

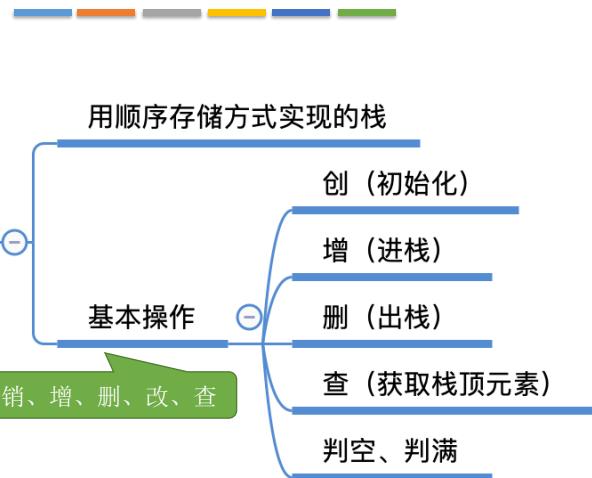
本节内容

顺序栈 的实现

王道考研/CSKAOYAN.COM

1

知识总览



王道考研/CSKAOYAN.COM

2

顺序栈的定义

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 静态数组存放栈中元素
    ElemenType data[MaxSize]; // 栈顶指针
    int top;
} SqStack;

```

Sq: sequence —— 顺序

```

void testStack() {
    SqStack S; // 声明一个顺序栈(分配空间)
    // ...后续操作...
}

```

内存

顺序存储: 给各个数据元素分配连续的存储空间, 大小为 $\text{MaxSize} * \text{sizeof}(\text{ElemenType})$

王道考研/CSKAOYAN.COM

3

初始化操作

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 静态数组存放栈中元素
    ElemenType data[MaxSize]; // 栈顶指针
    int top;
} SqStack;

```

```

// 初始化栈
void InitStack(SqStack &S){           // 初始化栈顶指针
    S.top=-1;
}

```

```

void testStack() {
    SqStack S; // 声明一个顺序栈(分配空间)
    InitStack(S); // ...后续操作...
}

```

内存

判断栈空

bool StackEmpty(SqStack S){
 if(S.top== -1) // 栈空
 return true;
 else // 不空
 return false;}

王道考研/CSKAOYAN.COM

4

进栈操作

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 定义栈的结构
    ElemenType data[MaxSize]; // 静态数组存放栈中元素
    int top;                 // 栈顶指针
} SqStack;

```

// 新元素入栈

```

bool Push(SqStack &S, ElemenType x){
    if(S.top==MaxSize-1)      // 栈满, 报错
        return false;
    S.top = S.top + 1;         // 指针先加1
    S.data[S.top]=x;          // 新元素入栈
    return true;
}

```

注意: 错误写法!

真的很危险

内存

王道考研/CSKAOYAN.COM

5

进栈操作

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 定义栈的结构
    ElemenType data[MaxSize]; // 静态数组存放栈中元素
    int top;                 // 栈顶指针
} SqStack;

```

// 新元素入栈

```

bool Push(SqStack &S, ElemenType x){
    if(S.top==MaxSize-1)      // 栈满, 报错
        return false;
    S.top = S.top + 1;         // 指针先加1
    S.data[S.top]=x;          // 新元素入栈
    return true;
}

```

等价

内存

王道考研/CSKAOYAN.COM

6

出栈操作

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 静态数组存放栈中元素
    ElemenType data[MaxSize]; // 栈顶指针
    int top;                 // 栈顶指针
} SqStack;

// 出栈操作
bool Pop(SqStack &S, ElemenType &x){
    if(S.top == -1)         // 栈空, 报错
        return false;
    x = S.data[S.top];      // 栈顶元素先出栈
    S.top = S.top - 1;       // 指针再减1
    return true;
}

// 错误写法
S.top = S.top - 1;
x = S.data[S.top];

```

注意: 错误写法!

真的很危险

内存

top = 8

top

数据还残留在内存中, 只是逻辑上被删除了

王道考研/CSKAOYAN.COM

7

读栈顶元素操作

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 静态数组存放栈中元素
    ElemenType data[MaxSize]; // 栈顶指针
    int top;                 // 栈顶指针
} SqStack;

// 出栈操作
bool Pop(SqStack &S, ElemenType &x){
    if(S.top == -1)         // 栈空, 报错
        return false;
    x = S.data[S.top - 1];   // 先出栈, 指针再减1
    return true;
}

// 读栈顶元素
bool GetTop(SqStack S, ElemenType &x){
    if(S.top == -1)         // 栈空, 报错
        return false;
    x = S.data[S.top];       // x记录栈顶元素
    return true;
}

```

唯一区别

内存

top = 8

top

唯一区别

王道考研/CSKAOYAN.COM

8

另一种方式

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 静态数组存放栈中元素
    ElemenType data[MaxSize]; // 栈顶指针
    int top;
} SqStack;

// 初始化栈
void InitStack(SqStack &S){
    S.top=0;               // 初始化栈顶指针
}

void testStack() {
    SqStack S; // 声明一个顺序栈(分配空间)
    InitStack(S);
    // ...后续操作...
}
    
```

内存

top = 0

top

空栈

王道考研/CSKAOYAN.COM

9

另一种方式

```

#define MaxSize 10           // 定义栈中元素的最大个数
typedef struct{           // 静态数组存放栈中元素
    ElemenType data[MaxSize]; // 栈顶指针
    int top;
} SqStack;

S.data[S.top] = x;           // 等价
S.top = S.top + 1;           // 进栈

S.top = S.top - 1;           // 出栈
x=S.data[–S.top];
    
```

内存

top = 5

top

top 指向下一个可以插入的位置

不会犯错

注意审题啊喂！

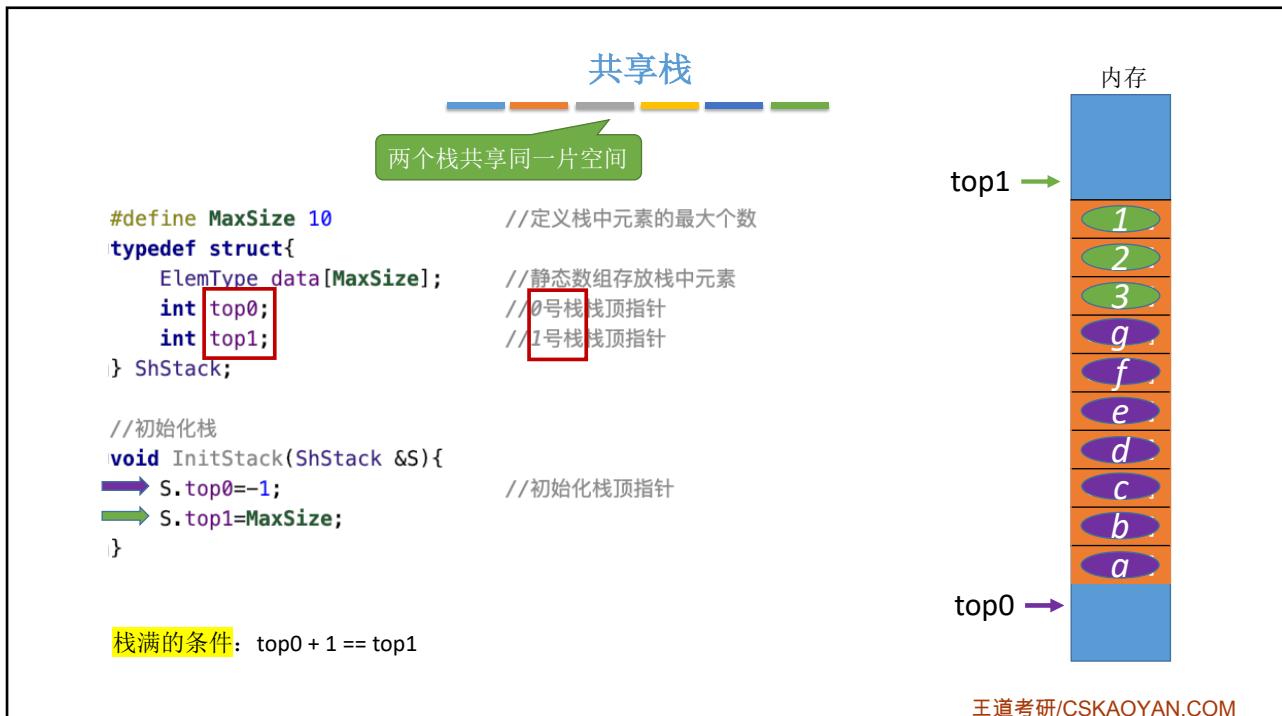
题目不对劲

栈满的条件: top == MaxSize

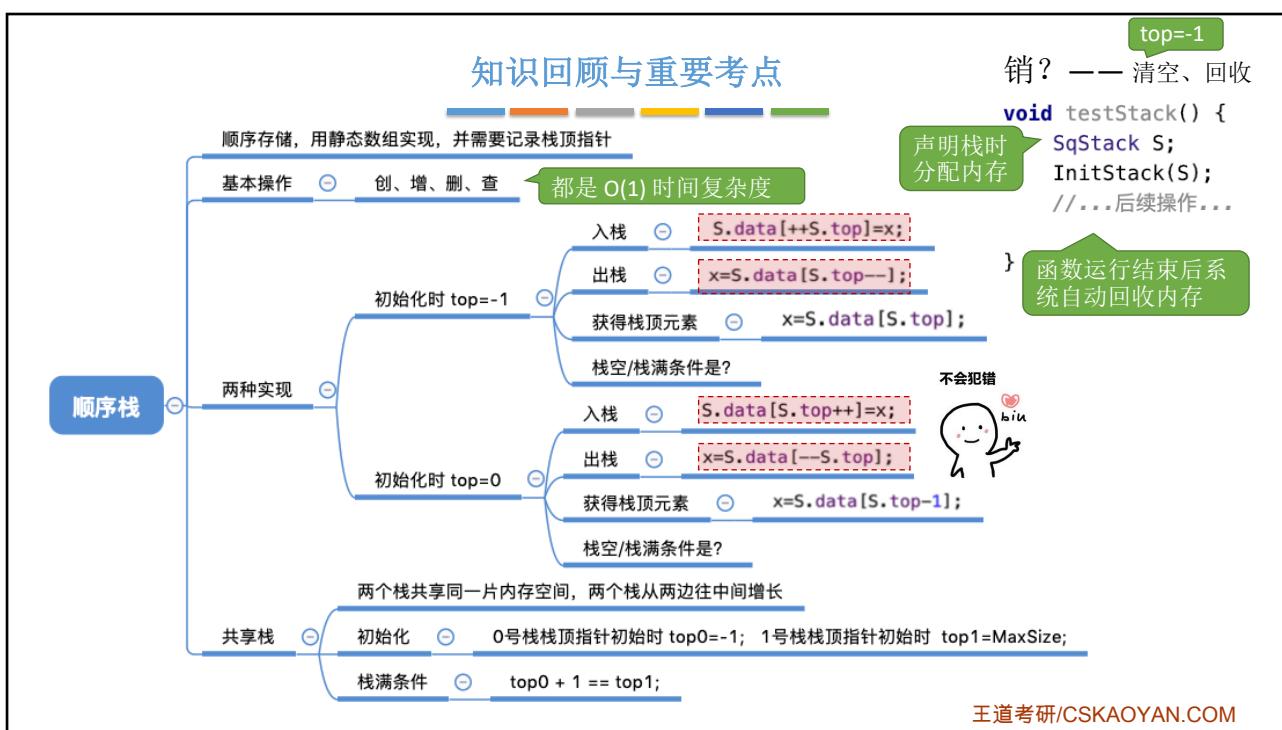
顺序栈的缺点:
栈的大小不可变

王道考研/CSKAOYAN.COM

10



11



12