3.2.1

```c
#include <stdio.h>
#include <stdlib.h>

#define SElemType int
#define MAXSIZE 100

typedef struct
{
    int top[2], bot[2];
    SElemType *v;
    int m
} DblStack; // 双栈

// 初始化双栈
void InitDblStack(DblStack *s)
{
    s->m = MAXSIZE;
    s->top[0] = s->bot[0] = -1;
    s->top[1] = s->bot[1] = s->m;
    s->v = (SElemType *)malloc(MAXSIZE * sizeof(SElemType));
    if (!s->v)
    {
        printf("InitDblStack:分配内存失败\n");
        exit(0);
    }
}

//  判断双栈是否为空
_Bool isStackEmpty(DblStack *s)
{
    return s->top[0] == -1 && s->top[1] == s->m;
}

// 判断双栈是否已满
_Bool isStackFull(DblStack *s)
{
    return s->top[0] + 1 == s->top[1];
}

// 入栈
void Push(DblStack *s, int i, SElemType e)
{
    if (i < 0 || i > 1)
    {
        printf("Push:栈号错误\n");
        exit(0);
    }
    if (isStackFull(s))
    {
        printf("Push:栈已满\n");
```

```
            exit(0);
        }
        if (i == 0)
        {
            s->v[++s->top[0]] = e;
        }
        else
        {
            s->v[--s->top[1]] = e;
        }
    }

    // 出栈
    void Pop(DblStack *s, int i, SElemType *e)
    {
        if (i < 0 || i > 1)
        {
            printf("Pop:栈号错误\n");
            exit(0);
        }
        if (isStackEmpty(s))
        {
            printf("Pop:栈已空\n");
            exit(0);
        }
        if (i == 0)
        {
            *e = s->v[s->top[0]--];
        }
        else
        {
            *e = s->v[s->top[1]++];
        }
    }
```

3.2.5

① ACD ②

```
    _Bool isLegal(char *s)
    {
        int push_count = 0;
        int pop_count = 0;
        while(*s)
        {
            if (*s=='I')
                push_count++;
            if (*s=='O')
                pop_count++;
            if (pop_count>push_count)
                return false;
```

```
            s++;
        }
        return true;
    }
```