线性查找算法实现报告

王资 18214668 计算机技术

1.基本思想

基本算法实现如课件所示，即 1）设置递归结束条件为 n < 75，当满足该条件时，直接排序并返回第 k 个元素，2）递归地寻找整串列表的中位数，3）找到中位数 x 后使用归并，即小的放在 x 之前，大的在其之后，4）标识上述中位数的位置 j，若 k<=j 则递归地在前 j 之前的部分查找，即(p, j]，否则递归地在 j 之后的部分查找，即(j, r]，此时 k <- k - j, 因为前面 j 个元素都比 x 小，所以可以跳过这 j 个元素，而 k 对于 j 之后的元素来说，排第 k-j.

2.代码中的数据结构

我实现该算法的代码中使用了一个集合 set，用来在随机生成数组时，避免产生相同的元素导致算法失效；此外只使用了一个数组（或者说一个指针，并申请了相应大小的内存）作为实现该算法的数据结构。

3.程序流程

1.产生一串长度为 length 的不重复数组

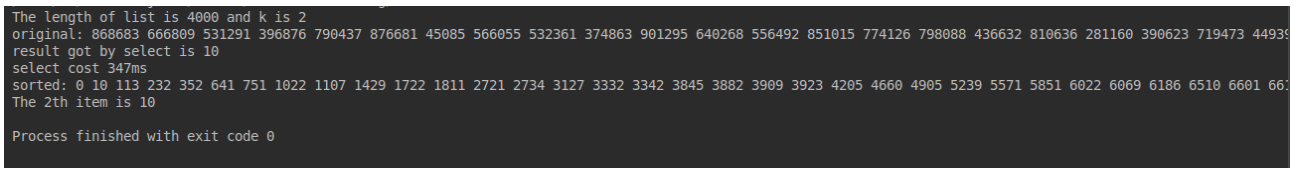
2.使用线性查找算法，找到第 k 小的元素

3.计算上述算法运行的时间，验证该算法的时间复杂度是否为线性

4.对数组进行排序，验证查找出来的元素是正确的

4.调试过程出现的问题和解决方法

1.数组中多了一个元素，通常是 0，这就导致了最终只找到了第 k-1 个元素，如下图所示。



出现问题的原因是查找函数的调用 select(a, p, r, k)中 r 是数组 a 的最后一个元素的位置，而我错误地输入了 a 的长度，因此相当于 a 的长度多了 1，多出了一个元素，而这个元素通常为 0（视计算机系统和编译器的情况而定），当然，这是超出了输入数组的内存分配长度的，在如 Java 等语言会报溢出错误，而 C/C++则可以直接读取程序未分配的内存。

解决方法是在主函数调用的时候令 r=len(a)-1 即可解决这个问题。

5.运行结果

1.n=30 k=10

```
The length of list is 30 and k is 10
original: 79 87 120 62 156 109 2 181 85 158 105 26 145 12 103 27 179 69 74 8 118 32 155 7 53 161 144 194 78 200
result got by select is 69
select cost 4us
sorted: 2 7 8 12 26 27 32 53 62 69 74 78 79 85 87 103 105 109 118 120 144 145 155 156 158 161 179 181 194 200
The 10th item is 69
```

2.n=1000 k=20

```
The length of list is 1000 and k is 10
original: 945 861 563 864 382 222 407 1047 1025 514 1107 1168 892 60 661 677 478 934 81 1067 400 405 890 979 626 658 149 822 866 978 1178 130 990 489 108 266 7
result got by select is 11
select cost 101us
sorted: 1 2 3 4 5 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 26 28 29 30 31 32 33 34 35 36 38 39 40 42 43 44 45 46 47 48 50 51 52 53 55 56 57 58 59 60
The 10th item is 11
```

3.n=2000 k=1024

```
The length of list is 2000 and k is 1024
original: 1840 909 1639 2078 1506 391 605 30 748 491 45 689 1796 1954 64 1397 1716 1958 937 509 964 175 1529 695 74 1288 669 423 713 672 541 1869 2080 1147 1202
result got by select is 1071
select cost 177us
sorted: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54
The 1024th item is 1071
```

时间复杂度验证

令 n 分别为 1000 2000 4000 8000，每个 n 值运行 3 次，取时间（单位为 us）的平均值，验证时间复杂度为线性。时间只计算线性查找部分的代码，即随机生成列表部分和排序校验部分不会算入其中。

n	t1	t2	t3	平均时间	时间差
1000	99	107	95	100	-
2000	180	176	186	181	81
4000	309	369	325	334	153
8000	650	593	578	607	273

通过简单的验证可以看出，输入的规模扩大一倍后，算法实现运行需要的时间的相差值的增长也是接近 1 倍，实验中比 1 倍小一点，所以可以从该验证中证实该算法的实现的时间复杂度是线性的。

6.总结

本实验实现了线性时间查找无重复元素列表的算法，通过输入、输出校验得知了实现成功，通过对比各个输入规模下算法耗费的时间，得出算法的实现的时间复杂度确实是线性的，综上，说明实现是成功的。此外，该算法的奇思妙想也值得让人钦佩，在线性的时间内找出不需要查找的部分，最终得出的递归表达式的复杂度就能下降，是一个很好的分治思想，也是学习分治策略的一个很好的例子。

7.存在问题和改进设想

该算法的实现存在大量递归，在实际程序运行中会造成效率下降，而很容易发现，查找中位数的部分并不需要递归操作，而是能够很简单地替换成用一个循环实现，这样就能在实际程序中提高运行的效率，同时，使用循环替代函数递归更容易在开发过程中对代码进行调试，在遇到莫名错误的时候能更清晰地追踪每个变量的状态。