队列：queue

是逻辑线性结构；

两个方向进行数据出/入，出或者入是固定在一端进行的。

特点：FIFO

与现实世界中的排队一致

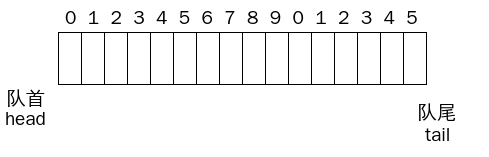
队列的实现：

存储空间；

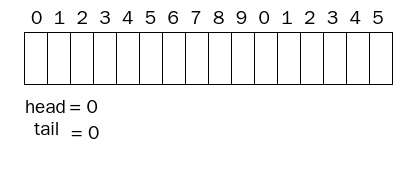
队首指针；

队尾指针；

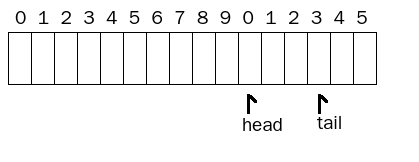
队首出数据，队尾入数据。



队列出时状态是空队列：队首指针、队尾指针指向同一空间。



设head和tail分别是一个队列的队首和队尾指针，则，该队列的有效元素个数为：tail-head



循环队列：

循环数组：

设有n个元素的数组，下标的取值范围：

0到n-1；对于数组元素的遍历基本算法：

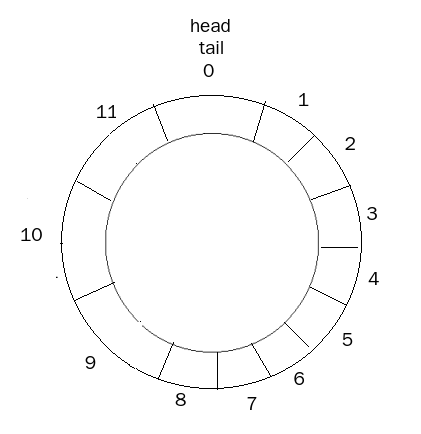
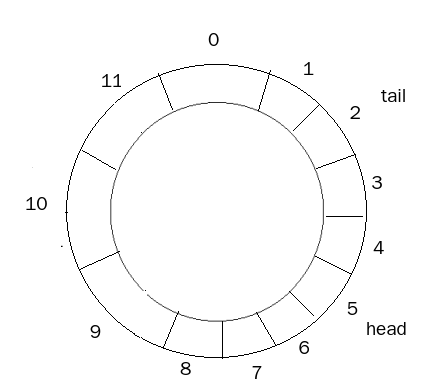
for(i = 0; i < n; i++)

...;

循环数组遍历基本算法： i = ++i % n

for(i = 0; i < n; i = (i + 1)%n)

...;



队列种有效元素的个数：(tail - head + MaxRoom) % MaxRoom

增加一个动态标志：

lastAction有两种取值：IN和OUT

只有上一次动作为入队列操作才有可能出现“队满”状态；

只有上一次动作为出队列操作才有可能出现“队空”状态；

由上述可知：在实现“入队列”和“出队列”运算时，除了出/入队列操作之外，应对赋相应的值：

入队列为IN，出队列为OUT。

由上可知：

队满条件 head == tail && lastAction == IN

队空条件 head == tail && lastAction == OUT

这说明，lastAction应该是控制队列的另一个控制量。

队列的程序实现：

队列控制头部；

队列空间。

typedef struct QUEUE{

USET\_TYPE \*data; //它将指向队列空间

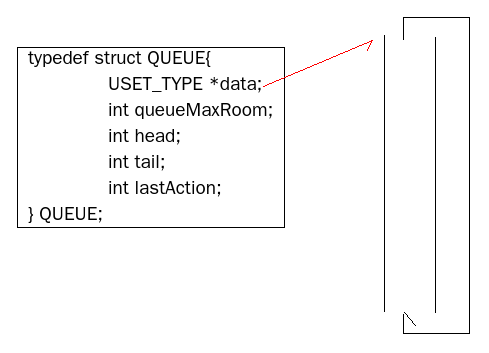
int queueMaxRoom; //队列最大空间

int head; //队首指针（下标）

int tail; //队尾指针（下标）

int lastAction; //末次操作类型（IN/OUT）

} QUEUE;

元素应存放到：data[i];