# Experiment10-董皓彧

环境：

```
gcc.exe (x86_64-win32-seh-rev0, Built by MinGW-W64 project) 8.1.0
Visual Stdio Code 1.84.2
```

作业仓库地址：
https://github.com/FHYQ-Dong/Tsinghua-Program-Design-Assignments/tree/main/Experiment10

## 必做题

### Experiment10-1

题目：

```
给定两个字符串 sA 和 sB：长字符串 sA 长长度不超过 30，可能由字母/数字/符号等任意字符构成，但不含空格、换行符；模式字符串 sB 长度不超过 sA 的长度，可能包含除空格和换行符外的任意字符。现要求在长字符串 sA 中查找匹配模式字符串 sB，请你找到 sB 在 sA 中出现的所有位置。注意：模式字符串 sB 中含有一个或若干个特殊字符 '?'，在匹配过程中，每个 '?' 可以匹配 sA 中的任意一个字符，而 sB 中的其他非 '?' 字符必须与 sA 中匹配的子串完全相同。要求：输入两行，第一行为长字符串 sA，第二行为模式字符串 sB；输出：sB 在 sA 中出现的所有位置（用若干非负整数表示），sA 的起始位置从 0 开始计算；如果没有找到任何匹配，输出 No match found。
```

输入格式：

```
共 2 行，分别为 sA 与 sB
```

输出格式：

```
共 1 行，No match found 或若干非负整数，表示 sB 在 sA 中出现的所有位置
```

代码：

```
#include <stdio.h>
#include <string.h>
typedef char bool;
#define true 1
#define false 0

int main() {
    char sA[50], sB[50];
    bool found = false;
    scanf("%s", sA); scanf("%s", sB);
    int lengthA = strlen(sA), lengthB = strlen(sB);
    for (int i=0; i<lengthA; ++i) {
        bool flag = true;
        for (int j=0; j<lengthB; ++j) {
            if (i+j >= lengthA) {
```

```
                flag = false;
                break;
            }
            if (sB[j] == '?') continue;
            else {
                if (sA[i+j] != sB[j]) {
                    flag = false;
                    break;
                }
            }
        }
        if (flag) {
            printf("%d ", i);
            found = true;
        }
    }
    if (!found) printf("No match found");
    return 0;
}
```

输入1：

```
abcdefghc*exyzcferpk
c?e
```

输出1：

```
2 8 14
```

输入2：

```
wioeu4r2i;4ehn;hqi342ih
?i
```

输出2：

```
0 7 16 20
```

输入3：

```
OEUYUR#H*(Wrwdjksv
#??*
```

输出3：

```
No match found
```

输入4：

```
111111111111111
11?1?1
```

输出4：

```
0 1 2 3 4 5 6 7 8 9
```

输入5：

```
3quroi3
3?
```

输出5：

```
0
```

## Experiment10-2

题目：

```
对于一个由 a-z（小写）组成的字符串，将其中的元音反转，而辅音不反转。如对于字符串 "hello" ，替换后的字符串为 "holle"。（注：元音为 a，e，i，o，u）
```

输入格式：

```
共 1 行，一个字符串
```

输出格式：

```
共 1 行，一个翻转元音后的字符串
```

代码：

```
#include <stdio.h>
#include <string.h>
#define MAXN 100001
typedef char bool;
#define true 1
#define false 0

bool is_vowel(char c) { return c=='a' || c=='e' || c=='i' || c=='o' || c=='u'; }

void swap(char *a, char *b) { char t = *a; *a = *b; *b = t; }

int main() {
    char s[MAXN];
    scanf("%s", s);
    int length = strlen(s);
    int l=0, r=length-1;
    while (l<r) {
        while (l<r && !is_vowel(s[l])) ++l;
        while (l<r && !is_vowel(s[r])) --r;
        if (l<r) swap(&s[l], &s[r]);
        ++l; --r;
```

```
    }
    printf("%s", s);
    return 0;
}
```

输入1：

```
aeiou
```

输出1：

```
uoiea
```

输入2：

```
ewoyurekadhkwrjscvwew
```

输出2：

```
ewayerukodhkwrjscvwew
```

输入3：

```
oqekfhvjfsvnsa
```

输出3：

```
aqekfhvjfsvnso
```

输入4：

```
aaaaaaaaaaaaaaa
```

输出4：

```
aaaaaaaaaaaaaaa
```

输入5：

```
andsjklvhseruiohwfvnbjf
```

输出5：

```
ondsjklvhsirueahwfvnbjf
```

# Experiment10-3

题目：

> 输入一段由英文字母（区分大小写）组成的字符串，将其按 ASCII 码从大到小的顺序输出。

输入格式：

> 共 1 行：一个字符串，区分大小写

输出格式：

> 共 1 行，一个排序后的字符串

代码：

```
#include <stdio.h>
#include <string.h>
#define MAXN 100001
typedef char bool;
#define true 1
#define false 0

void qsort(char *begin, char *end, bool (*cmp)(char, char)) {
    if (begin >= end) return;
    char *pivot = begin;
    char *left = begin, *right = end;
    while (left < right) {
        while (left < right && !cmp(*right, *pivot)) --right;
        while (left < right && cmp(*left, *pivot)) ++left;
        if (left < right) {
            char temp = *left;
            *left = *right;
            *right = temp;
        }
    }
    char temp = *left;
    *left = *pivot;
    *pivot = temp;
    qsort(begin, left-1, cmp);
    qsort(left+1, end, cmp);
    return;
}

bool cmp(char a, char b) { return a > b; }

int main() {
    char s[MAXN];
    scanf("%s", s);
    int length = strlen(s);
    qsort(s, s+length-1, cmp);
    printf("%s", s);
    return 0;
```

```
}
```

输入1：

```
ljkAdsdlkvjASABVJK
```

输出1：

```
vsllkkjjddVSKJBAAA
```

输入2：

```
AAAAAAAAAaaaaaaaaa
```

输出2：

```
aaaaaaaaaAAAAAAAAA
```

输入3：

```
AKJDncDKsdkjvHfADgCKDJSA
```

输出3：

```
vsnkjgfdcSKKKJJHDDDDCAAA
```

输入4：

```
ewidsuvkjnraiu
```

输出4：

```
wvuusrnkjiieda
```

输入5：

```
KADCHBNJDSVKJDSV
```

输出5：

```
VVSSNKKJJHDDDCBA
```

# 选做题

# Optional-Experiment10-1

题目：

> 从键盘输入两个字符串，判断它们是否属于异位字符串。所谓异位字符串，就是把一个字符串中字母的顺序改变，得到的字符串。为了简化，假设字符串只包含小写英文字母。

输入格式：

> 共 2 行，两个字符串

输出格式：

> 1 或 0，表示是否是异位字符串

代码：

```
#include <stdio.h>
#include <string.h>
#define MAXN 100001
typedef char bool;
#define true 1
#define false 0

void qsort(char *begin, char *end, bool (*cmp)(char, char)) {
    if (begin >= end) return;
    char *pivot = begin;
    char *left = begin, *right = end;
    while (left < right) {
        while (left < right && !cmp(*right, *pivot)) --right;
        while (left < right && cmp(*left, *pivot)) ++left;
        if (left < right) {
            char temp = *left;
            *left = *right;
            *right = temp;
        }
    }
    char temp = *left;
    *left = *pivot;
    *pivot = temp;
    qsort(begin, left-1, cmp);
    qsort(left+1, end, cmp);
    return;
}

bool cmp(char a, char b) { return a > b; }

int main() {
    char s[MAXN], t[MAXN];
    scanf("%s", s); scanf("%s", t);
    int lengthS = strlen(s), lengthT = strlen(t);
    if (lengthS != lengthT) {
        printf("0");
```

```
        return 0;
    }
    else {
        qsort(s, s+lengthS-1, cmp);
        qsort(t, t+lengthT-1, cmp);
        for (int i = 0; i < lengthS; ++i) {
            if (s[i] != t[i]) {
                printf("0");
                return 0;
            }
        }
        printf("1");
    }
    return 0;
}
```

输入1：

```
hello
olleh
```

输出1：

```
1
```

输入2：

```
wo3efohwe
wefo3hwoe
```

输出2：

```
1
```

输入3：

```
111111111111
11111112
```

输出3：

```
0
```

输入4：

```
wkeiljflwi4fj
wkeiljflwi4fj
```

输出4：

```
1
```

输入5：

```
1
0
```

输出5：

```
0
```

## Optional-Experiment10-2

题目：

在网络编程时，经常需要把 IP 地址转换为计算机内部的整型数来处理。C++系统提供的 atoi() 就是实现该功能。参考该函数，编写另一个函数（如aton() ），其功能是将输入入的 IPv4 地址（字符串，例如 166.111.64.89 ）字符串转换为 32 位整数输出。

输入格式：

```
共 1 行，一个 IPv4 地址
```

输出格式：

```
共 1 行，一个 32 位整数
```

代码：

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

int aton(const char s[]) {
    int res = 0, tmp = 0;
    int p = 0;
    while(s[p]) {
        if (s[p] == '.') {
            res = (res << 8) + tmp;
            tmp = 0;
        }
        else tmp = tmp * 10 + (s[p] - '0');
        ++p;
    }
    return (res << 8) + tmp;
}

int main() {
    char s[100];
    scanf("%s", s);
    printf("%d", aton(s));
    return 0;
}
```

输入1：

```
192.168.1.1
```

输出1：

```
-1062731519
```

输入2：

```
1.1.1.1
```

输出2：

```
16843009
```

输入3：

```
127.0.0.1
```

输出3：

```
2130706433
```

输入4：

```
10.0.0.1
```

输出4：

```
167772161
```

输入5：

```
172.18.0.1
```

输出5：

```
-1408106495
```