HW4 林宸昊 PB20000034

4.7

s	a	a	a	b
next	-1	0	1	2
nextval	-1	-1	-1	2

t	a	b	c	a	b	a	a
next	-1	0	0	0	1	2	1
nextval	-1	0	0	-1	0	2	1

u	a	b	c	a	a	b	b	a	b	c	a	b	a	a	c	b	a	c	b	a
next	-1	0	0	0	1	1	2	0	1	2	3	4	2	1	1	0	0	1	0	0
nextval	-1	0	0	-1	1	0	2	-1	0	0	-1	4	2	1	1	0	-1	1	0	-1

4.10

```
//递推公式为：此次操作得到的串为上一个已逆置首尾的串再次置换尚未变动的首尾
void reverse(String &s){
    char temp;
    for(int i = 0;i < s.length / 2;i++){
        temp = s.elem[i];
        s.elem[i] = s.elem[s.length - i - 1];
        s.elem[s.length - 1 - i] = temp;
    }
}
```

4.16

```
//此处认为参与比较的字符串均以进行初始化
int compare(String s, String t){
    if(s.length && t.length){
        int i;
        for(i = 0;(s.elem[i] != t.elem[i]) && (i < s.length) && (i <
t.length);i++);
        return s.elem[i] - t.elem[i];
    }
    if(!s.length && t.length){
        return t.elem[0];
    }
    if(!t.length && s.length){
        return s.elem[0];
    }
```

```
    return 0;
}
```

4.17

```
//先找出所有目标子串的起始位置，然后在判断是否会溢出，最后进行插入
int Replace(String &s, String t, String v){
    int *pos;
    pos = (int *)malloc(s.length * sizeof(int));
    memset(pos, 0, s.length);
    int j = 0;
    for(int i = 0;i <= s.length - t.length;i++){
        int temp = IndexString(s, t);
        if(temp > 0) pos[j++] = temp;
        else break;
        i = pos[j] + t.length;
    }
    //如果无法找到目标子串
    if(!j) return -1;
    //如果插入后会造成溢出
    int cur_size = (j + 1) * (v.length - t.length) + s.length;
    if(cur_size > s.max_size) return -1;

    init_string(ans, s.max_size);//初始化
    //开始替换
    for(int i = 0;i <= j;i++){
        StrInsert(s, v, pos[i]);
    }
    return 1;
}
```