

基础作业

1 什么是算法的时间复杂度？什么是算法的空间复杂度

- 时间复杂度
 - 算法运行所需要的时间
- 空间复杂度
 - 算法运行所占用临时空间的大小

2 八大排序是哪八大排序？

- \选择\插入\希尔\快排\堆\归并\基数

3 实现冒泡排序

```
1  #define N 1024;
2  void bubbleSort(int *Arr)
3  {
4      for(int i=0;i<=N;i++)
5      {
6          for(int j=0;j<N-i;j++)
7          {
8              if(Arr[i]>Arr[i+1])
9              {
10                 swap(Arr[i],Arr[i+1]);
11             }
12         }
13     }
14 }
```

4 证明冒泡排序的正确性

```
1  #include<iostream>
2  #include<cstdio>
3  #include<cmath>
4  #include<cstdlib>
5  #include<ctime>
6  void bubbleSort(int* Arr);
7  void arrPrint(int* arr);
8  using namespace std;
9  int main()
10 {
11     int Arr[10];
12     srand(time_t(NULL)); //随机数种子
13     time_t start, end;
14     for (int i = 0; i < 10; i++)
15     {
16         Arr[i] = rand() % 100;
17     }
18     cout << "原数据为" << endl;
19     arrPrint(Arr);
```

```

20     cout << "-----" << endl;
21     bubbleSort(Arr);
22     cout << "排序后为" << endl;
23     cout << "-----" << endl;
24     arrPrint(Arr);
25 }
26
27 void bubbleSort(int* a)
28 {
29     for (int i = 0; i <= 10-1; i++)
30     {
31         for (int j = 0; j < 10 -1-i; j++)
32         {
33             if (a[j] > a[j + 1])
34             {
35                 swap(a[j], a[j + 1]);
36             }
37         }
38     }
39 }
40 void arrPrint(int* arr)
41 {
42     for (int i = 0; i < 10-1; i++)
43     {
44         printf("%3d", arr[i]);
45     }
46     printf("\n");
47 }

```



```

MICROSOFT VISUAL STUDIO 调试控制台
原数据为
38 19 38 37 55 97 65 85 50
-----
排序后为
-----
12 19 37 38 38 50 55 65 85
C:\Users\CK\Desktop\code\code=learning\day12\sort\Debug\冒泡排序.exe (进程 5208) 已挂起

```

5 画图说明插入排序、选择排序和希尔排序的过程

- 插入排序
- 选择排序
- 希尔排序

6 重排一个数组，以a[n-1]为标准，所有小于标准的数字放在标准左边，所有大于标准的数字放在标准的右边。

```

1  #include<iostream>
2  #include<cstdio>
3  #include<cmath>
4  #include<cstdlib>
5  #include<ctime>
6  void bubbleSort(int* Arr);
7  void arrPrint(int* arr);
8  using namespace std;
9  int main()
10 {

```

```

11     int Arr[10];
12     srand(time_t(NULL)); //随机数种子
13     time_t start, end;
14     for (int i = 0; i < 10; i++)
15     {
16         Arr[i] = rand() % 100;
17     }
18     int pivotVal=Arr[9];
19     cout << "原数据为" << endl;
20     arrPrint(Arr);
21     cout << "-----" << endl;
22     partition(Arr,pivotVal);
23     cout << "排序后为" << endl;
24     cout << "-----" << endl;
25     arrPrint(Arr);
26 }
27 void partition (int *N,int pivotVal) //设这里数组长度为10
28 {
29     int begin=0; //头指针
30     int end=9; //尾指针
31     int pivot=pivotVal;
32     while(begin<end)
33     {
34         if(N[begin]<=pivot)
35         { //如果后面发现大于pivot的值则向前移动指针
36             begin++;
37         }
38         else if //如果比pivot大, 则直接交换此时end和begin的值
39         {
40             N[begin]=N[end];
41         }
42         else if(N[end]>=pivot)
43         { //如果后面发现大于pivot的值则向前移动指针
44             --end;
45         }
46         else //如果比pivot大, 则直接交换此时end和begin的值
47         {
48             N[end]=N[begin];
49         }
50     }
51 }

```

•

7 说明冒泡、选择和插入排序的最坏和平均时间复杂度

排序方法	最坏时间复杂度	平均时间复杂度
冒泡	$O(n^2)$	$O(n^2)$
选择	$O(n^2)$	$O(n^2)$
插入	$O(n^2)$	$O(n^2)$

8 说明快速排序最坏和平均时间复杂度

- 快速排序时间复杂度最坏为 $O(n^2)$
- 快速排序时间复杂度平均为 $O(n\log n)$