In [1]:
```
import random
import time
import matplotlib.pyplot as plt
```

In [2]:
```
import sys
sys.setrecursionlimit(100000)
sys.getrecursionlimit()
```

Out[2]:
```
100000
```

In [3]:
```
def Hoare_partition(arr, low, high):

    pivot = arr[int((low + high)/2)]
    #print("pivot" , pivot)
    i = low - 1
    j = high + 1
    #print(i,j)

    while (True):
        i += 1
        while (arr[i] < pivot):
            i += 1
        j -= 1
        while (arr[j] > pivot):
            j -= 1
        if (i >= j):
            #print("new pivot", j , arr[j])
            return j

        #print("i,j:", i, j)
        arr[i], arr[j] = arr[j], arr[i]
    #print(arr)
```

In [4]:
```
def Quick_sort(A , start, end):

    if start < end:
        pivot =  Hoare_partition(A, start, end)
        Quick_sort(A, start, pivot)
        Quick_sort(A, pivot+1 , end)
```

A = [4, 1 , 2, 0 , 1] n = len(A) Quick_sort(A, 0 , n-1) print(A)

In [5]:
```
Best_case = [0, 1, 3, 2, 4, 5, 6, 7, 8, 10, 9]

length = len(Best_case)
n = length - 1

st = time.time()
Quick_sort(Best_case, 0 , n)
et = time.time()
elapsed_time = et - st
print('Best Case Execution time:', elapsed_time, 'seconds')
```

```
Best Case Execution time: 4.029273986816406e-05 seconds
```

In [6]:
```
worst_case = [10, 9, 8 , 7, 0, 6, 5, 4,3, 2, 1]
length = len(worst_case)
n = length - 1

st = time.time()
Quick_sort(worst_case, 0 , n)
et = time.time()
elapsed_time = et - st
print('Worst Case Execution time:', elapsed_time, 'seconds')
```

```
Worst Case Execution time: 4.482269287109375e-05 seconds
```

In [7]:
```
avg_case = [1, 2 , 4, 6, 10, 9 ,  8 , 7]
length = len(avg_case)
n = length - 1

st = time.time()
Quick_sort(avg_case, 0 , n)
et = time.time()
elapsed_time = et - st
print('Avg Case Execution time:', elapsed_time, 'seconds')
```

```
Avg Case Execution time: 3.886222839355469e-05 seconds
```

In [8]:
```
input_list = [10, 1000, 5000, 10000]
time_taken = []
for i in input_list:
    random.seed(10)
    randomlist = random.sample(range(0 , i), i)
    length = len(randomlist)
    n = length - 1
    st = time.time()
    Quick_sort(randomlist, 0 , n)
    et = time.time()
    elapsed_time = et - st
    time_taken.append(elapsed_time)
    print('When input is', i ,': Execution time:', elapsed_time, 'seconds')
```

```
When input is 10 : Execution time: 1.0013580322265625e-05 seconds
When input is 1000 : Execution time: 0.0015249252319335938 seconds
When input is 5000 : Execution time: 0.008076190948486328 seconds
When input is 10000 : Execution time: 0.018208026885986328 seconds
```

plt.figure(figsize=(10,10)) plt.plot(input_list,time_taken, 'ro') plt.xticks(input_list) plt.yticks(time_taken) plt.xlabel("input size") plt.ylabel("time taken for sorting") plt.savefig('quick_sort_hoare.pdf') plt.savefig('quick_sort_hoare.png') plt.show() plt.close()