第 2 章 算法知识、栈、队列

2.1 算法知识

算法是对特定问题求解步骤的描述，算法有 5 个重要特征。

（1）有穷性：对于任意一组合法的输入，算法能在有限的时间内完成。

（2）确定性：算法的每一步有明确的定义，没有歧义。

（3）输入：算法应有 0 个或多个输入。（一般 0 个输入指的是有些算法的输入是嵌入到算法之中的）

（4）输出：算法应有 1 个或者多个输出。

（5）可行性：算法必须遵循特定条件下的解题规则，算法描述的操作都应该是特定规则中允许使用的、可执行的，并通过执行有限次来实现。

常见的算法有：穷举、高精度计算、排序、递推、递归、贪心、分治、搜索、动态规划等。

1、算法复杂度

影响程序运行时间的因素：

（1）计算机的计算速度；　（2）计算数据的大小；　（3）算法的效率；

一个算法的评价一般从时间复杂度和空间复杂度来考虑。

时间复杂度：指算法所需要的计算工作量，用算法所执行的基本运算次数来度量。常见的时间复杂度有：常数阶 O(1)，对数阶 $O(\log_2 n)$，线性阶 O(n)，线性对数阶 $O(n* \log_2 n)$，平方阶 $O(n^2)$，立方阶 $O(n^3)$，指数阶 $O(2^n)$等，上述时间复杂度随着问题规模 n 的增加，时间复杂度不断增加，算法效率降低。

时间复杂度的计算步骤：

求解算法的时间复杂度的具体步骤是：

1、找出算法中的基本语句：

算法中执行次数最多的那条语句就是基本语句，通常是最内层循环的循环体。

2、计算基本语句的执行次数的**数量级**：

（1）只需计算基本语句执行次数的数量级，这就意味着只要保证基本语句执行次数的函数中的**最高次幂正确**即可，可以忽略所有低次幂和最高次幂的系数。

（2）这样能够简化算法分析，并且使注意力集中在最重要的一点上：增长率。

3、用大 O 记号表示算法的时间性能：

（1）将基本语句执行次数的数量级放入大 O 记号中。

（2）如果算法中包含嵌套的循环，则基本语句通常是最内层的循环体，如果算法中包含并列的循环，则将并列循环的时间复杂度相加。例如：

```
for(i=1;i<=n;i++)      x++;
for(i=1;i<=n;i++){
    for(j=1;j<=n;j++)      x++;
}
```

第一个 for 循环的时间复杂度为 O(n)，第二个 for 循环的时间复杂度为 $O(n^2)$，则整个算法的时间复杂度为 $O(n+n^2)=O(n^2)$。

下面按从快到慢的顺序列出了你经常会遇到的 5 种大 0 运行时间。（时间复杂度只需要

计算到对应的数量级，不需要计算到具体的值）

（1）O(log n)，也叫对数时间，这样的算法包括二分查找。

（2）O(n)，也叫线性时间，这样的算法包括简单查找。

（3）O(n * log n)，这样的算法包括快速排序——一种速度较快的排序算法。

（4）O(n^2)，这样的算法包括第选择排序、冒泡排序——一种速度较慢的排序算法。

（5）O(n!)，这种情况很少见，n 越大，所消耗的时间就越慢。

大 O 表示法是一种特殊的表示法，指出了算法的速度有多快。例如，假设列表包含 n 个元素。简单查找需要检查每个元素，因此需要执行 n 次操作，这个运行时间为 O(n)。

一般不特别说明，讨论的时间复杂度均是最坏情况下的时间复杂度。这就保证了算法的运行时间不会比任何其他情况更长。

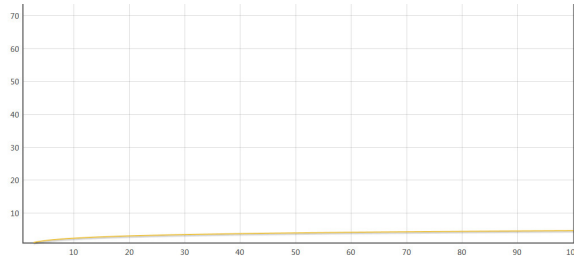


图 O(logn)

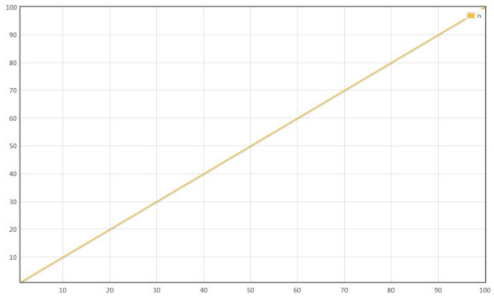


图 O(n)

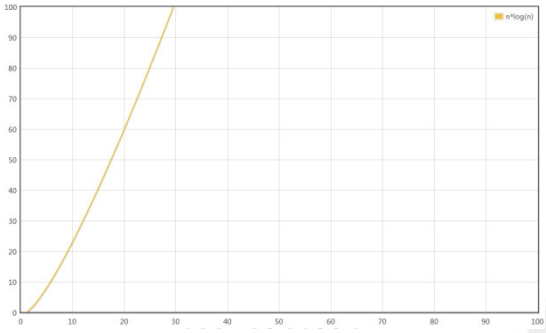


图 O(n * log(n))

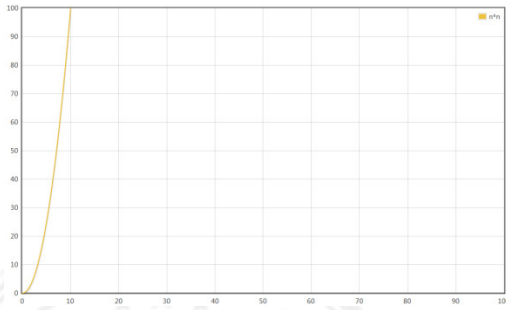


图 O(n * n)

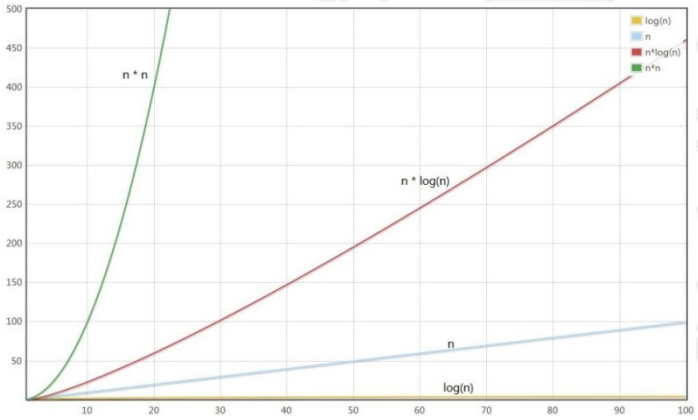


图 四种算法效率对比

空间复杂度：指执行这个算法所需要的内存空间。

2、常见算法的算法复杂度

（1）查找算法

查找策略	时间复杂度	备注

顺序查找	O(n)	
二分查找	$O(\log_2 n)$	要求数列有序
插值查找	$O(\log_2 n)$	要求数列有序且相对均匀

（2）排序算法

排序	算法复杂度
冒泡排序	$O(n^2)$
直接插入	$O(n^2)$
快速排序、堆排序、归并排序	$O(n\log_2 n)$
选择排序	$O(n^2)$
桶排序	O(n)

【NOIP2018 普及组】8.以下排序算法中，不需要进行关键字比较操作的算法是（　　）。

A. 基数排序　　B. 冒泡排序

C. 堆排序　　D. 直接插入排序

答案：A

解析：基数排序，根据键值的每位数字来分配桶；

冒泡排序、堆排序和插入排序都是基于两两元素关键字比较的排序。基数排序是直接将元素通过某些规则放到数组的相应位置，不是基于两两元素关键字比较的排序。

【NOIP2018 提高组】5. 设某算法的时间复杂度函数的递推方程是 T(n) = T(n - 1) + n（n 为正整数）及 T(0) = 1，则该算法的时间复杂度为（ ）。

A. O(log n)　B. O(n log n)　C. O(n)　D. $O(n^2)$

答案：D

解析：

$$
\begin{aligned}
T(n) &= T(n-1) + n \\
&= T(n-2) + (n-1) + n \\
&= T(n-3) + (n-2) + (n-1) + n \\
&= T(1) + 2 + \cdots\cdots + (n-2) + (n-1) + n \\
&= T(0) + 1 + 2 + \cdots\cdots + (n-2) + (n-1) + n \\
&= 1 + n*(n+1) / 2 = n^2 / 2 + n / 2 + 1
\end{aligned}
$$

2.2 什么是数据结构？

1、数据结构的定义

数据结构是计算机存储、组织数据的方式。数据结构是指相互之间存在一种或多种特定关系的数据元素的集合。通常情况下，精心选择的数据结构可以带来更高的运行或者存储效率。

如：数组。

2、数组有什么特点？

（1）数组(array)的元素(element)或项(item) 的类型是相同的

（2）数组对某元素的存取是 O(1)时间

（3）数组的插入、删除操作是 O(n)时间

由于数组通常的插入、删除操作是 O(n)时间的，在某些特定的条件下就显得低效了。因此我们通过对数组元素操作的限制，来达到操作的高效---算法优化的突破点。

常见的“订制”数组有：栈、队列、堆等，它们操作的时候效率都很高。

注：虽然栈、队列、堆可以不用数组实现，但 NOIP 的实践中，用数组实现更简单实用。

2.3 栈（Stack）

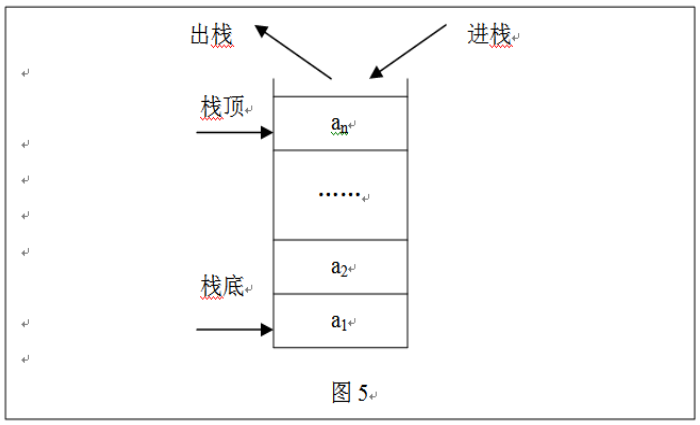


图 5

1、栈的特点

（1）栈(stack)是后进先出(last-in-first-out,LIFO)或先进后出(FILO)的

（2）栈有三个基础操作压入(push),弹出(pop),取数(getTop)操作都为 O(1)时间

（3）栈有一个计数器 counter 或栈顶指针

2、栈的基本操作

（1）建栈（init）：在使用栈之前，首先需要建立一个空栈，称建栈；

（2）压栈（push）：往栈顶加入一个新元素，称进栈（压栈）；

（3）出栈（pop）：删除栈顶元素，称出栈（退栈、弹出）；

（4）取栈顶（gettop）：查看当前的栈顶元素，称读栈；

（5）测试栈（empty）在使用栈的过程中，还要不断测试栈是否为空或已满，称为测试栈；

（6）显示栈（display）：输出栈的所有元素；

（7）释放栈（setnull）：清空栈；

3、栈的操作代码

初始化、入栈、出栈、显示栈！

```
#include <iostream>
//常量：栈的长度
```

```
#define MAXN 5
using namespace std;

int stack[MAXN];//数组模拟栈
int top = -1;//初始化栈指针

//出栈
int pop(){
    int r;
    if(top < 0){
        r = -1;
        cout<<"栈空! "<<endl;
    }else{
        r = stack[top];
        top--;
    }

    return r;
}

//入栈
void push(int value){
    if(top >= MAXN - 1){
        cout<<"栈满"<<endl;
    }else{
        top++;
        stack[top] = value;
    }
}

//显示栈
void display(){
    int i;
    for(i = top;i >= 0;i--){
        cout<<stack[i]<<" ";
    }
    cout<<endl;
}

int main(){
    int order;//指令
    int x,t;
    cout<<"输入指令: ";
    while(1 == 1){
        cout<<"1:入栈,2:出栈,3:显示栈!"<<endl;
        cin>>order;
        if(order == 1){
            cin>>x;
            push(x);
            display();
        }else if(order == 2){
            t = pop();
            if(t != -1){
                cout<<t<<"出栈!"<<endl;
                display();
            }
        }else if(order == 3){
            display();
        }
    }
   return 0;
}
```

4、栈的练习题

【noip2015 普及组】15. 今有一空栈 S，对下列待进栈的数据元素序列 a,b,c,d,e,f 依次进行进栈，进栈，出栈，进栈，进 栈，出栈的操作，则此操作完成后，栈 S 的栈顶元素为

（　　）

A. f　　B. c　　C. a　　D. b

答案：B

解析：画一个栈，按照题目所描述的操作模拟一遍。

【noip2017 提高组】2.　对于入栈顺序为 a，b，c，d，e，f，g 的序列，下列（　）不可能是合法的出栈序列。【不定项选择题】

A. a,b,c,d,e,f,g　　B. a,d,c,b,e,g,f　　C. a,d,b,c,g,f,e　　D. g,f,e,d,c,b,a

答案：C

解析：画一个栈，逐个选项判断模拟是否可行。

【noip2012 普及组】18．在程序运行过程中，如果递归调用的层数过多，会因为（　）引发错误。

A．系统分配的栈空间溢出　　B．系统分配的堆空间溢出

C．系统分配的队列空间溢出　　D．系统分配的链表空间溢出

答案：A

解析：递归是用栈这种数据结构来实现的。

执行时，外层的函数先进栈，内层的函数后进栈。内层的函数把结果返回给外层的函数并出栈。

【noip2010 普及组】15．元素 R1、R2、R3、R4、R5 入栈的顺序为 R1、R2、R3、R4、R5。如果第一个出栈的是 R3，那么第五个出栈的不可能是（　　）。

A．R1　　B．R2　　C．R4　　D．R5

答案：B

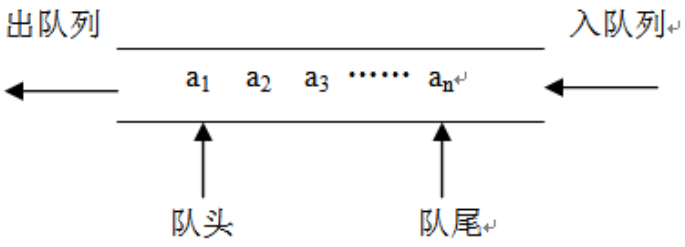
答案：C

解析：画一个栈，逐个选项判断模拟是否可行。

2.4 队列

1、什么是队列?

队列就是允许在一端进行插入，在另一端进行删除的线性表。允许插入的一端称为队尾，通常用一个队尾指针 r 指向队尾元素，即 r 总是指向最后被插入的元素；允许删除的一端称为队首，通常也用一个队首指针 f 指向排头元素的前面。初始时 f=r=0。



举例 1：到医院看病，首先需要到挂号处挂号，然后，按号码顺序救诊。

举例 2：乘坐公共汽车，应该在车站排队，车来后，按顺序上车。

结论：在队列这种数据结构中，最先插入在元素将是最先被删除；反之最后插入的元素将最后被删除，因此队列又称为“先进先出”（FIFO—first in first out）的线性表。

2、队列的基本操作

（1）新建队列

（2）入队

（3）出队

（4）判断队列是否为空

（5）判断队列是否为满

（6）显示队列元素

3、队列操作的代码实现

```
#include <iostream>
#define MAXN 5
using namespace std;
//队列
int queue[MAXN] = {0};
//头指针
int front = 0;
//尾指针
int rear = 0;

//入队
void addqueue(int value){
    if(rear >= MAXN){
        cout<<"队满!"<<endl;
    }else{
        queue[rear] = value;
        rear++;
    }
}

//出队
int delqueue(){
    int r;
    if(front == rear){
        cout<<"队空"<<endl;
        r = -1;
    }else{
        r = queue[front];
        front++;
    }

    return r;
}

//显示队
void display(){
    int i;
    for(i = front;i < rear;i++){
        cout<<queue[i]<<" ";
    }
    cout<<endl;

}
```

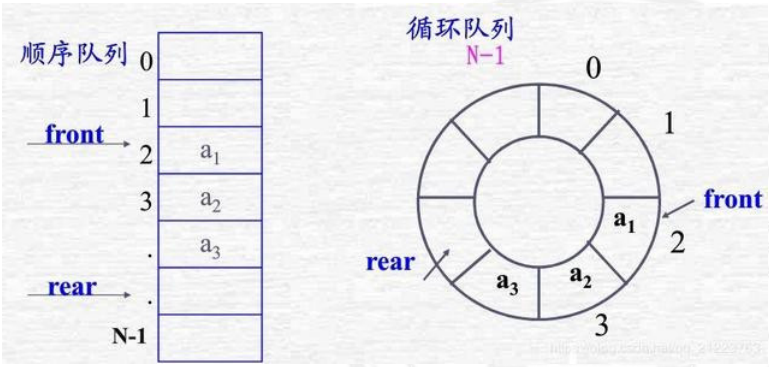
```
int main(){
    int order;//口令
    int value,t;
    cout<<"输入指令"<<endl;
    while(1 == 1){
        cout<<"1 入队,2 出队,3 显示！"<<endl;
        cin>>order;
        if(order == 1){
            cin>>value;
            addqueue(value);
            display();
        }else if(order == 2){
            t = delqueue();
            if(t != -1){
                cout<<t<<"已经出队"<<endl;
                display();
            }else{
                cout<<"队列已空"<<endl;
            }
        }else if(order == 3){
            display();
        }else{
            cout<<"口令错误！"<<endl;
        }
    }
}
```

10	20	30		
0	1	2	3	4

front=-1　　rear=2

4、循环队列的使用

上述通过数组来实现队列的方法，如果进行插入一次，删除一次的操作，只要数组使用过一遍数组就会被用光。当数组仿真队列的元素全部出队以后，队的首部就会出现需多空位无法使用，导致空间浪费。



解决方法：

将线型数组模拟成环形数组。但无论在线型数组中还是环形数组中，都有一个问题：无法判断队空还是队满；

因为队空的条件是：

front==rear;

队满的条件也是：

front==rear;

为了解决这个问题，**我们在入队时少用一个数据元素空间**。

这样，判断队满的方式为：(rear + 1) % MAXN == front。

判断队空的方式为：rear == front。

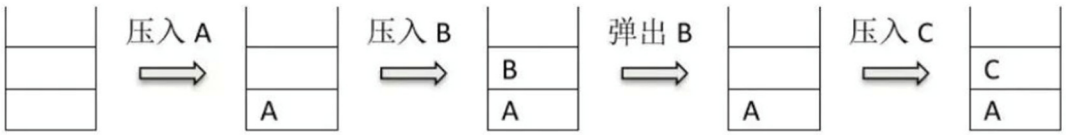
循环队列求队列元素个数：(rear-front+n)%n。

```
#include <iostream>
#define MAXN 5
using namespace std;
//队列
int queue[MAXN] = {0};
//头指针
int front = 0;
//尾指针
int rear = 0;

//入队
void addqueue(int value){
    //元素从未出队的情况下，队满
    if(front == 0 && (rear + 1) % MAXN == front){
        cout<<"队满"<<endl;
    //有元素出队，队满
    }else if((rear + 1) % MAXN == front){
        cout<<"队满"<<endl;
    }else{
        queue[rear] = value;
        rear = (rear + 1) % MAXN;
    }
}

//出队
int delqueue(){
    int r;
    if(front == rear){
        cout<<"队空"<<endl;
        r = -1;
    }else{
        r = queue[front];
        queue[front] = -1;//出队标记
        front = (front + 1) % MAXN;
    }

    return r;
}

//显示队
void display(){
    if(front == rear){
        cout<<"队空"<<endl;
    }else{
        int i = front;
        do{
            cout<<queue[i]<<" ";
            i = (i + 1) % MAXN;
        }while(i != rear);
    }
    cout<<endl;
}

int main(){
    int order;//口令
    int value,t;
    cout<<"输入指令"<<endl;
    while(1 == 1){
        cout<<"1 入队,2 出队,3 显示！"<<endl;
        cin>>order;
        if(order == 1){
            cin>>value;
            addqueue(value);
            display();
```

```
        }else if(order == 2){
            t = delqueue();
            if(t != -1){
                cout<<t<<"已经出队"<<endl;
                display();
            }else{
                cout<<"队列已空"<<endl;
            }
        }else if(order == 3){
            display();
        }else{
            cout<<"口令错误！"<<endl;
        }
    }
}
```

5、队列练习题

【noip2018 普及组】15.下图中所使用的数据结构是（　　）。



A.哈希表　　B. 栈　　C. 队列　　D. 二叉树

答案：B

【noip2012 普及组】2.（　　）是一种先进先出的线性表。

A. 栈　　B. 队列　　C. 哈希表（散列表）　　D. 二叉树

答案：B

【noip2011 普及组】11. 广度优先搜索时，需要用到的数据结构是（　　）。

A. 链表　　B. 队列　　C. 栈　　D. 散列表

答案：B

2.5 栈练习题

1、若已知一个栈的入栈顺序是 1，2，3，…，n，其输出（出栈）序列为 P1，P2，P3，…，Pn，若 P1 是 n，则 Pi 是(　　)。

A)i　　B)n-1　　C)n-i+1　　D)不确定

2、以下哪一个不是栈的基本运算(　　)。

A)删除栈顶元素　　B)删除栈底的元素

C)判断栈是否为空　　D)将栈置为空栈

3、设栈 S 的初始状态为空,现有 5 个元素组成的序列{1,2,3,4,5},对该序列在 S 栈上依次进行如下操作(从序列中的 1 开始,出栈后不再进栈)：进栈,进栈,进栈,出栈,进栈,出

栈,进栈,问出栈的元素序列是:_____,栈中元素个数_____，栈顶元素为:_____。

4、已知元素（8，25，14，87，51，90，6，19，20），问这些元素以怎样的顺序进入栈，才能使出栈的顺序满足：8 在 51 前面；90 在 87 的后面；20 在 14 的后面；25 在 6 的前面；19 在 90 的后面。（　　）。

A　20，6，8，51，90，25，14，19，87

B　51，6，19，20，14，8，87，90，25

C　19，20，90，8，6，25，51，14，87

D　6，25，51，8，20，19，90，87，14

E　25，6，8，51，87，90，19，14，20

5、[多选]设栈 S 的初始状态为空，元素 a，b，c，d，e，f，g 依次入栈，以下出栈序列不可能出现的有

（　　）。

A. a，b，c，e，d，f，g

B. b，c，a，f，e，g，d

C. a，e，c，b，d，f，g

D. d，c，f，e，b，a，g

E. g，e，f，d，c，b，a

6、某个车站呈狭长形，宽度只能容下一台车，并且只有一个出入口（出入同一口）。已知某时刻该车站状态为空，从这一时刻开始的出入记录为:“进，出，进，进，进，出，出，进，进，进，出,出”。假设车辆入站的顺序为 1，2，3，……，则车辆出站的顺序为（　　）。

A. 1，2，3，4，5　　B. 1，2，4，5，7　　C. 1，4，3，7，6

D. 1，4，3，7，2　　E. 1，4，3，7，5

7、设栈 S 的初始状态为空，元素 a，b，c，d，e 依次入栈，以下出栈序列不可能出现的有（　　）。

A. a，b，c，e，d　　B. b，c，a，e，d

C. a，e，c，b，d　　D. d，c，e，b，a

8、设栈 S 的初始状态为空，元素 a，b，c，d，e，f 依次入栈 S，出栈的序列为 b，d，c，f，e，a，则栈 S 的容量至少应该是（　　）。

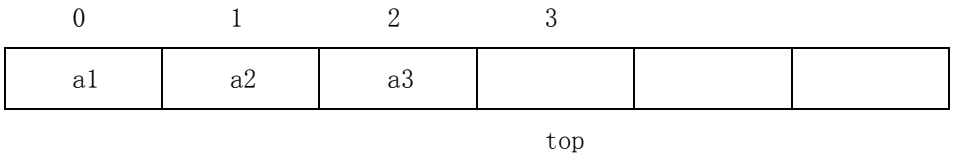
A. 6　　B. 5　　C. 4　　D. 3　　E. 2

9、设有一顺序栈 S，元素 s1,s2,s3,s4,s5,s6 依次进栈，如果有 6 个元素出栈的顺序是 s2,s3,s6,s5,s4,s1，则栈的容量至少是（　　）。

A、2　　B、3　　C、4　　D、5

10、设有一顺序栈已含 3 个元素，如下图所示，元素 a4 正等待进栈。那么下列 4 个序

列中不可能出现的出栈序列是（　　）。



A、a3,a1,a4,a2　　B、a3,a2,a4,a1

C、a3,a4,a2,a1　　D、a4,a3,a2,a1

11、若一个栈的输入序列为 1,2,3,….n，输出序列的第一个元素是 i，则第 j 个输出元素是（　　）。

A、i - j - 1　　B、i-j　　C、j-i+1　　D、不确定

12、设一个站的输入序列是 1,2,3,4,5，则下列序列中，是栈的合法输出序列的是(　　)。

A、5,1,2,3,4　　B、4,5,1,3,2　　C、4,3,1,2,5　　D、3,2,1,5,4

参考答案：

1.C　2.B　3.出栈序列为{3，4}，栈顶指针值为 3，栈顶元素为 5。　4.D　5.CE

6.C　7.C　8.D　9.C　10.A

11.D 12.D

2.6 队列练习题

1. 已知队列（13，2，11，34，41，77，5，7，18，26，15），第一个进入队列的元素是 13，则第五个出队列的元素是（　　）。

A. 5　　B. 41　　C. 77　　D. 13　　E. 18

2.设栈 S 和队列 Q 初始状态为空，元素 e_1，e_2，e_3，e_4，e_5，e_6依次通过栈 S，一个元素出栈后即进入队列 Q，若出队顺序为 e_2，e_4，e_3，e_6，e_5，e_1，则栈 S 的容量至少应该为_____。

3.设循环队列中数组的下标范围是 1..n，其头尾指针分别为 f 和 r，则其元素个数为：_____。

4. 若用一个大小为 6 的数组来实现循环队列,且当前 rear 和 front 的值分别为 0 和 3,当从队列中删除一个元素,再插入两个元素后,rear 和 front 的值分别为多少?（　）

A. 4 和 2　　B.2 和 4　　C. 3 和 0　　D.0 和 3

5. 在具有 n 个单元的顺序存储的循环队列中，假定 front 和 rear 分别为队头指针和队尾指针，则判断队满的条件为(　　)。

A. rear%n= = front　　B. (front+1) %n= = rear

C. rear%n -1= = front　　D. (rear+1)%n= = front

7. 在具有 n 个单元的顺序存储的循环队列中，假定 front 和 rear 分别为队头指针和队尾指针，则判断队空的条件为(　　)。

A. rear%n= = front　　B. front+1= rear

C. rear= = front　　D. (rear+1)%n= front

参考答案：

1.B　2.3　3. (r-f+n) mod n　4.B　5.D　6.C

2.7 栈与队列作业题

1.　栈中元素的进出原则是(　　)

A. 先进先出　B. 后进先出　C. 栈空则进　D. 栈满则出

2. 数组 Q[n]用来表示一个循环队列，f 为当前队列头元素的前一位置，r 为队尾元素的位置，假定队列中元素的个数小于 n，计算队列中元素的公式为(　　)

A.r－f;　　B. (n＋f－r) % n;　　C.n＋r－f;　　D. (n＋r－f) % n

3.　设有 4 个数据元素 a1、a2、a3 和 a4，对他们分别进行栈操作或队操作。在进栈或进队操作时，按 a1、a2、a3、a4 次序每次进入一个元素。假设栈或队的初始状态都是空。

现要进行的栈操作是进栈两次，出栈一次，再进栈两次，出栈一次；这时，第一次出栈得到的元素是__(1)__，第二次出栈得到的元素是__(2)__；类似地，考虑对这四个数据元素进行的队操作是进队两次，出队一次，再进队两次，出队一次；这时，第一次出队得到的元素是__(3)__，第二次出队得到的元素是__(4)__。经操作后，最后在栈中或队中的元素还有__(5)__个。

(1)(　　)　A、a1　B、a2　C、a3　D、a4

(2)(　　)　A、a1　B、a2　C、a3　D、a4

(3)(　　)　A、a1　B、a2　C、a3　D、a4

(4)(　　)　A、a1　B、a2　C、a3　D、a4

(5)(　　)　A、1　B、2　C、3　D、0

4.　栈是一种线性表，它的特点是__(1)__。设用一维数组 A[1,…,n]来表示一个栈，A[n]为栈底，用整型变量 T 指示当前栈顶位置，A[T]为栈顶元素。往栈中推入 (PUSH) 一个新元素时，变量 T 的值__(2)__；从栈中弹出 (POP) 一个元素时，变量 T 的值 __(3)__ 。设栈空时，有输入序列 a，b，c，经过 PUSH，POP，PUSH，PUSH，POP 操作后，从栈中弹出的元素的序列是__(4)__，变量 T 的值是__(5)__。

供选择的答案：

(1)(　) A.先进先出　B 后进先出　C 进优于出　D 出优于进　E 随机进出

(2)(　) A. 加 1　B.减 1　C.不变　D.清 0　E. 加 2　F.减 2

（3）（　）A. 加 1　　B. 减 1　　C. 不变　　D. 清 0　　E. 加 2　　F. 减 2

（4）（　）A. a, b　　B. b, c　　C. c, a　　D. b, a　　E. c, b　　F. a, c

（5）（　）A. n+1　　B. n+2　　C. n　　D. n-1　　E. n-2

5. 在做进栈运算时，应先判别栈是否（1）；在做退栈运算时，应先判别栈是否（2）。当栈中元素为 n 个，做进栈运算时发生上溢，则说明该栈的最大容量为（3）。

为了增加内存空间的利用率和减少溢出的可能性，由两个栈共享一片连续的内存空间时，应将两栈的（4）分别设在这片内存空间的两端，这样，只有当（5）时，才产生上溢。

供选择的答案：

（1）（　）A、空　　B、满　　C、上溢　　D、下溢

（2）（　）A、空　　B、满　　C、上溢　　D、下溢

（3）（　）A、n-1　　B、n　　C、n+1　　D、n/2

（4）（　）A、长度　　B、深度　　C、栈顶　　D、栈底

（5）（　）A、两个栈的栈顶同时到达栈空间的中心点

B、其中一个栈的栈顶到达栈空间的中心点

C、两个栈的栈顶在达栈空间的某一位置相遇

D、两个栈均不空，且一个栈的栈顶到达另一个栈的栈底

6. 一个栈的入栈序列是 1，2，3，4，5，则栈的不可能输出序列是（　　）。

A. 3，5，4，2，1

B. 3，2，4，5，1

C. 1，2，3，4，5

D. 5，4，3，1，2

7. 一个队列的入队序列是 1，3，5，7，9，则出队的输出序列只能是（　　）

A. 9，7，5，3，1

B. 1，3，5，7，9

C. 1，5，9，3，7

D. 9，5，1，7，3

8. 设循环队列中数组的下标范围是 1～n，其头尾指针分别为 f 和 r，则其元素个数为（　）

A. r-f　　B. r-f+1　　C. (r-f)% n+1　　D. (r-f+n)% n

9. 设数组 data[m]作为循环队列 SQ 的存储空间，front 为队头指针，rear 为队尾指针，则执行入队操作后其尾指针 rear 值为（　　）

A. rear=rear+1　　B. rear=(rear-1)%m

C. rear=(rear+1)%(m-1)　　D. rear=(rear+1)%m

10. 递归过程或函数调用时，处理参数及返回地址，要用一种称为（　　）的数据结构。

A. 队列　　B. 多维数组　　C. 栈　　D. 线性表

11. 若用大小为 6 的数组来实现循环队列，且当前 front 和 rear 的值分别为 0 和 4。当从队列中删除两个元素，再加入两个元素后，front 和 rear 的值分别为多少？（　　）

A. 2 和 6

B. 2 和 0

C. 6 和 2

D. 2 和 2

12. 栈是一种特殊的线性表，允许插入和删除运算的一端称为_____。不允许插入和删除运算的一端称为_____。

13. _____是被限定为只能在表的一端进行插入运算，在表的另一端进行删除运算的线性表。

14. 在具有 n 个单元的循环队列中，队满时共有_____个元素。

参考答案：

1.B　2.D

3.B D A B B

4.B B A F D

5.B A B D C

6.D　7.B　8.D　9.D　10.C

11.B　12.栈顶、栈底　13.队列　14.n-1