**栈**：限定仅在表的一端进行操作的线性表。 先入后出。

**顺序栈：**栈的顺序存储结构，它利用一组地址连续的存储单元来依次存放由栈底到栈顶的所有元素，同时附加一个top指针来指示栈顶元素在顺序栈中的位置

typedef struct

{

datatype data[MAXSIZE];

int top;

}SeqStack;

**链栈:**克服顺序栈容易溢出的的问题 //和链表差不多

typedef struct node

{

datatype data[MAXSIZE];

struct node \*next;

}StackNode;

**队列**：只能在线性表的一端进行插入，而在线性表的另一端进行删除。只能删除的叫队头（front），只能插入的叫队尾（rear）

**顺序队列**：顺序存储结构，一用一组地址连续的存储单元存放队列中的元素。除了存放队列元素的一维数组外，还必须设置队头指针和队尾指针来分别指示当前的队头元素和队尾元素

typedef struct

{

datatype data[MAXSIZE];

int rear, front;

}SeQueue;

**循环队列**：解决假溢出的方法，首尾相连的圆环。但此时出现队空队满的条件为：q->rear = q->front;为了解决，采取损失一个数据元素的存储空间将队满条件改为：(q->rear+1)%MAXSIZE = q->front; 队空条件任然是:q->rear = q->front;

所以循环队列的元素个数为：(q-rear – q->front+MAXSIZE)%MAXSIZE;

入队操作: q->rear = (q->rear+1)%MAXSIZE;

q->data[q->rear] = x;

出队操作为：q->front = (q->front+1)%MAXSIZE;

x = q->data[q->front];

**链队列：**队列的链式存储。需要表示队头和队尾的指针，且只要一个（所以单独存放）

typedef struct node

{

datatypd data;

struct node \*next;

}QNode;

typedef struct

{

QNode \*front,\*rear; //把两个指针纳入一个结构体链队中

}LQueue;