第三章：

1、 相比较普通线性表，栈的区别是什么？栈的元素的进出原则是什么？请简单解释下列术语：栈顶、栈顶元素、栈底、进栈、出栈。

答：栈是受限制的线性表，是只允许在一端进行插入和删除的线性表。

原则：只允许在一端插入和删除，后进先出。 栈顶：允许插入和删除的一端，即表尾。

栈顶元素：存储在栈顶的元素 栈底：栈顶的另一端，即表头。

入栈：插入元素 出栈：删除栈顶元素。

2、 顺序栈的定义：

```
typedet struct {
    ElemType *base; //动态分配方案，指针 base 指向栈元素空间的及地址；
    int top; //栈顶指针，实际是记录栈顶元素下一单元的下标，简称为栈顶指针；
    int stacksize; //栈空间的最大长度
}SqStack; SqStack s; //s 为顺序栈的变量
```

请根据上述定义，给出元素 e 进栈（注意判断栈是否已满）程序段，给出出栈的程序段（注意判断是否为空栈）。

解：进栈：

```
Status Push(SqStack &s, SElemType e)
{ if(s.top - s.base >= s.stacksize)
    return ERROR;
  *s.top++=e;
  return ok; }
```

出栈：

```
Status Pop(SqStack &s, SElemType &e)
{ if(s.top == s.base)
    return ERROR;
  e=*--s.top;
  return OK;
}.
```

3、 链栈的定义如下：

```
struct node {
    ElemType data; //栈元素
    Struct node *next; //指下一个结点
} *top=NULL; //链栈没有头结点，用指针 top 指向栈顶结点，top 为空时为空栈初始化链栈为空栈
```

根据上述定义，请给出元素 e 进栈的程序段，给出出栈的程序段（注意判断栈空）

解：进栈：

```
Status Push(LinkStack &S, SElemType e)
{ p=new SNode;
  if(!p) exit(OVERFLOW);
  p->data=e; p->next=S; S=p;
  return OK; }
```

出栈：

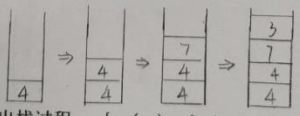
```
Status Pop(LinkStack &S, SElemType &e)
{ if(S==NULL) return ERROR;
  e=S->data; p=S; S=S->next;
  delete p; return Ok;}.
```

4、 给出将十进制数 2020 转换为 8 进制的过程，写出计算顺序，画出元素进栈过程。

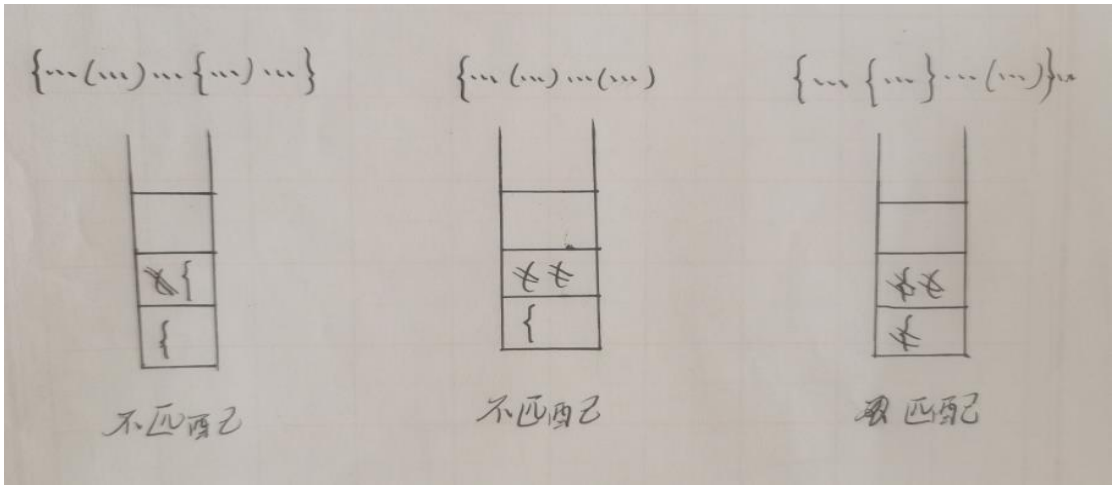
解：

	N	N/8	N%8
计算顺序↓	2020	252	4
	252	31	4
	31	3	7
	3	0	3

↑输出顺序

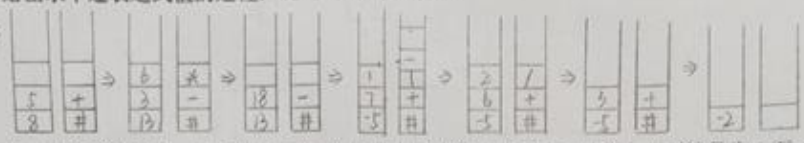


5、 检测下面括号是否匹配，给出结论及进出栈过程。 {…(…)…{…}…(…)} {…(…)…(…)…} {…{…}…(…)…}



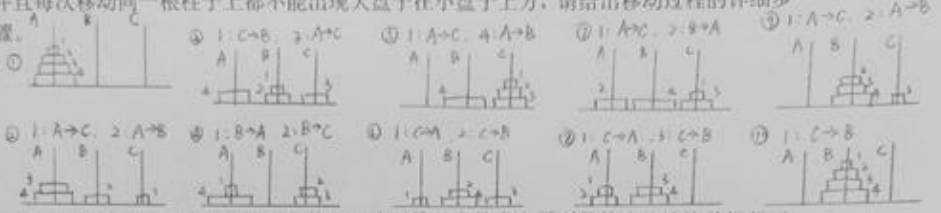
6、给出求下述表达式值的过程：#8+5-3*6+(7-1)/2#

解：



7、有三根相邻的柱子，标号为A,B,C，A柱子上按直径由小到大，且至上而下编号为1至4的不同大小的圆盘，要把所有盘子一个一个移动到柱子B上，柱子C可以作为中间辅助，并且每次移动同一根柱子上都不能出现大盘子在小盘子上方，请给出移动过程的详细步骤。

解：



8、相比较普通线性表，队列的区别是什么？队列的元素的进出原则是什么？请简单解释下列术语：队首元素、队尾元素、入队、出队。

答：区别：队列是只允许在表的一端进行插入，另一端删除的线性表。原则：先进先出

队首元素：允许删除的一端的元素；队尾元素：允许插入的一端所存储的元素

入队：插入元素　出队：删除元素

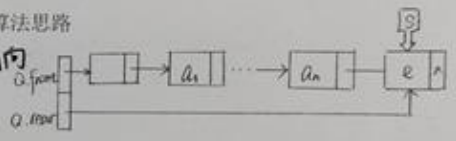
9、在链式存储的队列中插入一个新元素，描述算法思路

解：①为入队元素分配结点空间，用指针s指向

②将新结点数据域置为e，指针域为空

③将新结点插入队尾

④修改队尾指针为s，长度加1。



10、在链式存储的队列中删除一个元素，描述算法思路

解：①判断队列是否为空

②临时保存队头元素的空间，以备释放，为e赋值

③修改队头指针，指向下一个结点

④判断出队元素是否为最后一个元素，若是，则将队尾指针重新赋值，指向头结点

⑤释放原队头元素的空间，长度减1

11、简述解决队列顺序存储结构时出现的假溢出问题，给出具体解决方案。

解：将队列元素存放数组首尾相接

形成循环队列

12、在循环队列中，当用 front 记录队首元素，rear 记录队尾元素下一个空间，当 front=rear 时，循环队列可能为空，也可能是已满，怎么进行区分，给出解决方案，并给出不同解决方案中，判断队空和队满的条件。

解：(法一) 少用一个队中元素的空间　当 front == rear 时，队列为空。

(只使用 maxSize-1 个)　当 (rear+1)%maxSize == front 队列为满。

(法二) 附设一个队中元素个数变量　当变量 i = 0 时，队列为空。

当变量 i = maxSize 时，队列为满

13、 给出循环队列元素入队的算法思路

解：①判断队列是否已满

②将新元素插入队尾，作为新队尾元素。

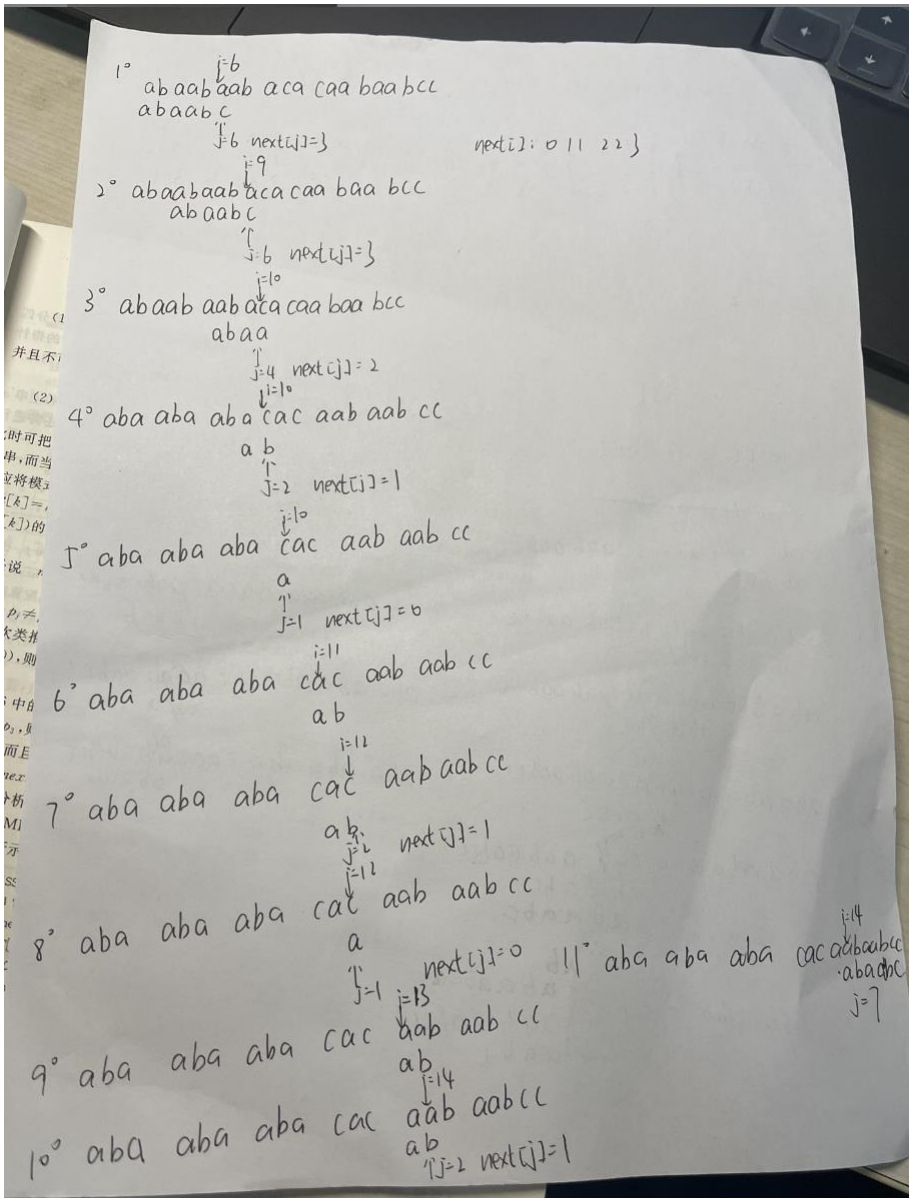
14、 给出循环队列元素出队的算法思路。

解：①判断队列是否为空

②不为空则删除当前队列中的队头元素。

第四章：

1、 已知字符串 S 为“abaabaabacacaabaabcc”，模式串 t 为“abaabc”，采用 KMP 算法进行匹配，求出 next[]值，并给出匹配过程。



$$\begin{pmatrix} 3 & 0 & 0 & 0 & 7 \\ 0 & 0 & -1 & 0 & 0 \\ -1 & -2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 \end{pmatrix}$$

1. 给出上图所示稀疏矩阵的三元组顺序表存储表示
2. 给出上图所示稀疏矩阵的转置矩阵及三元组顺序存储表示
3. 按照快速转置算法思路,填写下表。

col	1	2	3	4	5
num[col]					
cpot[col]					

4. num 数组和 cpot 数组存储数据的意义是什么?