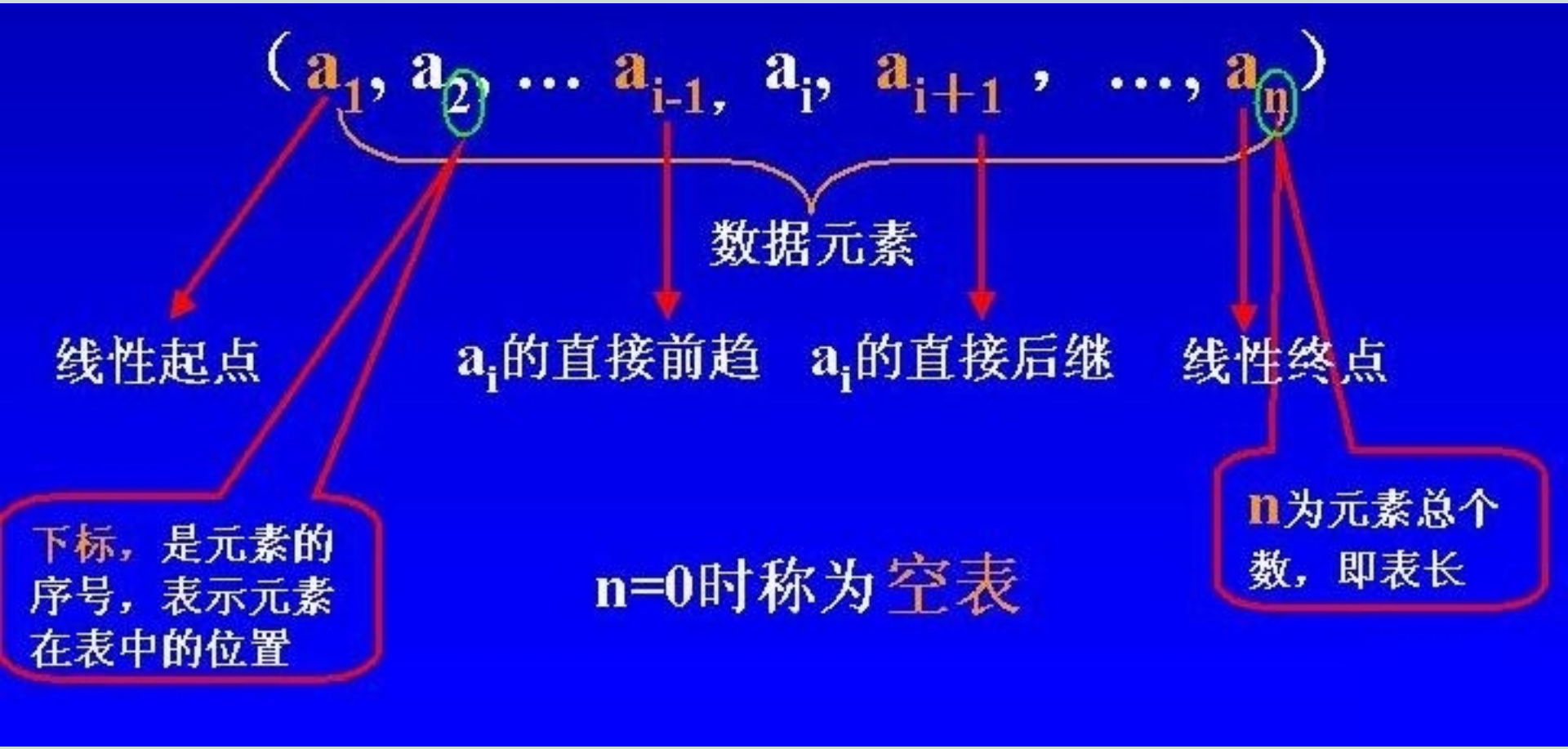
线　性　表

2．2 线性表

2.2.1 线性表的基本概念

线性表是n个元素的有限序列，它们之间的关系可以排成一个线性序列：$a_1, a_2, \ldots, a_i, \ldots, a_n$



在线性表上常用的运算有：

- 初始化
- 求长度
- 取元素
- 修改
- 前插
- 删除
- 检索
- 排序

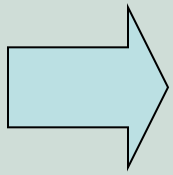
2.2.2 线性表的存储结构及其运算

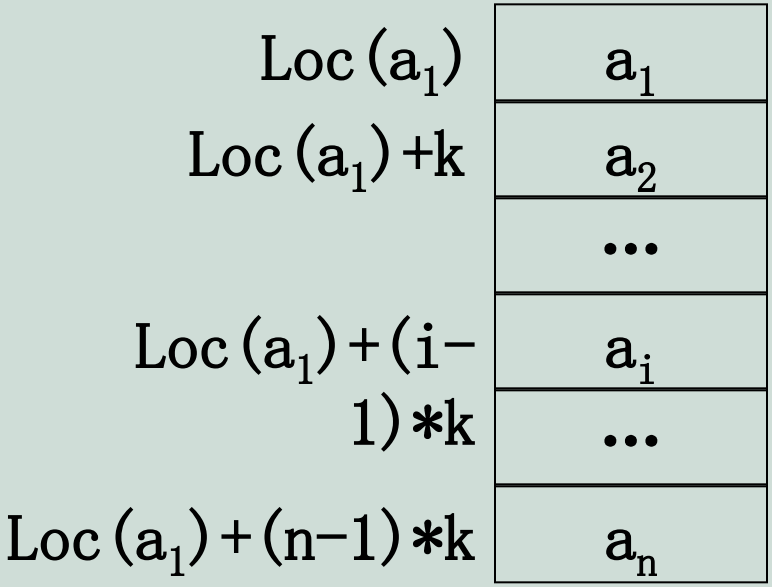
1. 线性表的顺序存储结构

把数据元素按照逻辑顺序依次存放到一组连续的存储单元中。

设线性表的基地址为：$LOC(a_1)$,a_i的存储地址为：

$$LOC(a_i) = LOC(a_1) + (i-1)*k \qquad 1<=i<=n$$

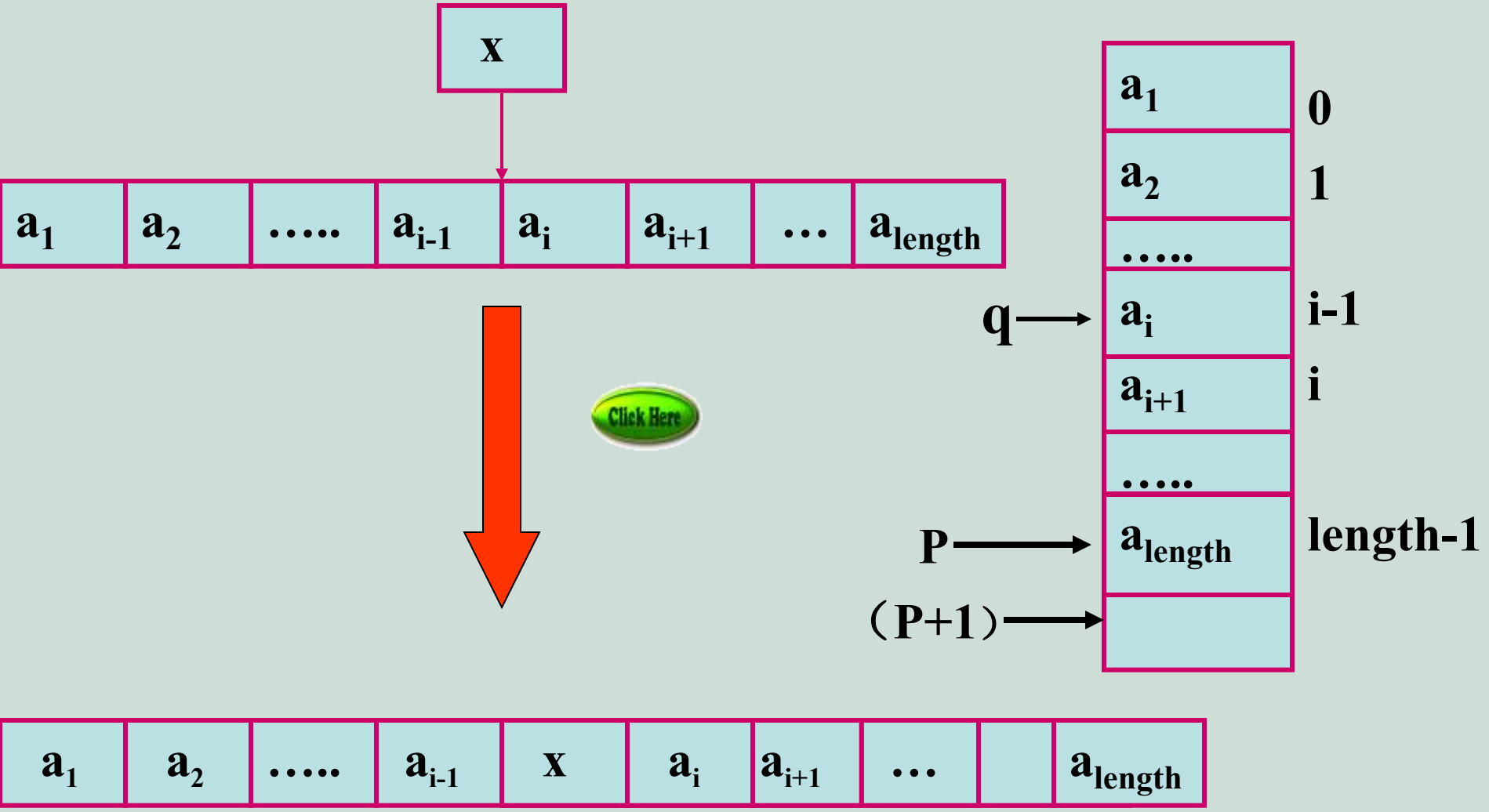




线性表的顺序存储结构——可用C语言中的一维数组来描述.

```
#define M  100  // 定义M为常数100，M的值作为数组的最大容量
 int V[M]；// V是数组的名字，假设数组中的数据元素是整型类型
int length；       //当前长度
```

1顺序表的插入运算


x
a1 a2 ….. ai-1 ai ai+1 … alength
a1
a2
…..
ai
ai+1
…..
alength
0
1
i-1
i
length-1
q
P
(P+1)
a1 a2 ….. ai-1 x ai ai+1 … alength

int insq(int i , int x, int V[ ],int M,int *p）

```
void main()
{
        int M=10,n=4;
        int result,k;
        static int V[10]={3,5,7,10};
        result=insq(2,4,V,M,&n);
        if(result==1) {printf("success\n");

        for(k=0;k<n;k++)printf("%d ",V[k]);}
        else printf("failure");
}
}
```

```
success
3 4 5 7 10 Press any key to continue_
```

Back

顺序表的删除运算

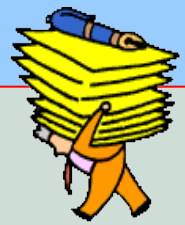
int delsq（int i ,int V[], int *p)

```
void main()
{
        int M=10,n=4;
        int result,k;
        static int V[10]={3,5,7,10};
        result=delsq(2,V,&n);
        if(result==1)
             {printf("success\n");
              for(k=0;k<n;k++) printf("%d ",V[k]);}
        else printf("failure");
}
```

```
This element is not in the list
failurePress any key to continue
```

```
success
3 7 10 Press any key to continue
```

}



- **特点：**
- 便于随机存取表中的任一个数据元素，做插入、删除时需移动大量元素。
- 静态分配空间，无法动态分配。
- 表中元素个数是动态变化时，按最大空间分配。
- 适用于需要频繁查找操作的线性表

线性表的链式存储结构

链表是一种常见的重要的数据结构。它是*动态*地进行存储分配的一种结构。

线性表的链式存储结构的含义是指逻辑上相邻的数据元素在内存中的物理存储空间不一定相邻。

每个数据元素对应内存一个存储空间，叫存储结点，简称结点。

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：https://d.book118.com/937124011053006145