|  |  |  |  |
| --- | --- | --- | --- |
| Void SelectionSort(int a[],int n) { | 0 | 0 | 0 |
| //对数组a[0:n-1]中的*n*个元素由小到大进行排序 | 0 | 0 | 0 |
| //当给定数组已经有序时，算法能够及时终止 | 0 | 0 | 0 |
| bool sorted = false; | 1 | 1 | 1 |
| for(int size = n; !sorted && (size>1); size--) { | 1 | Ω(1), Ο(*n*) | Ω(1), Ο(*n*) |
| int pos = 0; sorted = true; | 2 | 1 | 2 |
| //找最大元素，最大元素的下标为pos | 0 | 0 | 0 |
| for (int i=1; i<size; i++) | 1 | Ω(1), Ο(*n*) | Ω(n), Ο(*n*2) |
| if (a[pos]<=a[i]) | 1 | Ω(1), Ο(*n*) | Ω(n),Ο(*n*2) |
| pos = i; | 1 | Ω(1), Ο(*n*) | Ω(n),Ο(*n*2) |
| else sorted = false; //未按序排列 | 1 | Ω(1), Ο(*n*) | Ω(n),Ο(*n*2) |
| swap(a[pos], a[size-1]); //交换数据元素 | 1 | Ω(1), Ο(*n*) | Ω(1), Ο(*n*) |
| }//for size | 0 | 0 | 0 |
| }//SelectionSort | 0 | 0 | 0 |
| 总计 | Ω(*n*), Ο(*n*2) | | |

|  |  |  |  |
| --- | --- | --- | --- |
| 语句 | s/e | 频率 | 总步数 |
| void MergeLinkList(LinkList &La , LinkList Lb) { | 0 | 0 | 0 |
| //带头结点的单链表La、Lb是非递减的，表长分别为m，n。 | 0 | 0 | 0 |
| //表中元素为正整数，将Lb归并到La中且La为非递增的单链表。 | 0 | 0 | 0 |
| pa = La→next; pb = Lb→next; La→next = null; | 3 | 1 | 3 |
| LinkList s; | 1 | 1 | 1 |
| while(pa && pb) { //当pa、pb都不为空时，做归并处理 | 1 | Ω(min(m,n))，О(m+n) | Ω(min(m,n))，О(m+n) |
| if(pa→data<=pb→data) { | 1 | Ω(min(m,n))，О(m+n) | Ω(min(m,n))，О(m+n) |
| s=pa; | 1 | 0，О(m+n) | 0，О(m+n) |
| pa=pa→next;} | 1 | 0，О(m+n) | 0，О(m+n) |
| else { | 1 | Ω(min(m,n))，О(m+n) | Ω(min(m,n))，О(m+n) |
| s=pb; | 1 | 0，О(m+n) | 0，О(m+n) |
| pb=pb→next; } | 1 | 0，О(m+n) | 0，О(m+n) |
| s→next = La→next ; La→next = s; | 2 | Ω(min(m,n))，О(m+n) | Ω(min(m,n))，О(m+n) |
| }//while; | 0 | 0 | 0 |
| if(!pa) pa=pb; | 1 | 1 | 1 |
| while(pa) { //归并剩余的一个单链表。 | 1 | 1，О(max(m, n)) | 1，О(max(m, n)) |
| s=pa; pa=pa→next; | 2 | 1，О(max(m, n)) | 2，О(max(m, n)) |
| s→next = La→next ; La→next = s; | 2 | 2，О(max(m, n)) | 2，О(max(m, n)) |
| }//while; | 0 | 0 | 0 |
| free(Lb); | 1 | 1 | 1 |
| }// MergeLinkList | 0 | 0 | 0 |
| 总计 | О(m+n) | | |

|  |  |  |  |
| --- | --- | --- | --- |
| bool bubble\_sort(int a[],int n) {  //把a[0:n-1]中的最大元素冒泡至最后  bool swapped = false; //尚未发生交换  for(int i = 0; i<n-1; i++) {  if(a[i]>a[i+1]) {  Swap(a[i],a[i+1]);  swapped = true; //发生了交换  };  };  return swapped;  }  void BubbleSort(int a[], int n) {  //能及时终止的冒泡排序算法  for(int i = n; i>1 && bubble\_sort( a, i ); i--);  } | 0  0  1  1  1  3  1  0  0  1  0  0  0  1 | 0  0  θ (1)  θ (n)或n+1  θ (n)  Ω(0)，Ο(n)  Ω(0)，Ο(n)  0  0  θ (1)  0  0  0  Ω(n)，Ο(n2)  0 | 0  0  θ (1)  θ (n)  θ (n)  Ω(0)，Ο(n)  Ω(0)，Ο(n)  0  0  θ (1)  0  0  0  Ω(n)，Ο(n2)  0 |
| 总计 | Ω(n), Ο(n2) | | |