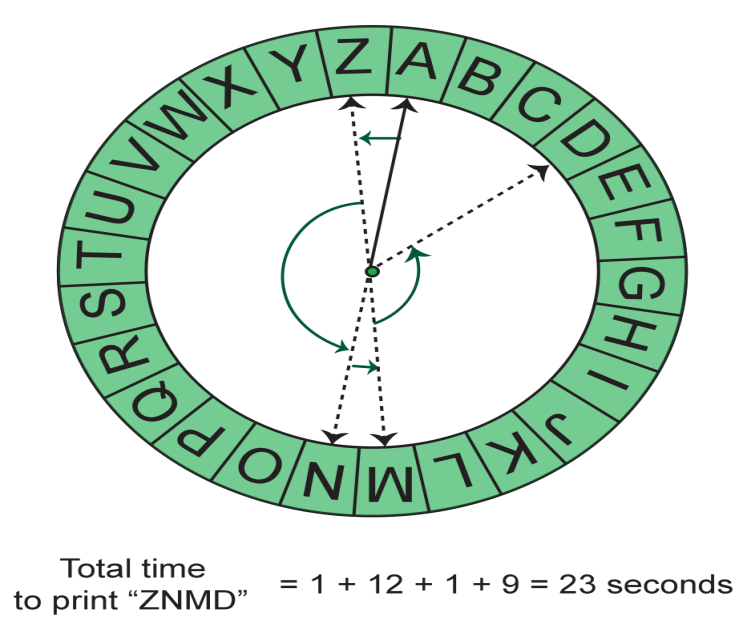
**Sample Case 1**

**Sample Input For Custom Testing**

ZNMD

**Sample Output**

23

**Explanation**

First, move the pointer from 'A' to 'Z' (1 second), then from 'Z' to 'N' (12 seconds), then from 'N' to 'M' (1 second), and finally from 'M' to 'D' (9 seconds). The total time is 1 + 12 + 1 + 9 = 23.

import java.io.\*;

import java.math.\*;

import java.security.\*;

import java.text.\*;

import java.util.\*;

import java.util.concurrent.\*;

import java.util.function.\*;

import java.util.regex.\*;

import java.util.stream.\*;

import static java.util.stream.Collectors.joining;

import static java.util.stream.Collectors.toList;

class Result {

/\*

\* Complete the 'getTime' function below.

\*

\* The function is expected to return a LONG\_INTEGER.

\* The function accepts STRING s as parameter.

\*/

public static long getTime(String s) {

// Write your code here

}

}

public class Solution {

public static void main(String[] args) throws IOException {

BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));

BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT\_PATH")));

String s = bufferedReader.readLine();

long result = Result.getTime(s);

bufferedWriter.write(String.valueOf(result));

bufferedWriter.newLine();

bufferedReader.close();

bufferedWriter.close();

}

}

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