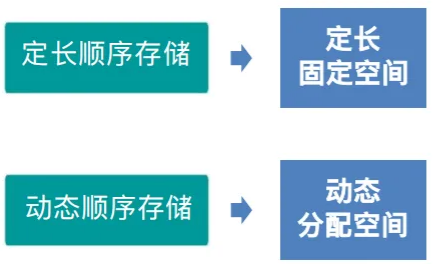
C4-字符串（🌟🌟）

1.空串VS空格串VS空白串

空白串：含任意空白字符

2.存储实现：顺序存储实现与链式存储实现




```
#define MaxLen 255        // 1. 定义字符串的最大长度
typedef struct SString {  // 2. 定义顺序串
    char ch[MaxLen];      // 2.1 定义字符串序列
    int length;           // 2.2 定义串内字符长度变量
}SString;
```

```
#typedef struct HString {   // 1. 定义动态存储的堆字符串
    char *ch;               // 2. 按串长分配存储区
    int length;             // 3. 定义串内字符长度变量
}HString;
```

存储方式	空间分配时机	位置	空间灵活性	内存管理	适用场景
定长顺序存储	编译时固定分配	栈	大小固定（由数组定义决定），超出则栈溢出	函数 / 作用域结束时自动回收	长度固定且较小的字符串
动态顺序存储	运行时动态分配	堆	可按需调整大小（通过 realloc 等）	需手动分配（malloc/new）和释放（free/delete）	长度不确定或较大的字符串

插入（元素后移实现），删除 （前移），

截取（从pos开始复制到新串），连接，比较，五种操作在顺序与链式中的实现

3.字符串的模式匹配

（1）BF算法（暴力）O（nm）

BF

```c
Position PatternMatchBF(String s, String t)
{
int n, m, i;
Position p;
n = s->length;
m = t->length;
for (p=0; p<=(n-m); p++) {
    for (i=0; i<m; i++) {
        if (s->data[p+i] != t->data[i]) {
            break; /* 不匹配则从下一个位置p开始 /}
}
    if (i==m) { / 匹配成功 /
        break; }
}
    if (p>(n-m)) { / 匹配不成功 */
        p = NIL; }
return p;
}
```

(2) KMP算法　最坏O（m+n）其中求next数组花O（m）

核心：主串指针不回溯

步骤：先求next数组，再开始匹配。

1. **next数组next[j]实质是存储当模式串第j个位置匹配不上时，应该把j调整到多少，　（注意特例当第一个元素就匹配不上时，将j=0,再将i与j统一都加一）**

```c
if(s[i]!=t[j])
    j=next[j];
if(j==0 | s[i]==t[j]){
    i++;
    j++;
}
```

朴素模式匹配 v.s. KMP算法

```
int Index(SString S,SString T){
    int i=1,j=1;
    while(i<=S.length && j<=T.length){
        if(S.ch[i]==T.ch[j]){
            ++i; ++j;    //继续比较后继字符
        }
        else{
            i=i-j+2;
            j=1;         //指针后退重新开始匹配
        }
    }
    if(j>T.length)
        return i-T.length;
    else
        return 0;
}
```

匹配失败时，主串指针 i 疯狂回溯

```
int Index_KMP(SString S,SString T,int next[]){
    int i=1, j=1;
    while(i<=S.length&&j<=T.length){
        if(j==0||S.ch[i]==T.ch[j]){
            ++i;
            ++j;                 //继续比较后继字符
        }
        else
            j=next[j];           //模式串向右移动
    }
    if(j>T.length)
        return i-T.length;       //匹配成功
    else
        return 0;
}
```

匹配失败时，主串指针 i 不回溯

注：只改动两处，添加next数组

2. 求next数组

模式串 T = ababaa

序号j	1	2	3	4	5	6
模式串	a	b	a	b	a	a
next[j]	0	1				

next[1]都无脑写 0
next[2]都无脑写 1

其他 next：在不匹配的位置前，划一根美丽的分界线
模式串一步一步往后退，直到分界线之前“能对上”，或模式串完全跨过分界线为止。此时 j 指向哪儿，next数组值就是多少

或是直接等于在去掉不匹配位置后的子串最大公共前后缀的长度加一

（3）KMP算法优化

思想：基础运算匹配代码不变，只改变优化next数组（当j跳转的位置与原字母相同时再次转换为next[j]）

```c
nextval[1]=0;
for(int i=2;i<T.length;j++){
  if(T.ch(next[j]==T.ch[j]))
      nextval[j]=nextval[next[j]];
    else
        nextval[j]=next[j]
}
```

错题：

4. 首次提交时间: 2025-10-16 15:57:55 最后一次提交时间: 2025-10-16 16:11:13 得分：0.00

若串 S1='ABCDEFG', S2='9898' ,S3='###',S4='012345',执行
concat(replace(S1,substr(S1,length(S2),length(S3)),S3),substr(S4,index(S2,'8'),length(S2)))
index(S2,'8') 含义为查找 '8' 在 S2 中第一次出现的位置。其结果为（ ）【f _____ 【正确答案: E】

A ABC###G0123
B ABCD###2345
C ABC###G2345
D ABC###2345
E ABC###G1234
F ABCD###1234
G ABC###01234

解析

a.计算机中默认串的位置从1开始

b.substr(S4,2,4)表示从位置2开始截取4个字符，而非从2截到4！！！

c.length不包含空字符

7. 首次提交时间: 2025-10-16 16:16:44 最后一次提交时间: 2025-10-16 16:16:44

以下给字符数组str定义和赋值正确的是B _____ 【正确答案: B】

(A)char str[10]; str = "China";

(B)char str[] = "China";

(C)char str[10]; strcpy (str, "abcdefghijklmn");

(D)char str[10] = "abcdefghigklmn";

str不能直接赋值！

代码题，子串匹配

BF C

```c
int main()
{
    char p1[1000]={'\0'},p2[1000]={'\0'};
    int i=0,j=0,len1=0,len2=0;
    fgets(p1,1000,stdin);
    fgets(p2,1000,stdin);
    len1=strlen(p1)-1;
    len2=strlen(p2)-1;
    for (i=0;i<len1;i++)
    {
        for (j=0;j<len2;j++)
        {
            if ((i+j)>len1)
            {
                break;
            }
            if (p2[j]==p1[i+j])
            {
                if (j==(len2-1))
                {
                    printf("%d",i);
                    return 0;
                }
            }
            else
                break;
        }
    }
    printf("-1");
}
//可以写出，但缺少 if ((i+j)>len1)检查长度是否足够步骤，浪费时间!
```