第二次上机作业

- 做计时实验，验证List的按索引取值确实是O(1)
- 做计时实验，验证Dict的get item和set item都是O(1)的
- 做计时实验，比较List和Dict的del操作符性能
  del lst[i]/del dic[key]
- 给定一个随机顺序的数列表，写一个复杂度为O(nlogn)的求第k小的数的算法
- 请改进上述的算法，使之复杂度降低为O(n)
- 4月7日23:59前提交

No.1

- **问题规模**

  列表大小

- **典型操作**

  首个

  中间

  尾部

- **得到结论**

  O(1)，与规模大小无关

```
from timeit import Timer
t= Timer("lst[n]", "from __main__ import lst, n")
print "scale, times, first, middle, last"
times= 1000000
for i in range(100000, 1000001, 100000):
    lst= list(range(i))
    n= 0
    t1= t.timeit(number= times)
    n= i/2
    t2= t.timeit(number= times)
    n= i- 1
    t3= t.timeit(number= times)
    print "%d, %d, %8f, %8f, %8f" % (i, times, t1, t2, t3)
```

```
>>>
scale, times, first, middle, last
100000, 1000000, 0.119498, 0.069085, 0.068040
200000, 1000000, 0.067517, 0.067820, 0.066463
300000, 1000000, 0.068341, 0.066858, 0.066857
400000, 1000000, 0.070941, 0.066951, 0.051696
500000, 1000000, 0.030407, 0.029029, 0.028964
600000, 1000000, 0.029381, 0.029446, 0.029236
700000, 1000000, 0.030384, 0.029986, 0.029105
800000, 1000000, 0.030072, 0.028931, 0.029167
900000, 1000000, 0.029303, 0.030195, 0.029228
1000000, 1000000, 0.029758, 0.029222, 0.029304
```

No.2

- **问题规模**

  字典大小

- **典型操作**

  get item

  set item

- **得到结论**

  O(1)，与规模大小无关

```
from timeit import Timer
t= Timer("dic[n]", "from __main__ import dic, n")
u= Timer("dic[n]= n", "from __main__ import dic, n")
print "scale, times, get itme, set item"
times= 1000000
for i in range(100000, 1000001, 100000):
    dic= {j:None for j in range(i)}
    n= i/2
    t1= t.timeit(number= times)
    u1= u.timeit(number= times)
    print "%d, %d, %8f, %8f" % (i, times, t1, u1)
```

```
scale, times, get itme, set item
100000, 1000000, 0.106218, 0.078400
200000, 1000000, 0.056124, 0.078067
300000, 1000000, 0.053278, 0.067994
400000, 1000000, 0.054034, 0.065150
500000, 1000000, 0.052998, 0.065650
600000, 1000000, 0.054682, 0.065785
700000, 1000000, 0.055126, 0.065764
800000, 1000000, 0.054864, 0.064838
900000, 1000000, 0.052875, 0.065154
1000000, 1000000, 0.054184, 0.066413
```

No.3

- **问题规模**

  列表，字典大小

- **典型操作**

  del lst[n]

  del dic[n]

- **得到结论**

  列表的删除时间

  代价远超字典

  字典删除为O(1)，与规模无关

  列表删除是O(n)

```
from timeit import Timer
t= Timer("del lst[n]", "from __main__ import lst, n")
u= Timer("del dic[n];dic[n]=None", \
         "from __main__ import dic, n")
print "scale, times, del lst[n], del dic[n]"
times= 10000
for i in range(100000, 1000001, 100000):
    lst= list(range(i))
    dic= {j:None for j in range(i)}
    n= i/2
    t1= t.timeit(number= times)
    u1= u.timeit(number= times)
    print "%d, %d, %8f, %8f" % (i, times, t1, u1)
```

```
scale, times, del lst[n], del dic[n]
100000, 10000, 0.149438, 0.001372
200000, 10000, 0.185508, 0.000810
300000, 10000, 0.248758, 0.000812
400000, 10000, 0.336087, 0.001070
500000, 10000, 0.431946, 0.000854
600000, 10000, 0.506342, 0.000793
700000, 10000, 0.593136, 0.000901
800000, 10000, 0.679218, 0.000807
900000, 10000, 0.764684, 0.000813
1000000, 10000, 0.855565, 0.000807
```