**队列操作**

问题描述

　　﻿队列操作题。根据输入的操作命令，操作队列（1）入队、（2）出队并输出、（3）计算队中元素个数并输出。

输入格式

　　第一行一个数字N。  
　　下面N行，每行第一个数字为操作命令（1）入队、（2）出队并输出、（3）计算队中元素个数并输出。

输出格式

　　若干行每行显示一个2或3命令的输出结果。注意：2.出队命令可能会出现空队出队（下溢），请输出“no”，并退出。

样例输入

7  
1 19  
1 56  
2  
3  
2  
3  
2

样例输出

19  
1  
56  
0  
no

数据规模和约定

1<=N<=50

C++代码：

#include "stdlib.h"

#include "stdio.h"

#include <iostream>

#include <queue>

using namespace std;

int main()

{

queue<int> q;

int n=0,num=0,i,element=0,j;

cin>>n;

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

{

cin>>num;

switch(num)

{

case 1:

{

cin>>element;

q.push(element);

}

break;

case 2:

{

if(q.empty()==0)

{

cout<<q.front()<<endl;

q.pop();

}

else

{

cout<<"no"<<endl;

return 0;

}

}

break;

case 3:

{

cout<<q.size()<<endl;

}

break;

}

}

return 0;

}

C代码：

#include<stdio.h>

#define MAX 100

typedef struct

{

long long int data[MAX];

int front,rear;

}SEQUEUE;

int inqueue(SEQUEUE \*q,int x)

{

if(q->front==(q->rear+1)%MAX)

{

return 0;

}

else

{

q->rear=(q->rear+1)%MAX;

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

}

}

int outqueue(SEQUEUE \*q)

{

if(q->front==q->rear)

{

printf("no\n");

return 0;

}

else

{

q->front=(q->front+1)%MAX;

printf("%lld\n",q->data[q->front]);

}

}

void countqueue(SEQUEUE \*q)

{

int sum;

sum=(q->rear-q->front+MAX)%MAX;

printf("%d\n",sum);

}

int main()

{

int flag,x,n;

SEQUEUE a,\*p;

p=&a;

a.front=-1;

a.rear=-1;

scanf("%d",&n);

while(n--)

{

scanf("%d",&flag);

switch(flag)

{

case 1: scanf("%lld",&x); inqueue(p,x); break;

case 2:

if(outqueue(p)==0)

{

return 0;

}

break;

case 3: countqueue(p); break;

}

}

return 0;

}

Java代码：

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner in = new Scanner(System.in);

int N = in.nextInt();

int []a = new int[N];

int k = 0;

for (int i=0; i<N; i++)

{

int n = in.nextInt();

if (n==1)

{

a[k++] = in.nextInt();

}else if(n==2)

{

if (k>0)

{

System.out.println(a[0]);

for (int j=0; j<k; j++)

a[j] = a[j+1];

k--;

}

else

{

System.out.println("no");

break;

}

}

else

System.out.println(k);

}

in.close();

}

}