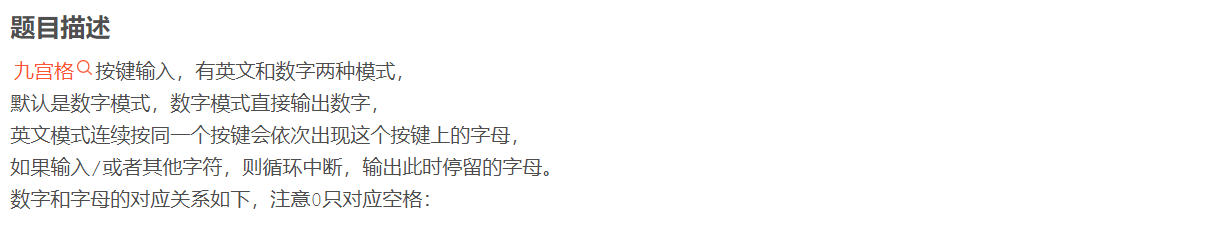
# **E卷-九宫格按键输入[200分]（ Java | Python3 | C++ | C语言 | JsNode | Go ）**

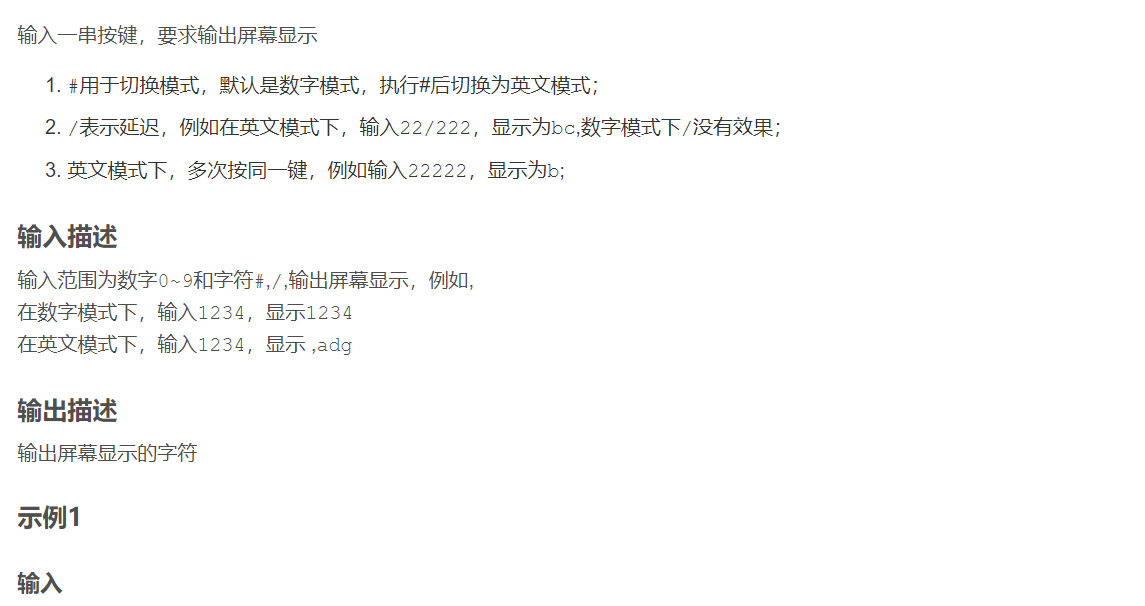


1(,.) 2(abc) 3(def)

4(ghi) 5(jkl) 6(mno)

7(pqrs) 8(tuv) 9(wxyz)

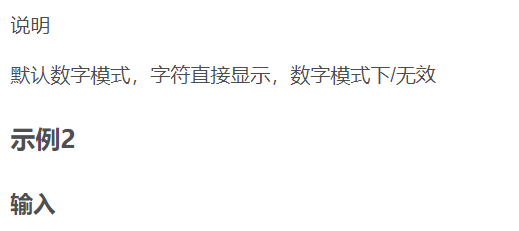
# 0(空格) /



2222/22



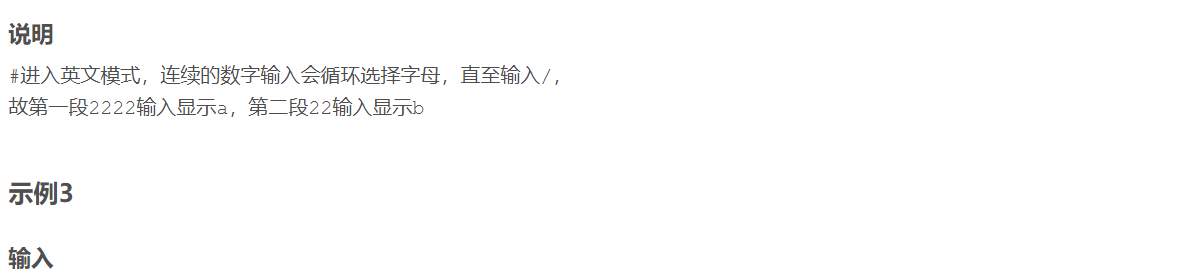
222222



#2222/22



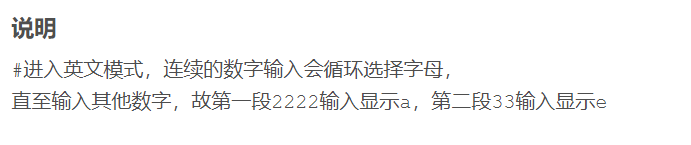
Ab



#222233



Ae





import java.util.Scanner;

public class Main {

// 定义手机键盘按键对应的字符表，mp[i] 代表按键 i 对应的字符

static String[] mp = {" ", ",.", "abc", "def", "ghi", "jkl", "mno", "pqrs", "tuv", "wxyz"};

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// 读取输入字符串

String line = scanner.nextLine();

// 初始化mod变量为0，用于模式切换

int mod = 0;

// 循环处理输入字符串中以 '#' 开头的部分

while (line.length() > 0 && line.charAt(0) == '#') {

// mod异或1，用于切换模式

mod ^= 1;

// 删除字符串的第一个字符（即去掉前面的#）

line = line.substring(1);

}

// 将剩余的字符串按 '#' 分割成数组 a

String[] a = line.split("#");

// 初始化结果字符串 res

StringBuilder res = new StringBuilder();

// 遍历分割后的数组 a，分别根据当前模式处理各个部分

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

// 如果当前部分为空字符串，则切换模式

if (a[i].length() == 0) {

mod ^= 1;

continue;

}

// 非第一部分且当前部分不为空时，切换模式

if (i > 0 && a[i].length() != 0) {

mod ^= 1;

}

if (mod == 0) {

// 如果当前是数字模式

res.append(num\_mod(a[i]));

} else {

// 如果当前是字母模式

res.append(eng\_mod(a[i]));

}

}

// 输出处理后的结果

System.out.println(res.toString());

}

// 定义处理数字模式的函数 num\_mod，将字符串中不是 '/' 的字符拼接起来

public static String num\_mod(String s) {

StringBuilder res = new StringBuilder(); // 存放结果

for (char c : s.toCharArray()) {

if (c == '/') { // 忽略 '/' 字符

continue;

}

res.append(c); // 直接将数字加入结果

}

return res.toString(); // 返回拼接后的结果

}

// 定义处理字母模式的函数 eng\_mod，将按键的数字组合转换为对应字母

public static String eng\_mod(String s) {

StringBuilder res = new StringBuilder(); // 存放最终结果

StringBuilder t = new StringBuilder(); // 临时保存部分结果

t.append(s.charAt(0)); // 初始化t为字符串的第一个字符

// 遍历字符串的其余字符

for (int i = 1; i < s.length(); i++) {

// 如果当前字符与前一个字符不同，则在中间插入 '/'

if (s.charAt(i) != s.charAt(i - 1)) {

t.append("/");

}

t.append(s.charAt(i)); // 将当前字符加入 t 中

}

// 按 '/' 分割 t，得到各个按键分组

String[] tmp = t.toString().split("/");

// 遍历每个按键分组，转换成对应字母

for (String str : tmp) {

if (str.length() == 0) { // 忽略空字符串

continue;

}

// 获取对应按键的字母集合

String x = mp[Character.getNumericValue(str.charAt(0))];

// 根据按键的按次数计算字符索引

int idx = str.length() % x.length();

idx = (idx - 1 + x.length()) % x.length(); // 调整索引以防负数

res.append(x.charAt(idx)); // 添加对应字母到结果字符串

}

return res.toString(); // 返回字母转换后的结果

}

}



# 读取输入字符串

line = input()

# 初始化mod变量为0，用于模式切换

mod = 0

# 循环处理输入字符串中以 '#' 开头的部分

while len(line) and line[0] == '#':

# mod异或1，用于切换模式

mod ^= 1

# 删除字符串的第一个字符（即去掉前面的#）

line = line[1:]

# 将剩余的字符串按 '#' 分割成数组 a

a = line.split("#")

# 定义手机键盘按键对应的字符表，mp[i] 代表按键 i 对应的字符

mp = [" ", ",.", "abc", "def", "ghi", "jkl", "mno", "pqrs", "tuv", "wxyz"]

# 定义处理数字模式的函数 num\_mod，将字符串中不是 '/' 的字符拼接起来

def num\_mod(s):

res = "" # 存放结果

for i in s:

if i == '/': # 忽略 '/' 字符

continue

res += i # 直接将数字加入结果

return res # 返回拼接后的结果

# 定义处理字母模式的函数 eng\_mod，将按键的数字组合转换为对应字母

def eng\_mod(s):

res = "" # 存放最终结果

t = s[0] # 初始化t为字符串的第一个字符

# 遍历字符串的其余字符

for i in range(1, len(s)):

# 如果当前字符与前一个字符不同，则在中间插入 '/'

if s[i] != s[i-1]:

t += "/"

t += s[i] # 将当前字符加入 t 中

# 按 '/' 分割 t，得到各个按键分组

tmp = t.split("/")

# 遍历每个按键分组，转换成对应字母

for i in range(len(tmp)):

if not len(tmp[i]): # 忽略空字符串

continue

# 获取对应按键的字母集合

x = mp[int(tmp[i][0])]

# 根据按键的按次数计算字符索引

idx = len(tmp[i]) % len(x)

idx = (idx - 1 + len(x)) % len(x) # 调整索引以防负数

res += x[idx] # 添加对应字母到结果字符串

return res # 返回字母转换后的结果

# 初始化结果字符串 res

res = ""

# 遍历分割后的数组 a，分别根据当前模式处理各个部分

for i in range(len(a)):

if not len(a[i]): # 如果当前部分为空字符串，则切换模式

mod ^= 1

continue

if i and len(a[i]): # 非第一部分且当前部分不为空时，切换模式

mod ^= 1

if not mod: # 如果当前是数字模式

res += num\_mod(a[i]) # 使用 num\_mod 处理

else: # 如果当前是字母模式

res += eng\_mod(a[i]) # 使用 eng\_mod 处理

# 输出处理后的结果

print(res)

# 示例输入：

# #222233

# 输出：ae

# ###233232/##2352##6336###3523235276899677568568006785673#0/3/32423/222233

# 输出：aeadaadjamem3523235276899677568568006785673 ddagadae



#include <iostream>

#include <vector>

#include <string>

using namespace std;

// 定义手机键盘按键对应的字符表，mp[i] 代表按键 i 对应的字符

vector<string> mp = {" ", ",.", "abc", "def", "ghi", "jkl", "mno", "pqrs", "tuv", "wxyz"};

// 定义处理数字模式的函数 num\_mod，将字符串中不是 '/' 的字符拼接起来

string num\_mod(string s) {

string res; // 存放结果

for (char c : s) {

if (c == '/') { // 忽略 '/' 字符

continue;

}

res += c; // 直接将数字加入结果

}

return res; // 返回拼接后的结果

}

// 定义处理字母模式的函数 eng\_mod，将按键的数字组合转换为对应字母

string eng\_mod(string s) {

string res; // 存放最终结果

string t;

t += s[0]; // 初始化t为字符串的第一个字符

// 遍历字符串的其余字符

for (int i = 1; i < s.size(); i++) {

// 如果当前字符与前一个字符不同，则在中间插入 '/'

if (s[i] != s[i - 1]) {

t += "/";

}

t += s[i]; // 将当前字符加入 t 中

}

// 按 '/' 分割 t，得到各个按键分组

vector<string> tmp;

string tempStr;

for (char c : t) {

if (c == '/') {

tmp.push\_back(tempStr);

tempStr = "";

} else {

tempStr += c;

}

}

tmp.push\_back(tempStr);

// 遍历每个按键分组，转换成对应字母

for (string str : tmp) {

if (str.empty()) { // 忽略空字符串

continue;

}

// 获取对应按键的字母集合

string x = mp[str[0] - '0'];

// 根据按键的按次数计算字符索引

int idx = str.size() % x.size();

idx = (idx - 1 + x.size()) % x.size(); // 调整索引以防负数

res += x[idx]; // 添加对应字母到结果字符串

}

return res; // 返回字母转换后的结果

}

int main() {

// 读取输入字符串

string line;

getline(cin, line);

// 初始化mod变量为0，用于模式切换

int mod = 0;

// 循环处理输入字符串中以 '#' 开头的部分

while (!line.empty() && line[0] == '#') {

// mod异或1，用于切换模式

mod ^= 1;

// 删除字符串的第一个字符（即去掉前面的#）

line = line.substr(1);

}

// 将剩余的字符串按 '#' 分割成数组 a

vector<string> a;

string tempStr;

for (char c : line) {

if (c == '#') {

a.push\_back(tempStr);

tempStr = "";

} else {

tempStr += c;

}

}

a.push\_back(tempStr);

// 初始化结果字符串 res

string res;

// 遍历分割后的数组 a，分别根据当前模式处理各个部分

for (int i = 0; i < a.size(); i++) {

// 如果当前部分为空字符串，则切换模式

if (a[i].empty()) {

mod ^= 1;

continue;

}

// 非第一部分且当前部分不为空时，切换模式

if (i > 0 && !a[i].empty()) {

mod ^= 1;

}

if (mod == 0) {

// 如果当前是数字模式

res += num\_mod(a[i]);

} else {

// 如果当前是字母模式

res += eng\_mod(a[i]);

}

}

// 输出处理后的结果

cout << res << endl;

return 0;

}



#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#define MAX\_LEN 1000

// 定义手机键盘按键对应的字符表，mp[i] 代表按键 i 对应的字符

char\* mp[] = {" ", ",.", "abc", "def", "ghi", "jkl", "mno", "pqrs", "tuv", "wxyz"};

// 定义处理数字模式的函数 num\_mod，将字符串中不是 '/' 的字符拼接起来

void num\_mod(char\* s, char\* res) {

int len = strlen(s);

int j = 0; // res 的指针

for (int i = 0; i < len; i++) {

if (s[i] == '/') { // 忽略 '/' 字符

continue;

}

res[j++] = s[i]; // 直接将数字加入结果

}

res[j] = '\0'; // 终止符号

}

// 定义处理字母模式的函数 eng\_mod，将按键的数字组合转换为对应字母

void eng\_mod(char\* s, char\* res) {

int len = strlen(s);

char t[MAX\_LEN \* 2] = "";

t[0] = s[0]; // 初始化 t 为字符串的第一个字符

int j = 1; // t 的指针

// 遍历字符串的其余字符

for (int i = 1; i < len; i++) {

// 如果当前字符与前一个字符不同，则在中间插入 '/'

if (s[i] != s[i - 1]) {

t[j++] = '/';

}

t[j++] = s[i]; // 将当前字符加入 t 中

}

t[j] = '\0'; // 终止符号

// 按 '/' 分割 t，得到各个按键分组

char\* tmp = strtok(t, "/");

while (tmp != NULL) {

int firstNum = tmp[0] - '0';

char\* x = mp[firstNum];

int idx = strlen(tmp) % strlen(x);

idx = (idx - 1 + strlen(x)) % strlen(x); // 调整索引以防负数

strncat(res, &x[idx], 1); // 添加对应字母到结果字符串

tmp = strtok(NULL, "/");

}

}

int main() {

char line[MAX\_LEN];

// 读取输入字符串

fgets(line, MAX\_LEN, stdin);

line[strcspn(line, "\n")] = 0; // 去除换行符

// 初始化 mod 变量为0，用于模式切换

int mod = 0;

// 循环处理输入字符串中以 '#' 开头的部分

while (strlen(line) > 0 && line[0] == '#') {

// mod 异或1，用于切换模式

mod ^= 1;

// 删除字符串的第一个字符（即去掉前面的#）

memmove(line, line + 1, strlen(line));

}

// 将剩余的字符串按 '#' 分割成数组 a

char\* a[MAX\_LEN];

int aCount = 0;

char\* token = strtok(line, "#");

while (token != NULL) {

a[aCount++] = token;

token = strtok(NULL, "#");

}

// 初始化结果字符串 res

char res[MAX\_LEN] = "";

// 遍历分割后的数组 a，分别根据当前模式处理各个部分

for (int i = 0; i < aCount; i++) {

// 如果当前部分为空字符串，则切换模式

if (strlen(a[i]) == 0) {

mod ^= 1;

continue;

}

// 非第一部分且当前部分不为空时，切换模式

if (i > 0 && strlen(a[i]) != 0) {

mod ^= 1;

}

if (mod == 0) {

// 如果当前是数字模式

char buf[MAX\_LEN] = "";

num\_mod(a[i], buf);

strcat(res, buf);

} else {

// 如果当前是字母模式

char buf[MAX\_LEN] = "";

eng\_mod(a[i], buf);

strcat(res, buf);

}

}

// 输出处理后的结果

printf("%s\n", res);

return 0;

}



const readline = require("readline");

// 创建接口以读取标准输入

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout,

});

// 定义手机键盘按键对应的字符表，mp[i] 代表按键 i 对应的字符

const mp = [

" ",

",.",

"abc",

"def",

"ghi",

"jkl",

"mno",

"pqrs",

"tuv",

"wxyz",

];

// 定义处理数字模式的函数 num\_mod，将字符串中不是 '/' 的字符拼接起来

function num\_mod(s) {

let res = ""; // 存放结果

for (let c of s) {

if (c === "/") {

// 忽略 '/' 字符

continue;

}

res += c; // 直接将数字加入结果

}

return res; // 返回拼接后的结果

}

// 定义处理字母模式的函数 eng\_mod，将按键的数字组合转换为对应字母

function eng\_mod(s) {

let res = ""; // 存放最终结果

let t = "";

t += s[0]; // 初始化t为字符串的第一个字符

// 遍历字符串的其余字符

for (let i = 1; i < s.length; i++) {

// 如果当前字符与前一个字符不同，则在中间插入 '/'

if (s[i] !== s[i - 1]) {

t += "/";

}

t += s[i]; // 将当前字符加入 t 中

}

// 按 '/' 分割 t，得到各个按键分组

const tmp = t.split("/");

// 遍历每个按键分组，转换成对应字母

tmp.forEach((str) => {

if (str.length === 0) {

// 忽略空字符串

return;

}

// 获取对应按键的字母集合

const x = mp[Number(str[0])];

// 根据按键的按次数计算字符索引

let idx = str.length % x.length;

idx = (idx - 1 + x.length) % x.length; // 调整索引以防负数

res += x[idx]; // 添加对应字母到结果字符串

});

return res; // 返回字母转换后的结果

}

// 读取输入

let input = "";

rl.on("line", (line) => {

input = line;

// 初始化mod变量为0，用于模式切换

let mod = 0;

// 循环处理输入字符串中以 '#' 开头的部分

while (input.length > 0 && input[0] === "#") {

// mod异或1，用于切换模式

mod ^= 1;

// 删除字符串的第一个字符（即去掉前面的#）

input = input.slice(1);

}

// 将剩余的字符串按 '#' 分割成数组 a

const a = input.split("#");

// 初始化结果字符串 res

let res = "";

// 遍历分割后的数组 a，分别根据当前模式处理各个部分

for (let i = 0; i < a.length; i++) {

// 如果当前部分为空字符串，则切换模式

if (a[i].length === 0) {

mod ^= 1;

continue;

}

// 非第一部分且当前部分不为空时，切换模式

if (i > 0 && a[i].length !== 0) {

mod ^= 1;

}

if (mod === 0) {

// 如果当前是数字模式

res += num\_mod(a[i]);

} else {

// 如果当前是字母模式

res += eng\_mod(a[i]);

}

}

// 输出处理后的结果

console.log(res);

// 关闭接口

rl.close();

});



package main

import (

"bufio" // 引入bufio包，用于缓冲读写

"fmt" // 引入fmt包，用于格式化输入输出

"os" // 引入os包，用于操作系统功能接口

"strings" // 引入strings包，用于字符串操作

)

// 定义手机键盘按键对应的字符表，mp[i] 代表按键 i 对应的字符

var mp = []string{" ", ",.", "abc", "def", "ghi", "jkl", "mno", "pqrs", "tuv", "wxyz"}

// 定义处理数字模式的函数 num\_mod，将字符串中不是 '/' 的字符拼接起来

func num\_mod(s string) string {

res := "" // 存放结果的字符串

for \_, c := range s { // 遍历字符串的每个字符

if c == '/' { // 忽略 '/' 字符

continue

}

res += string(c) // 将数字字符添加到结果中

}

return res // 返回拼接后的结果

}

// 定义处理字母模式的函数 eng\_mod，将按键的数字组合转换为对应字母

func eng\_mod(s string) string {

res := "" // 存放最终结果的字符串

t := string(s[0]) // 初始化 t 为字符串的第一个字符

// 遍历字符串的其余字符

for i := 1; i < len(s); i++ {

// 如果当前字符与前一个字符不同，则在中间插入 '/'

if s[i] != s[i-1] {

t += "/"

}

t += string(s[i]) // 将当前字符加入 t 中

}

// 按 '/' 分割 t，得到各个按键分组

tmp := strings.Split(t, "/")

// 遍历每个按键分组，转换成对应字母

for \_, str := range tmp {

if len(str) == 0 { // 忽略空字符串

continue

}

// 获取对应按键的字母集合

x := mp[str[0]-'0']

// 根据按键的按次数计算字符索引

idx := len(str) % len(x)

idx = (idx - 1 + len(x)) % len(x) // 调整索引以防负数

res += string(x[idx]) // 添加对应字母到结果字符串

}

return res // 返回字母转换后的结果

}

func main() {

scanner := bufio.NewScanner(os.Stdin) // 创建新的扫描器从标准输入读取数据

if scanner.Scan() { // 扫描输入

line := scanner.Text() // 读取输入的一整行

// 初始化 mod 变量为0，用于模式切换

mod := 0

// 循环处理输入字符串中以 '#' 开头的部分

for len(line) > 0 && line[0] == '#' {

mod ^= 1 // mod 异或 1，用于切换模式

line = line[1:] // 删除字符串的第一个字符（即去掉前面的#）

}

a := strings.Split(line, "#") // 将剩余的字符串按 '#' 分割成数组 a

res := "" // 初始化结果字符串

// 遍历分割后的数组 a，分别根据当前模式处理各个部分

for i := 0; i < len(a); i++ {

// 如果当前部分为空字符串，则切换模式

if len(a[i]) == 0 {

mod ^= 1 // 切换模式

continue

}

// 非第一部分且当前部分不为空时，切换模式

if i > 0 && len(a[i]) != 0 {

mod ^= 1

}

if mod == 0 {

res += num\_mod(a[i]) // 数字模式

} else {

res += eng\_mod(a[i]) // 字母模式

}

}

fmt.Println(res) // 输出处理后的结果

}

}