String Reduction **EASY**

Given a string, reduce it in such a way that all of its substrings are distinct. To do so, you may delete any characters at any index. What is the minimum number of deletions needed?

****Note:**** A *substring* is a contiguous group of 1 or more characters within a string.

****Example****

*s = "abab"*

Substrings in *s* are { 'a', 'b', 'a', 'b', 'ab', 'ba', 'ab', 'aba', 'bab', 'abab'}. By deleting one "a" and one "b", the string becomes "ab" or "ba" and all of its substrings are distinct. This required 2 deletions.

**Function Description**

Complete the function *getMinDeletions* in the editor below.

getMinDeletions has the following parameter(s):

    string *s:*  the given string

Returns:

    int: the minimum number of deletions required

**Constraints**

1 ≤ *n* ≤ 105

窗体顶端

**Input Format For Custom Testing**

The first line contains a string, *s*.

**Sample Case 0**

**Sample Input For Custom Testing**

STDIN     Function

-----     --------

abcab → s = "abcab"

**Sample Output**

2

**Explanation**

By deleting the first 2 characters, the string becomes "cab", which contains only distinct substrings.

**Sample Case 1**

**Sample Input For Custom Testing**

STDIN     Function

-----     --------

abcabc →  s = "abcabc"

**Sample Output**

3

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 'getMinDeletions' function below.

\*

\* The function is expected to return an INTEGER.

\* The function accepts STRING s as parameter.

\*/

public static int getMinDeletions(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();

int result = Result.getMinDeletions(s);

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

bufferedWriter.newLine();

bufferedReader.close();

bufferedWriter.close();

}

}

**Test Cases:**

**Testcase 1: *Success***

**Input [[](https://istreet-questions-us-east-1.s3.amazonaws.com/757898/input000.txt?response-content-disposition=attachment; filename=input000.txt&response-content-type=application/octet-stream&versionId=qcExmRtTnYB9v4h6bo4wKuytojFi.7yC&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAR6O7GJNX5DNFO3PV/20211118/us-east-1/s3/aws4_request&X-Amz-Date=20211118T142232Z&X-Amz-Expires=600&X-Amz-SignedHeaders=host&X-Amz-Signature=f2f1c741e604a2333860f073c6393b50cdad5af13b7acf07118f4c4e907394a6" \o "Download Input)**

abcab

**Your Output**

2

**Expected Output [[](https://istreet-questions-us-east-1.s3.amazonaws.com/757898/output000.txt?response-content-disposition=attachment; filename=output000.txt&response-content-type=application/octet-stream&versionId=ZeNjekOKDOB8YMarjARVhBr2tfzqdyaj&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAR6O7GJNX5DNFO3PV/20211118/us-east-1/s3/aws4_request&X-Amz-Date=20211118T142232Z&X-Amz-Expires=600&X-Amz-SignedHeaders=host&X-Amz-Signature=c192fe1c5031f8628fe3ad5e33579bdbc293af3ac0abc4778dc6db9f8f4c35c9" \o "Download Output)**

2

**Testcase 2: *Success***

**Input**

abcabc

**Your Output**

3

**Expected Output [[](https://istreet-questions-us-east-1.s3.amazonaws.com/757898/output001.txt?response-content-disposition=attachment; filename=output001.txt&response-content-type=application/octet-stream&versionId=7PxYlDktNlNV7U_6RBLkkc4nI0X.NDh7&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAR6O7GJNX5DNFO3PV/20211118/us-east-1/s3/aws4_request&X-Amz-Date=20211118T142232Z&X-Amz-Expires=600&X-Amz-SignedHeaders=host&X-Amz-Signature=2011f6b3901f0dbcc0eaf7bb4a96ca8f6b8f77afdb776d12c7fc62528a3ea96c" \o "Download Output)**

3

**Testcase 3: *Success***

**Input [[](https://istreet-questions-us-east-1.s3.amazonaws.com/757898/input002.txt?response-content-disposition=attachment; filename=input002.txt&response-content-type=application/octet-stream&versionId=8FsTZbyHrEVEL01uzUY2FUXPR9kiSbTE&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAR6O7GJNX5DNFO3PV/20211118/us-east-1/s3/aws4_request&X-Amz-Date=20211118T142232Z&X-Amz-Expires=600&X-Amz-SignedHeaders=host&X-Amz-Signature=d8ff655b5beb1918ad59e0fdc463792ac12e2a4f16c630a93aa2ff44a1132eac" \o "Download Input)**

bbeadcebfp

**Your Output**

3

**Expected Output**

3

**Testcase 4: *Success***

**Input [[](https://istreet-questions-us-east-1.s3.amazonaws.com/757898/input003.txt?response-content-disposition=attachment; filename=input003.txt&response-content-type=application/octet-stream&versionId=cyXUs.qs2AwPuMSqgNH0HS73GMhLz2a8&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAR6O7GJNX5DNFO3PV/20211118/us-east-1/s3/aws4_request&X-Amz-Date=20211118T142232Z&X-Amz-Expires=600&X-Amz-SignedHeaders=host&X-Amz-Signature=e5944816620cb87885b3fe003d171b29a3302b9ebcf5f92a8510f5913c1e61eb" \o "Download Input)**

bfadhbhoeg

**Your Output**

2

**Expected Output [[](https://istreet-questions-us-east-1.s3.amazonaws.com/757898/output003.txt?response-content-disposition=attachment; filename=output003.txt&response-content-type=application/octet-stream&versionId=KovOy_7_xDIh4TcgLbofeJMkC2pr0mY.&X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=AKIAR6O7GJNX5DNFO3PV/20211118/us-east-1/s3/aws4_request&X-Amz-Date=20211118T142232Z&X-Amz-Expires=600&X-Amz-SignedHeaders=host&X-Amz-Signature=8f6dcd5fadb6447de500a45f6e22021692ce90a66106e437514e5cf73d4a64d1" \o "Download Output)**

2

**Testcase 5: *Success***

**Input**

ffcccdndha

**Your Output**

4

**Expected Output**

4

**Testcase 6: *Success***

**Input**

afcbdgigdrfacaafgbgeecadajadadebbofbdcaaidacacbdflbcfdbgaaidoaaafacaaiccdafhkcgaeeffebchbbaebfhiciabeaaabbmaefbagaabcafccbbaqdefaggddnjdafdibbeahjcjgebeddbababdgaegcbdcfbgabfhccecdibdccchbhbaccaebcbadbcecefcefcbdildkebngadeekasbbgfcafcfhbcafaaibcbcbickaaedafiaedbbccgpdedaccccclbggcoccaepbdaadgdfdbdbdfacbcacdaaabafadcibcbcbeehbffagibbabbcagbccbdcdddccdcdeahaafamngadgbcaaceeheabfcbedcfbbcbfacdagdaboddagfbaeagcbbclbcibcibhejceabdacbejfcdfbebccaggbcabdgiglcceaagaacchddacicacacgcbdffbogafcaacigdgcafcagbcaccbabecacahcdcgbhbgdcdcedagbaibbdadbkhcahbeaaaacbacagbaagbjcfaeadboakcaiacdaggfdgchacdjjpbalhadbbcfiaacdbeaidbeccebgajbifacblbbbcccebbemfancibbcdcdbbbadmlioqkccchcaiebacdaiddedgdbhifaccfaaaaebdffbbgdcbdefbhaabecdahfdjbfadaaabffdghhlabbabeaqdflkfcbbaaeeaeamocagaadabcdheafihbbgjhameiagdcjaahkadcibaafbacaadodakkdacgbdbffdbjmbbbafgddebihcaaaaeahhbhbbabhhdfdefcfeecadbgagicbebcambebcdeaaakgnbaacaafgfjagbbdeebgebdgbabdadgageachajedabcfekbebccgaacbfbfddbhaafdgmachacgdadaebgbadlbacjddbcbcaccbbjancdd

**Your Output**

981

**Expected Output**

981

**Testcase 7: *Success***

**Input**

jabideicafekegfibeeadecabbbgdhdaeccaebaaccffcaeebaiafdabaaahbbbaaccbcbggabaebbibdadhaacccabadedagcblbcagigadabceheccdajdbaaidacbdaakcaiadiaaicgagcbaekafbajgcaafldcaddebbahdbgbdaagaccidbbcgciaccdajcfacbfcceagaadglfaabdacdbfdadccafabaiccbfaejcaaaaciafeekadmacajicfjedgabaafdceecaeadeaacadajbacdcbcabbcddccgabafagdaccagfbeacdgfbbddnfbfhcbicbahceaifafbddeakcbcbiadabbgacejbcbceaadbicddaacbkgeahaaeebecbegfbdfifdicbaaiadabqecfbkfaeggfbdcdqbdffbaanadfabcaagbbhagccbdejidafcbddigiadbegbbaahabdhfcaogfddiedaacbcfechacfbcahadaacegkdlebcheanbacdcgeabmdcgdacaaigmacalaiabbbcccjfbacebekccipiebabdcfadaaeachebgdabiabhabajbacaambahcnaadaebebabehibbbfbdfdcafjcbaacmeedabaaeedcjbbbcacaceaaafddababijeadijbgaddgfdcaaedbbfabpadiebaeaiacbgaagfakdbkfcadbcddceaigedbbdeacdfdahdcebgbcbjbadcdfefefbdaaceegbabbdebbbbbebcaddbbkdgbacbabicbbdcacebcaebchachecbcbadkaddacfcdedjdaaegdcbbbbkbeagbbbcaebbiacbjdbcecbcafebfbgebcacadbkecbafaaceabdhcacdaahiegeaacdcaeebfbcaidafafchcafcfdbgahbaddechaafaefhgcabbfbdfaeacacaahbfecfebacaaaq

**Your Output**

983

**Expected Output**

983

**Testcase 8: *Success***

**Input**

abbbafjdcbefcbafgffbdaegclcccdkaafebchdbbacebjaabidcicaaaebaebaeaeccecakdbdajchbgaceabgaacbaacdcaefbccagccaacaagebgbaecechbfegiaaachccacihcedccifabafcebabifaacghafgccajefaabdabcbaacfjciddacchabcbabcbifbgifacaahdaabiegbbbbabekdabaacbdfabadbbaabkdhgccacfbbcaafcbcebbaacfaidbbmdfabcafcfdcaaaaeabecaddhgbebfbacahegdiaigafabagbedbbjcdaadacadcnbbhebggabmbbabaaeaaacaaabcccabecdaaacaebbabaccbbaaebdcdajddabbfacbdaaaaaacadceaacbeeebgfabchcecdcacaabbaadkbcbbdaabaabaeeabadcchggaebmdbcbbcadceaafeaajjdaabbdieaabhadcbngecaabbajagebebdbbcgagbbabcbbaaaeadbjabaaaadeacafabacacgahdeaedaidbbaaaafcbbcmabiadigceacbcbfchfbebbcaaebaffcfebcadbhgfededbbfdabacfdbefbccgaigbbbcjeaaaeaddcbabdeccdbabebgbdbabcfjbekcegajbebcbeehhbcgacbaedhambfbacjacheadaecchbddfcgaedbcabeadddaabbbbeeeadcejdacdcaadbbaahddbcaaaadddbccagacgchadhhaadladdjadcbaafkccaagaabaedhbbbfbcadgaabdhfadccbbdaaaakdaeeeabbcihaabdcdbkcfadfcaaefaabhcaiacacbgbbfdcabcccabdaaeeebahcjacbagbcaebadaafbbabdkjaaadbeccagfafbcbacedbgcbgbcbhbaafdbbheachaacbbfababefdae

**Your Output**

986

**Expected Output**

986

**Testcase 9: *Success***

**Input**

dbcchcefaaahaaieadagbfcacecbeacabcbcabcbaceccfcaabafijaebdbdagacbbadahaaabaaadabacaabdaaaaaacabbcadebccbfagacaaecafbbdaaeceaacafbabadaccbbeafbkgcaaabaabbbccdbahbaaaaabdekalaacaababbfddabeaagccaaaeedaadcaebcbgadfdabfbckedbbbeabaadaacicaaaccakacabccabeabbcecadddcjbaefabbaebbfaaeegadafaajcfedbbcfccebbmaddcabbidfaaccaabadiabdbbebbabaaabcacbbddbgaecaaadcdabbdbcchcaccdadcbcabeddacadbdafibccdaeaajgbafbcdaeabjecaaabefababbcaacbbacbbbeahabaedibabbceagcafdbadccdahabebafabfeacaalfaiaadakacbccbiadaddaaacbacbaaabaccfacbdabjacdgcbbcceabbdaibcbaedabgabacbccafbabaabadaacccaceahbaciadabbaacccadhfacdabaaabgddbadaadcbfalbbabaaafhdbcaabbaddaddaabddcahagcbaaaageehdbcdbceabaabcbccbkeddacbbeaaaabhaaafaaaefccdbbbbkfhdeciccbadaabacecabcbbcaagaaeaagbdeadifhgcedcbfbbdcbaebfafbkcaadaebbabccbaaafbaadbggagaacbafbadcabccbabaaddbacabeababbhfbccccaabeaaddacccbbbcacaaceciadaabbeaccegabfbaaabgadbabaabkdbbfaaafbbadabcbacfdjeabmbbdblaafkaaaheacbacacdabhkfebjfgababafabfcgbabcaecbbbaiaaadheaacdcfaagcdcaeaacaidaaaafabaaabdib

**Your Output**

987

**Expected Output**

987

**Testcase 10: *Success***

**Input**

acfaabacahadaadabcagabccbdaacbkbfhccbabeabcaabacdacebfaceaaaacadaaacbciaaaaeaaaababcbbdbbabcfaaaabbbaccaadaabahdaabbafcaifccaedddaccjccaafcbchbaeabdhacedcdabadbcbaccababbdaeaacacaaecaacibcadacdaafdeebcbeeackcdaabbebabagfaaacacgadabbbcbhbadbbddcabaaacacebffcababeceaibbcaajaabbbebcjababbdbeaacacfcaacaccacaaaabbdcaiabddfabdbbagaeaababcagbccagagccdcbaabgbdcbaaadebecabcaaebbbbgaeecbbaaabbaajdaacaaacbgcbiaabgbabdblabafbadacabdbbaaafecedcaeaccbababdebiaaeeadaccacaaacgmbaddbaaaaacbgababbaeadabedbaabaaacaccacaaaaeaafabaedaacdbbccaceadcabaabbcajgbcafadcaabacbachabacaaeacdbcbacaaaaacagacbbabfeaabebbbbbbbbdbbbbaagccaaccaaaabbaabbbebbfebgabcaafcaegeccebgcebagceacccebadbaaaababdccdcabaadbcafeadaaafaieeacbcbabaaacgeakaajccgaagabccacgadddedfaabbaaabadbdbbebeadcabgcbbccaadbdacbadaceeaeeackeacdadcaabcdcbbcbcccabcbdaaaefbbafaaafabdebeafngbdabcebcdbbabcahbabdahbfaaacbafacaadafbhbdbdbbbadcgeabbbcabadacbaeedbdcbeaaaehcacagccbbaaaaecabaeebfacbbcbaabhacabgabcdeccbfbdbbbhaecdcbcabbabcdeddchdadabcaadbcdffbiadaa

**Your Output**

986

**Expected Output**

986

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