数据结构作业(第三次)

PB20111686 黄瑞轩

3.15

```
typedef struct {
    int* elem;//基地址
    int top_1;
    int top_2;
}tws;

bool inistack(tws* t) {
    t->elem = new int[LIST_INIT_SIZE];
    if (t->elem) {
        t->top_1 = 0;
        t->top_2 = LIST_INIT_SIZE - 1;
        return true;
    }
    return false;
}

//设i=0表示[0]为栈底，i=1表示[length-1]为栈底
bool push(tws* t, int i, int x) {
    if (t->top_2 - t->top_1 <= 1)return false;
    if (i) {
        t->top_2--;
        t->elem[top_2] = x;
    }
    else {
        t->top_1++;
        t->elem[top_1] = x;
    }
    return true;
}

bool pop(tws* t, int i) {
    if (i) {
        if (top_2 == LIST_INIT_SIZE - 1) {
            return false;
        }
        else {
            top_2++;
            return true;
        }
    }
    else {
        if (top_1 == 0) {
            return false;
        }
        else {
            top_1--;
            return true;
        }
    }
}
```

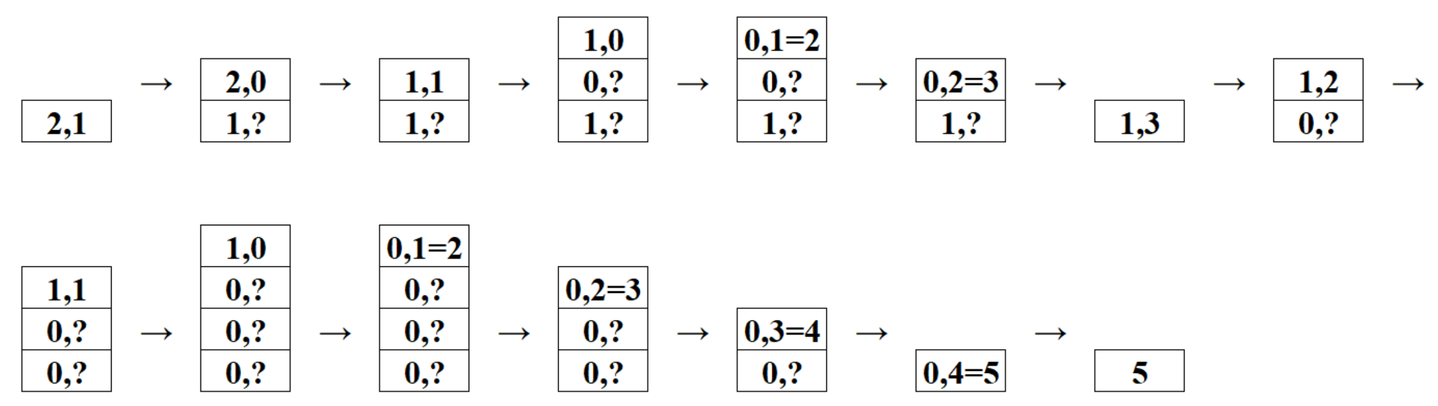
3.19

```
bool isValid(SqList* str) {
    int length = str->length;
    char* s = str->elem;
    stack<char> use;
    for (int i = 0; i < length; i++) {
        if (s[i] == '(' || s[i] == '[' || s[i] == '{') {
            use.push(s[i]);
        }
        else {
            if (use.empty() == true) {
                return false;
            }
            if (s[i] - use.top() <= 2) {
                use.pop();
            }
            else {
                return false;
            }
        }
    }
    if (use.empty() == true) {
        return true;
    }
    else {
        return false;
    }
}
```

3.27 (1)

```
int akm(int m, int n) {
    if (m == 0)return n + 1;
    if (n == 0)return akm(m - 1, 1);
    return akm(m - 1, akm(m, n - 1));
}
```

3.27 (3)



```
typedef{
    bool valValid = false;//表示当前的value是否是非缺省值，false表示不是
    int first = -1;
    int second = -1;
    int value = -1;
}akmPackage;

int akm(int m, int n) {
    stack<akmPackage> stk;
    akmPackage root = { 0,0,m,n,-1 };
    std.push(root);//先将目标压入栈
    do {
        akmPackage top = stk.top();
        if (top.valValid) {
            stk.pop();
            akmPackage subTop = stk.top();
            if (subTop.second != 0) {//m!=0,n!=0的情形
                subTop.second = top.value;
                stk.pop();
                stk.push(subTop);
            }
            else if (subTop.first != 0) {//m!=0,n==0的情形
                subTop.first = top.first - 1;
                subTop.second = 1;
                stk.pop();
                stk.push(subTop);
            }
        }
        else {
            if (top.second != 0) {//m!=0,n!=0的情形
                akmPackage newPackage = { 0,top.first,top.second - 1,-1 };
                stk.push(newPackage);
            }
            else if (top.first != 0) {//m!=0,n==0的情形
                akmPackage newPackage = { 0,top.first - 1,1,-1 };
                stk.push(newPackage);
            }
            else {//m==0,n==0的情形
                top.value = top.second + 1;
                top.valValid = true;
                stk.pop();
                stk.push(top);
            }
        }
    } while (stk.size() <= 1);
    return stk.top().value;
}
```

3.29

假定`ElemType`是`int`。

```
#define MAXQSIZE    100             /* 最大队列长度 */
typedef struct {
    pair<int, bool>* base;          /* 存储空间，使用std::pair */
    int front;                      /* 头指针，指向队列的头元素 */
    int rear;                       /* 尾指针，指向队尾元素的下一个位置 */
}SqQueue;

bool push(SqQueue* q, int x) {
    if (q->base[q->rear]->second) {
        return false;
    }
    else {
        q->base[q->rear]->first = x;
        q->base[q->rear]->second = true;
        q->rear = (q->rear + 1) % MAXQSIZE;
    }
    return true;
}

bool pop(SqQueue* q) {
    if (q->base[q->front]->second) {
        q->base[q->front]->second = false;
        q->front = (q->front + 1) % MAXQSIZE;
        return true;
    }
    return false;
}
```

设标志使用范围：

- 队列中每个元素占空间较多
- 循环队列容量较小

不设标志使用范围：

- 队列中每个元素占空间较小
- 循环队列容量较大