

本节内容

希尔排序

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先追求表中元素部分有序，再逐渐逼近全局有序

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

第一趟： $d_1=n/2=4$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

0	1	2	3	4	5	6	7	8

第一趟： $d_1=n/2=4$

49					76			
	38					13		
		65					27	
				97				49

子表1

子表2

子表3

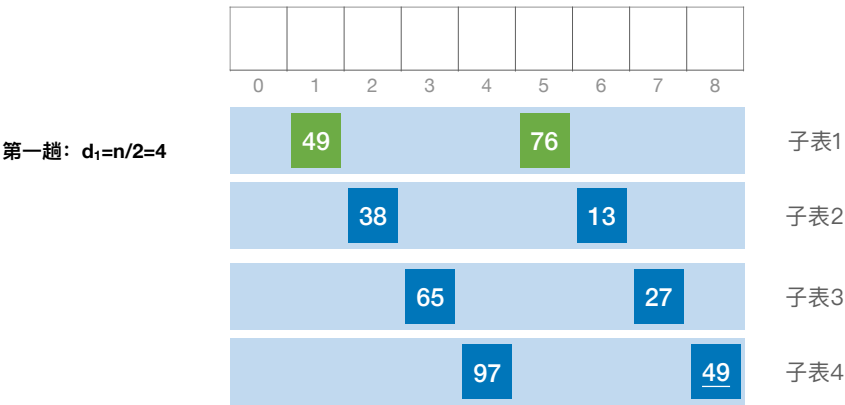
子表4

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

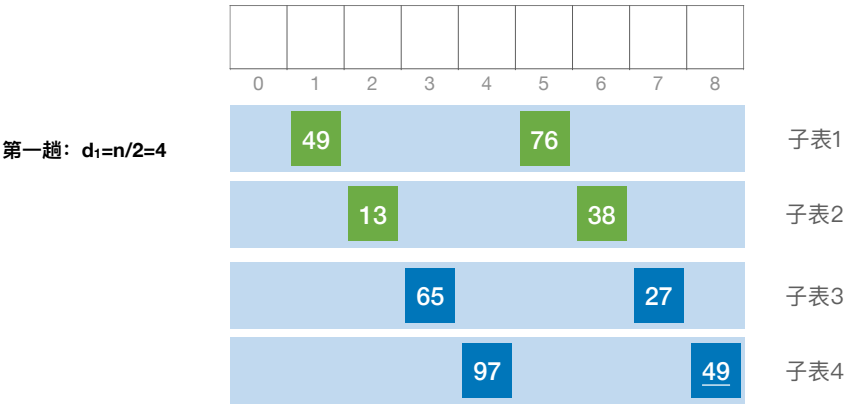


王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。



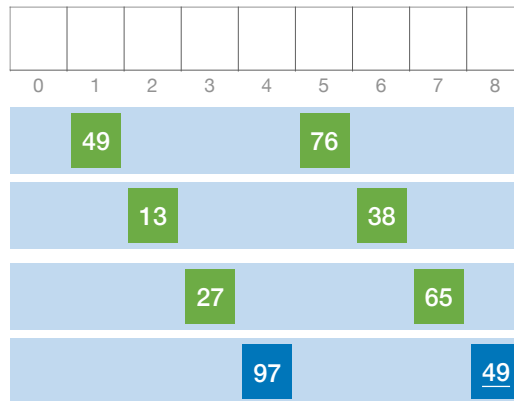
王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

第一趟： $d_1=n/2=4$



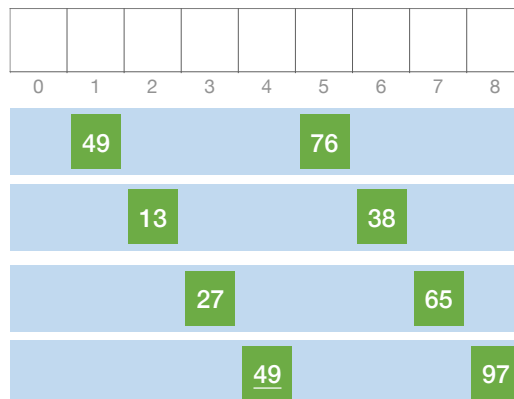
王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

第一趟： $d_1=n/2=4$



王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

	49	13	27	<u>49</u>	76	38	65	97
0	1	2	3	4	5	6	7	8

第二趟： $d_2 = d_1 / 2 = 2$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

0	1	2	3	4	5	6	7	8

第二趟： $d_2 = d_1 / 2 = 2$

49	27	76	65
13	<u>49</u>	38	97

子表1

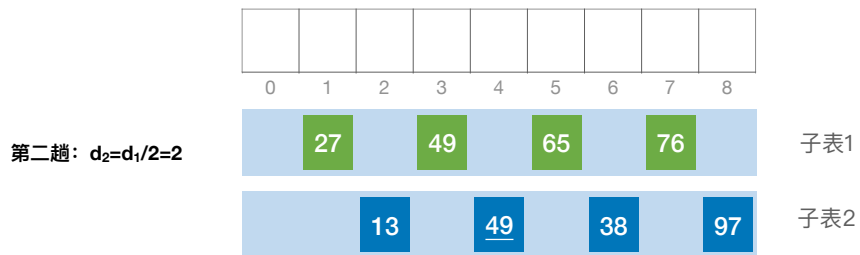
子表2

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

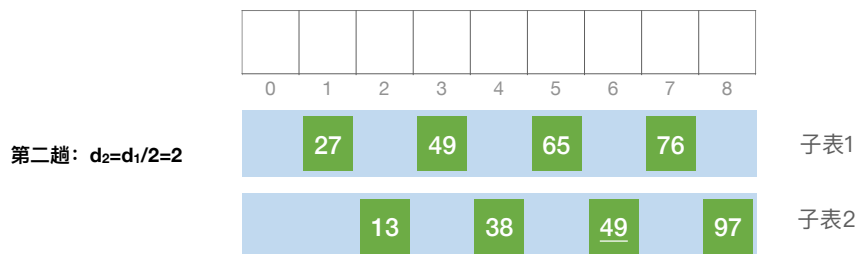


王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。



王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

	27	13	49	38	65	<u>49</u>	76	97
0	1	2	3	4	5	6	7	8

第三趟： $d_3=d_2/2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

0	1	2	3	4	5	6	7	8

第三趟： $d_3=d_2/2=1$

	27	13	49	38	65	<u>49</u>	76	97
--	----	----	----	----	----	-----------	----	----

子表1

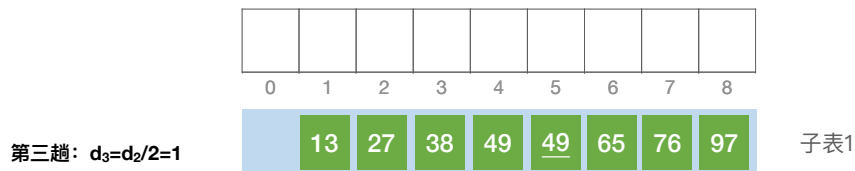
整个表已呈现出“基本有序”，对整体再进行一次“直接插入排序”

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。



整个表已呈现出“基本有序”，对整体再一次“直接插入排序”

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。



王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)

希尔排序：先将待排序表分割成若干形如 $L[i, i+d, i+2d, \dots, i+kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

希尔本人建议：每次
将增量缩小一半

第一趟： $d_1=n/2=4$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

	49	13	27	49	76	38	65	97
0	1	2	3	4	5	6	7	8

第二趟： $d_2=d_1/2=2$

	27	13	49	38	65	49	76	97
0	1	2	3	4	5	6	7	8

第三趟： $d_3=d_2/2=1$

	13	27	38	49	49	65	76	97
0	1	2	3	4	5	6	7	8

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)

希尔排序：先将待排序表分割成若干形如 $L[i, i+d, i+2d, \dots, i+kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

第一趟： $d_1=3$

	27	38	65	49	76	13	97	49
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

	27	38	65	49	76	13	97	49
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

	27	38	65	49	49	13	97	76
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

	27	38	65	49	49	13	97	76
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

	27	38	13	49	49	65	97	76
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

	27	38	13	49	49	65	97	76
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

王道考研/CSKAOYAN.COM

希尔排序 (Shell Sort)



希尔排序：先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表，对各个子表分别进行直接插入排序。缩小增量 d ，重复上述过程，直到 $d=1$ 为止。

考试中可能遇到各种增量

第一趟： $d_1=3$

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

	27	38	13	49	49	65	97	76
0	1	2	3	4	5	6	7	8

第二趟： $d_2=1$

	13	27	38	49	49	65	76	97
0	1	2	3	4	5	6	7	8

王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

	49	38	65	97	76	13	27	49
0	1	2	3	4	5	6	7	8

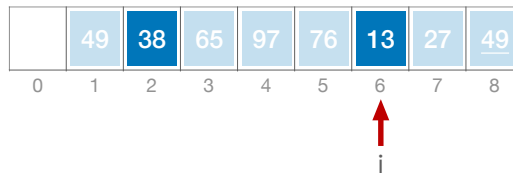
↑
i

王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

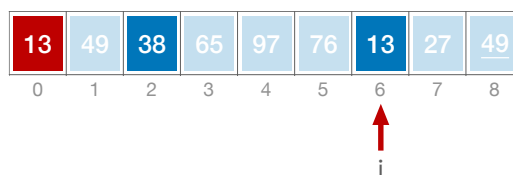


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

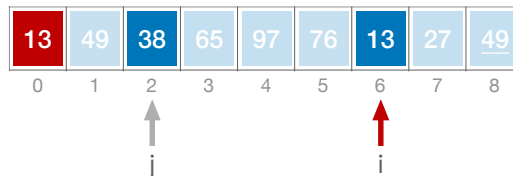


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }
}
```

第一趟：d=n/2=4

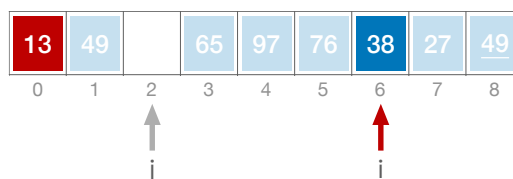


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }
}
```

第一趟：d=n/2=4

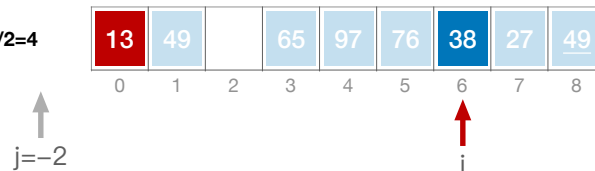


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟: $d=n/2=4$

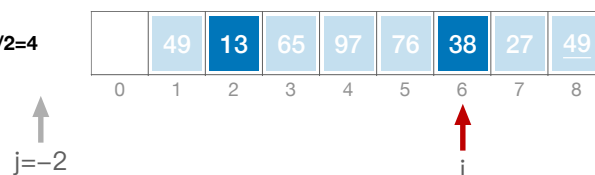


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟: $d=n/2=4$

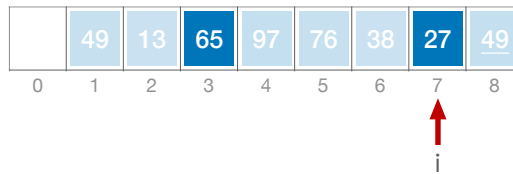


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

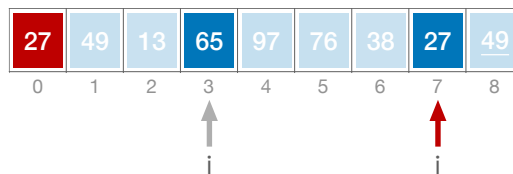


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

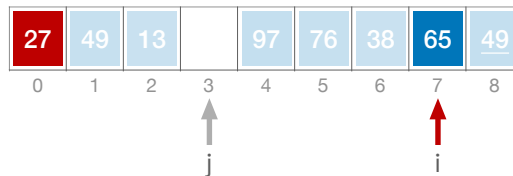


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟: $d=n/2=4$

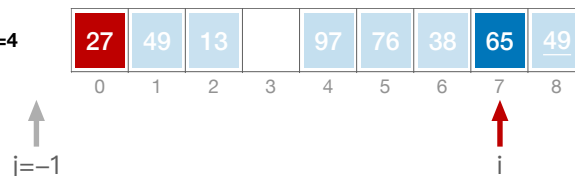


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟: $d=n/2=4$



王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

	49	13	27	97	76	38	65	49
0	1	2	3	4	5	6	7	8

↑
i

王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

	49	13	27	97	76	38	65	49
0	1	2	3	4	5	6	7	8

↑
i

王道考研/CSKAOYAN.COM

}

Diagram illustrating the array structure and indices for the selection sort algorithm. The array contains the values [49, 49, 13, 27, 97, 76, 38, 65, 49]. The element 49 at index 0 is highlighted in red. The element 97 at index 4 is highlighted in blue. The element 49 at index 8 is highlighted in blue. A grey arrow labeled 'j' points to the element 97 at index 4. A red arrow labeled 'i' points to the element 49 at index 8.

王道考研/CSKAOYAN.COM

}

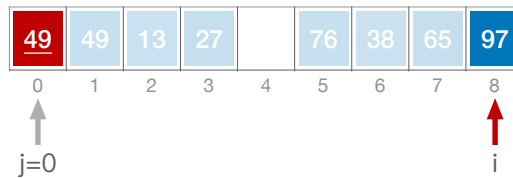
Diagram illustrating an array structure with 9 elements. The array contains the values: 49, 49, 13, 27, (blank), 76, 38, 65, 97. The first element (index 0) is highlighted in red. The element at index 8 (value 97) is highlighted in blue. An arrow labeled i points to the blank element at index 4. Another arrow labeled i points to the element at index 8.

王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟: $d=n/2=4$

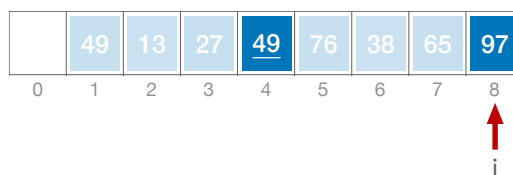


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟: $d=n/2=4$



王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第一趟：d=n/2=4

	49	13	27	49	76	38	65	97
0	1	2	3	4	5	6	7	8



i

王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

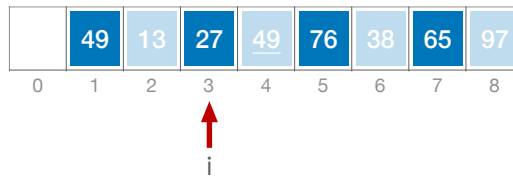
	49	13	27	49	76	38	65	97
0	1	2	3	4	5	6	7	8

王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

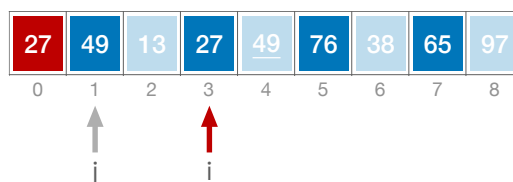


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

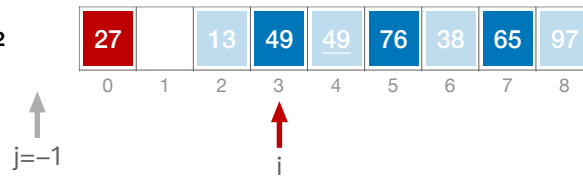


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

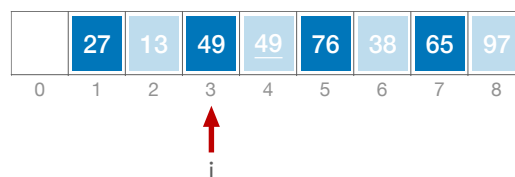


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

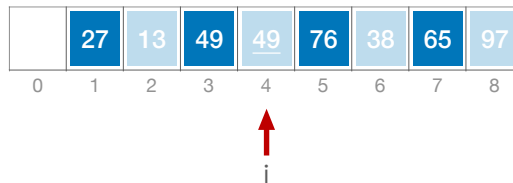


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

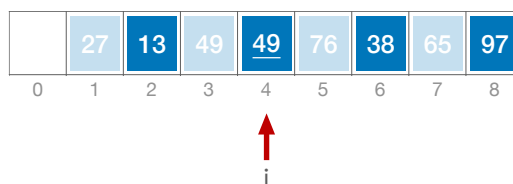


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

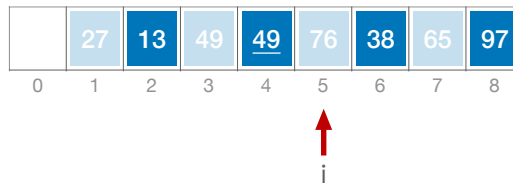


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

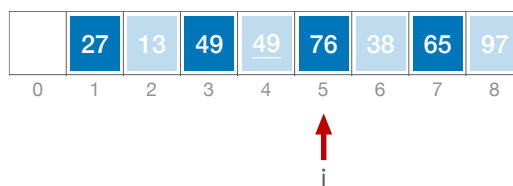


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

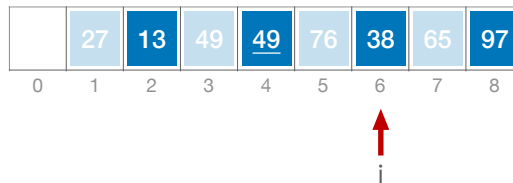


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

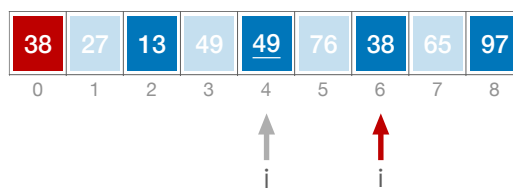


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

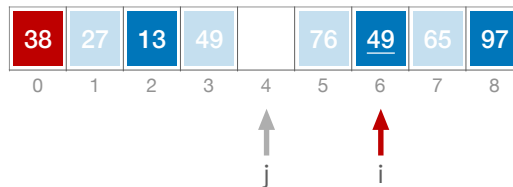


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

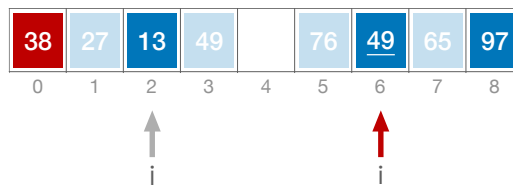


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

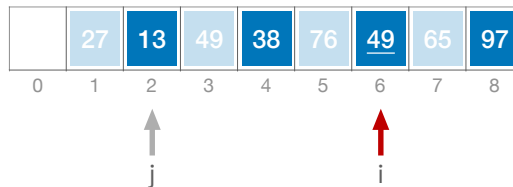


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

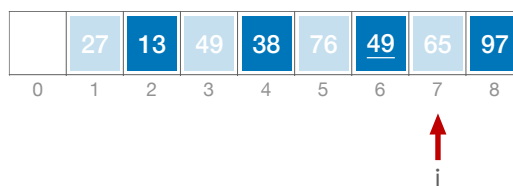


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

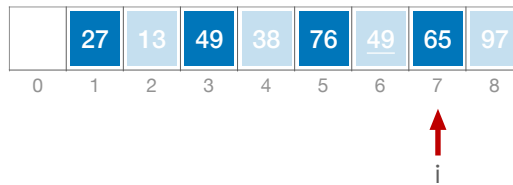


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

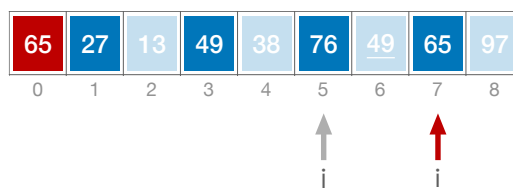


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

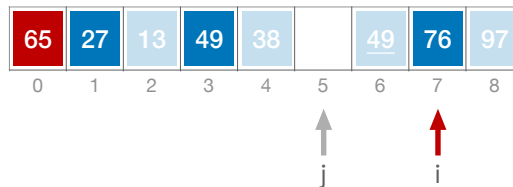


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

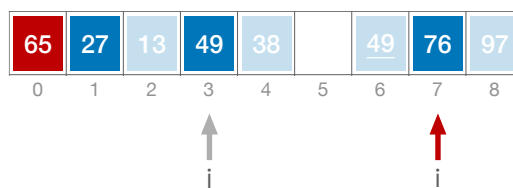


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

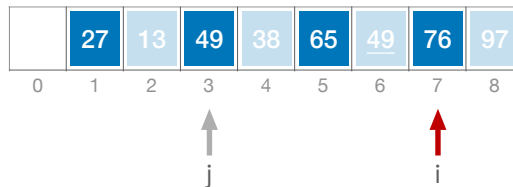


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

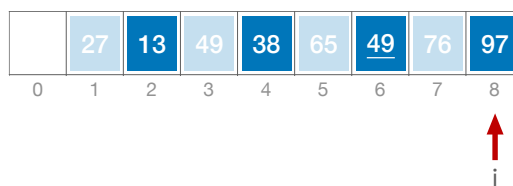


王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2



王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第二趟：d=2

	27	13	49	38	65	49	76	97
0	1	2	3	4	5	6	7	8



王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2)    //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){    //需将A[i]插入有序增量子表
                A[0]=A[i];    //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j];    //记录后移，查找插入的位置
                A[j+d]=A[0];    //插入
            }//if
}
```

第三趟：d=1

	27	13	49	38	65	49	76	97
0	1	2	3	4	5	6	7	8



王道考研/CSKAOYAN.COM

算法实现

```
//希尔排序
void ShellSort(int A[],int n){
    int d, i, j;
    //A[0]只是暂存单元，不是哨兵，当j<=0时，插入位置已到
    for(d= n/2; d>=1; d=d/2) //步长变化
        for(i=d+1; i<=n; ++i)
            if(A[i]<A[i-d]){ //需将A[i]插入有序增量子表
                A[0]=A[i]; //暂存在A[0]
                for(j= i-d; j>0 && A[0]<A[j]; j-=d)
                    A[j+d]=A[j]; //记录后移，查找插入的位置
                A[j+d]=A[0]; //插入
            }
}
```

第三趟: d=1

	13	27	38	49	<u>49</u>	65	76	97
0	1	2	3	4	5	6	7	8

王道考研/CSKAOYAN.COM

算法性能分析

空间复杂度: $O(1)$

	49	38	65	97	76	13	27	<u>49</u>
0	1	2	3	4	5	6	7	8

第一趟: $d_1=n/2=4$

49	13	27	<u>49</u>	76	38	65	97
----	----	----	-----------	----	----	----	----

第一趟: $d_1=3$

27	38	13	49	<u>49</u>	65	97	76
----	----	----	----	-----------	----	----	----

第二趟: $d_2=d_1/2=2$

27	13	49	38	65	<u>49</u>	76	97
----	----	----	----	----	-----------	----	----

第二趟: $d_2=1$

13	27	38	49	<u>49</u>	65	76	97
----	----	----	----	-----------	----	----	----

第三趟: $d_3=d_2/2=1$

13	27	38	49	<u>49</u>	65	76	97
----	----	----	----	-----------	----	----	----



时间复杂度: 和增量序列 d_1, d_2, d_3, \dots 的选择有关, 目前无法用数学手段证明确切的时间复杂度
最坏时间复杂度为 $O(n^2)$, 当 n 在某个范围内时, 可达 $O(n^{1.3})$

王道考研/CSKAOYAN.COM

算法性能分析

原始序列:

65 49 49

第一趟: $d=2$

49 49 65

第二趟: $d=1$

49 49 65

稳定性: 不稳定!

适用性: 仅适用于顺序表, 不适用于链表

王道考研/CSKAOYAN.COM

知识回顾与重要考点

先将待排序表分割成若干形如 $L[i, i + d, i + 2d, \dots, i + kd]$ 的“特殊”子表, 对各个子表分别进行直接插入排序。缩小增量 d , 重复上述过程, 直到 $d=1$ 为止。

希尔排序

性能

空间复杂度 $O(1)$

时间复杂度 未知, 但优于直接插入排序

稳定性 不稳定

适用性 仅可用于顺序表

高频题型: 给出增量序列, 分析每一趟排序后的状态

王道考研/CSKAOYAN.COM