【华为OD机考 统一考试机试C卷】拼接URL（C++ Java JavaScript Python C语言）

华为OD机考:统一考试 C卷 + D卷 + B卷 +A卷

目前在考C卷，经过两个月的收集整理，C卷真题已基本整理完毕
抽到原题的概率为2/3到3/3，也就是最少抽到两道原题。请注意：大家刷完C卷真题，最好要把B卷的真题刷一下，因为C卷的部分真题来自B卷。
另外订阅专栏还可以联系笔者开通在线 OJ 进行刷题，提高刷题效率。
真题目录：华为OD机考机试 真题目录（C卷 + D卷 + B卷 + A卷） + 考点说明
专栏：2023华为OD机试(B卷+C卷+D卷)（C++JavaJSPy）
华为OD面试真题精选：华为OD面试真题精选
在线OJ：点击立即刷题，模拟真实机考环境

题目描述

给定一个url前缀和url后缀,通过,分割 需要将其连接为一个完整的url

- 如果前缀结尾和后缀开头都没有/，需要自动补上/连接符
- 如果前缀结尾和后缀开头都为/，需要自动去重

约束：不用考虑前后缀URL不合法情况

输入描述

url前缀(一个长度小于100的字符串) url后缀(一个长度小于100的字符串)

输出描述

拼接后的url

用例

输入	/abc/,/bcd
输出	/abc/bcd
说明	无

C++

```cpp
#include <iostream>
#include <string>
#include <sstream>
using namespace std;
int main() {
  string line;
  getline(cin, line);
  istringstream iss(line);
  string prefix, suffix;
  getline(iss, prefix, ',');
  getline(iss, suffix, ',');

  if (prefix.empty() || suffix.empty()) {
    cout << "/" << endl;
    return 0;
  }

  bool prefixHasSlash = (prefix.back() == '/');
  bool suffixHasSlash = (suffix.front() == '/');

  string url = prefix;
  if (!prefixHasSlash && !suffixHasSlash) {
    url += "/";
  }
  url += suffix;

  size_t pos = url.find("//");
  while (pos != string::npos) {
    url.replace(pos, 2, "/");
    pos = url.find("//", pos + 1);
  }

```

```
  cout << url << endl;

  return 0;
}
```

java

```java
import java.util.Scanner;

public class Main {
  public static void main(String[] args) {
    try (Scanner scanner = new Scanner(System.in)) {
      // 读取输入的url前缀和url后缀
      String line = scanner.nextLine();
      String[] split = line.split(",");

      // 如果没有输入前缀和后缀，则输出"/"
      if (split.length == 0) {
        System.out.println("/");
        return;
      }

      // 获取前缀和后缀
      String prefix = split[0];
      String suffix = split[1];

      // 检查前缀结尾和后缀开头是否有"/"
      boolean prefixHasSlash = prefix.endsWith("/");
      boolean suffixHasSlash = suffix.startsWith("/");

      // 拼接url
      StringBuilder urlBuilder = new StringBuilder();
      urlBuilder.append(prefix);

      // 如果前缀结尾和后缀开头都没有"/"，则补上"/"
      if (!prefixHasSlash && !suffixHasSlash) {
        urlBuilder.append("/");
      }

      urlBuilder.append(suffix);
```

```
        // 去重"/"
        String url = urlBuilder.toString().replaceAll("/+", "/");
        System.out.println(url);
    }
  }
}
```

javascript

```javascript
const readline = require('readline');

const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout
});

rl.on('line', (line) => {
  const split = line.split(",");

  if (split.length === 0) {
    console.log("/");
    rl.close();
    return;
  }

  const prefix = split[0];
  const suffix = split[1];

  const prefixHasSlash = prefix.endsWith("/");
  const suffixHasSlash = suffix.startsWith("/");

  const urlBuilder = [];
  urlBuilder.push(prefix);

  if (!prefixHasSlash && !suffixHasSlash) {
    urlBuilder.push("/");
  }

  urlBuilder.push(suffix);
```

```
31
32
33    const url = urlBuilder.join("").replace(/\/+/g, "/");
34    console.log(url);
35
36    rl.close();
});
```

python

```python
import re

# 读取输入的url前缀和url后缀
line = input()
split = line.split(",")

# 如果没有输入前缀和后缀，则输出"/"
if len(split) == 0:
    print("/")
    exit()

# 获取前缀和后缀
prefix = split[0]
suffix = split[1]

# 检查前缀结尾和后缀开头是否有"/"
prefixHasSlash = prefix.endswith("/")
suffixHasSlash = suffix.startswith("/")

# 拼接url
urlBuilder = []
urlBuilder.append(prefix)

# 如果前缀结尾和后缀开头都没有"/"，则补上"/"
if not prefixHasSlash and not suffixHasSlash:
    urlBuilder.append("/")

urlBuilder.append(suffix)

# 去重"/"

```

```
url = re.sub("/{2,}", "/", "".join(urlBuilder))
print(url)
```

C语言

```c
#include <stdio.h>
#include <string.h>
#include <stdbool.h>

#define MAX_URL_LEN 100

int main() {
    char prefix[MAX_URL_LEN], suffix[MAX_URL_LEN];
    scanf("%99[^,],%99s", prefix, suffix); // 读取两个字符串，以逗号分隔

    // 检查字符串是否为空
    if (strlen(prefix) == 0 || strlen(suffix) == 0) {
        printf("/\n");
        return 0;
    }

    bool prefixHasSlash = prefix[strlen(prefix) - 1] == '/';
    bool suffixHasSlash = suffix[0] == '/';

    // 拼接URL
    char url[2 * MAX_URL_LEN];
    strcpy(url, prefix);
    if (!prefixHasSlash && !suffixHasSlash) {
        strcat(url, "/"); // 如果两者都没有斜杠，则添加斜杠
    }
    if (prefixHasSlash && suffixHasSlash) {
        strcat(url, suffix + 1); // 如果两者都有斜杠，则跳过后缀的第一个斜杠
    } else {
        strcat(url, suffix);
    }

    // 处理连续的斜杠
    for (int i = 0; url[i] != '\0'; i++) {
        if (url[i] == '/' && url[i + 1] == '/') {
            memmove(&url[i], &url[i + 1], strlen(&url[i + 1]) + 1);
```

```
            i--; // 回退一步以处理新的当前位置
        }
    }

    printf("%s\n", url); // 输出最终的URL

    return 0;
}
```

完整用例

用例1

```
/acm,/bb
```

用例2

```
/abc/,/bcd
```

用例3

```
/acd,bef
```

用例4

```
/abc/def,/ghi/jkl
```

用例5

```
/abc,/def
```

用例6

```
,
```

用例7

```
/abc/,def
```

用例8

```
abc,def/
```

用例9

```
/abc,/
```

用例10

```
/abc/,def/ghi
```

文章目录

用例9
用例10