KMP

- KMP定义：在一个长字符串中匹配一个短子串的无回溯算法。
- 在Brute-Force字符串匹配中，在主串中查找子串是否存在的时候，主串难免要回溯。时间复杂度O(n*m)。

- 讨论一般情况：设 s="$s_0 s_1 \ldots s_{n-1}$" ,t="$t_0 t_1 \ldots t_{m-1}$",当，模式匹配到 $s_i \neq t_j$ (0<=i<n,0<=j<m)时，必存在："$s_{i-j} s_{i-j+1} \ldots s_{i-1}$"="$t_0 t_1 \ldots t_{j-1}$"此时，若模式串中存在可以相互重叠的真子串 ，

满足 "$t_0 t_1 \ldots t_{k-1}$"="$t_{j-k} t_{j-k+1} \ldots t_{j-1}$"$(0 < k < j)$ 则说明模式串中的子串 "$t_0 t_1 \ldots t_{k-1}$" 已和主串"$s_{i-k} s_{i-k+1} \ldots s_{i-1}$"匹配，下一次可以直接比较 $s_i t_k$

在模式串t中，所有真子串的寻找问题，就成了求next函数的问题

- 首先定义next[j]函数：

$$\text{next}[j]=\begin{cases} \text{Max}\{k \mid 0<k<j \text{ 且 } \text{“}t_0t_1\ldots t_{k-1}\text{”}=\text{“}t_{j-k}t_{j-k+1}\ldots t_{j-1}\text{”}\} & \text{此集合非空时} \\ 0 & \text{其他情况} \\ -1 & j=0 \end{cases}$$

- **模式串的真子串**：在模式串”$t_0 t_1 \ldots t_{j-1}$”中存在这样两个长度均小于j的字符串，其中一个以t_0为首字符，另一个字符串以t_{j-1}为末字符，满足”$t_0 t_1 \ldots t_{k-1}$”=“$t_{j-k} t_{j-k+1} \ldots t_{j-1}$”条件，且这样的相等子串是所有这样相等子串中长度最长的，如果规定这样的相等子串的长度可以等于子串“$t_0 t_1 \ldots t_{j-1}$”本身，即规定0<k<=j,则这样的相等子串为模式串t的非真子串；如果这样的相等子串的长度必须小于模式串“$t_0 t_1 \ldots t_{j-1}$”本身，及规定0<k<j,这样的相等子串为模式串t的真子串。

```
void GetNext(){
        int j=1,k=0;
        next[0]=-1, next[1]=0;
        while(j<len-1) {
            if(str[j]==str[k]){
              next[++j]=++k;
             } else if(k==0) {
                    next[++j]=0;
             } else{
                   k=next[k];
             }
        }
}
```