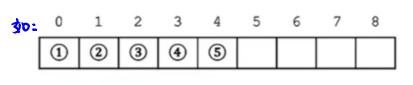
钜表与顺序表(基础线性表)

顺序表 → 省额外元素的教组,连续格



```
1、size = 9 → 大小
2、count = 5 → 為 Dota 个数
```

特点:读取Data速度快o(1)

结构操作:插入与删除

[]	;	右术多位元素	左移位补空缺
Size	:	۵	-
count	:	+1	-1

```
typedef struct vector
{
    int size,count;
    int *data;
}vector;
```

- 1> typedef 为叫vector的结构体 取一个叫vector的新类型名
- 2> 不知大小的教组→→*data 来方便扩客

①初始化代码—C:

```
//初始化一个空的顺序表
vector *init(int size)
{
    vector* v = (vector*)malloc(Size:sizeof(vector));
    v->size = size;
    v->count = 0;
    v->data = (int*)malloc(sizeof(int)*size);
    return v;
}
```

1>大小由初始化输入决定→参数
2> 先邢辟V的空间再开辟1师
序表中数组的空间

存储上限为Size

②销型代码—C:

```
//销毁一个顺序表
void destroy(vector *v)
{
    free(v->data);
    free(v);
}
```

17 逐层销毁的方法 27 从内到外开始销毁

malloc与free要-一对应!!!

③ 插入操作→C:

v->data[pos] = val:

printf(format: "You have inserted %d to %d\n", v->data[pos], pos);

```
int insert(vector *v,int pos,int val)
                                          口插入台法性验证必可少
   if(v->count == v->size)
                                          2> 逆序索引移位
      printf(format: "Your vector is full\n");
      return 0:
                                           3> count 故區 性值必须变
   for(int i = v->size; i > pos; i--)
      v->data[i] = v->data[i-1];
   v->data[pos] = val;
   v->count++:
   printf(format: "You have inserted %d to %d\n", v->data[pos], pos);
   return 1:
}
      册川紫操作 → C;
 int erase(vector *v,int pos)
                                                      1> 川灰序 漏沥前岭
                                                            逆序编历后线
    if(pos<0||pos>v->count)
        printf(format: "Your position is not right\n");
        return 0:
    for(int i = pos; i < v->size; i++)
        v->data[i] = v->data[i+1];
    v->count--;
    printf(format: "You have erased %d in %d\n", v->data[v->size-1], pos);
              入+扩雾 → C:
                        又文 realloc 第三个机制不发的有 bug
  int expend(vector *v)
     if(v = NULL)return 0;
                                          *P= (int *) realloc (v > data, size of (int ) + v > size + 2
     if(v->count == v->size)
                                           if (P == NULL) return 0;
       v->data = (int *)realloc(v->data, NewSize:sizeof(int)*v->size*2);
else v->data = P;
                                       1>利用 realloc扩容data 大小
     v->size = v->size * 2:
     return 1:
                                            同时同步修改Size大小
  //配套插入操作
  int insert_plus(vector *v,int pos,int val)
                                        2>在满的时候insert_plus
     if(v->count == v->size)
       expend(v);
                                             启动expend函数
     for(int i = v->size: i > pos: i--)
       v->data[i] = v->data[i-1];
```

realloc 机制: