106119029

```cpp
#include <algorithm>
#include <chrono>
#include <cstdlib>
#include <ctime>
#include <iostream>
#include <numeric>
#include <string>
#include <unordered_map>
#include <vector>

int main() {
  srand(time(NULL));
  // will be {10,100,200,400,800,1600,3200,6400,12800,25600}
  std::vector<int> size_array(10, 0);
  size_array[0] = 10;
  size_array[1] = 100;
  for (int i = 2; i < 10; i++)
    size_array[i] = 2 * size_array[i - 1];
  size_array[0] = 10;

  std::vector<int> vec;
  for (auto &&size : size_array) {
    vec.assign(size, 0);
    std::generate(vec.begin(), vec.end(), [&]() { return rand() % size; });

    int val_to_search = vec[rand() % size];

    auto start = std::chrono::high_resolution_clock::now();
    for (int i = 0; i < size; i++) {
      if (val_to_search == vec[i]) {
        break;
      }
    }
    auto end = std::chrono::high_resolution_clock::now();
    std::chrono::duration<double> elapsed =
        std::chrono::duration_cast<std::chrono::duration<double>>(end - start);
    printf("FoundCase:%d:%.10lf\n", size, elapsed.count());

    val_to_search = size + 100;
    start = std::chrono::high_resolution_clock::now();
    for (int i = 0; i < size; i++) {
      if (val_to_search == vec[i]) {
        break;
      }
    }
    end = std::chrono::high_resolution_clock::now();
    elapsed =
        std::chrono::duration_cast<std::chrono::duration<double>>(end - start);
    printf("NotfoundCase:%d:%.10lf\n", size, elapsed.count());
  }
}
```

- I have put the output of the code into a file which has the following content. This is the output of the above code. I then put the output in a file called `out` and plotted it using matplotlib python library

```
FoundCase:10:0.0000004450
NotfoundCase:10:0.0000005160
FoundCase:100:0.0000003360
NotfoundCase:100:0.0000014240
FoundCase:200:0.0000014470
NotfoundCase:200:0.0000026980
FoundCase:400:0.0000018170
NotfoundCase:400:0.0000052980
FoundCase:800:0.0000037200
```

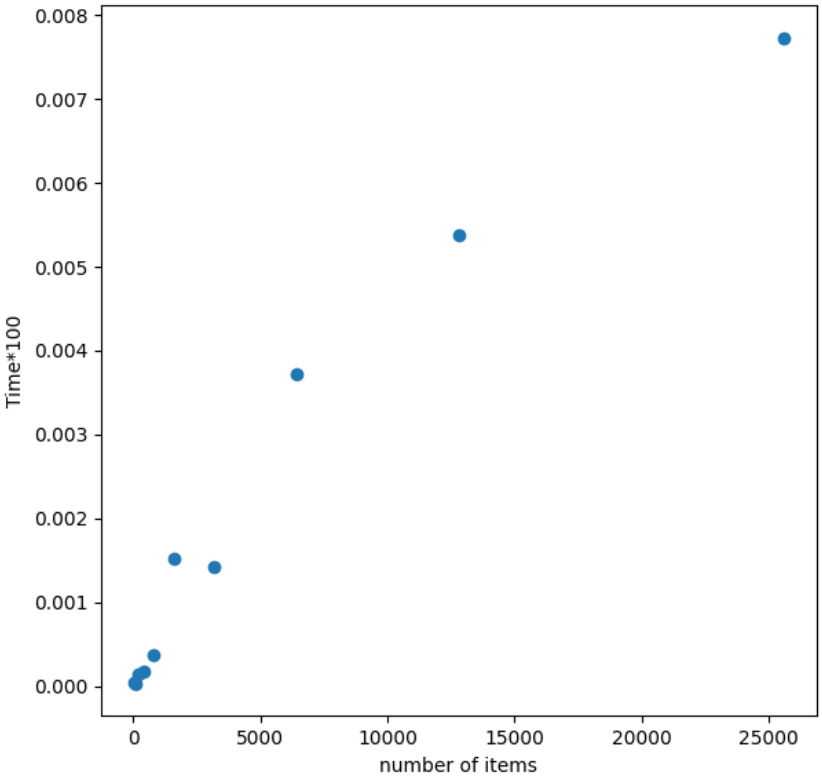
```
NotfoundCase:800:0.0000104970
FoundCase:1600:0.0000152090
NotfoundCase:1600:0.0000207800
FoundCase:3200:0.0000142310
NotfoundCase:3200:0.0000414270
FoundCase:6400:0.0000371370
NotfoundCase:6400:0.0000826650
FoundCase:12800:0.0000538130
NotfoundCase:12800:0.0001651470
FoundCase:25600:0.0000772520
NotfoundCase:25600:0.0003302860
```

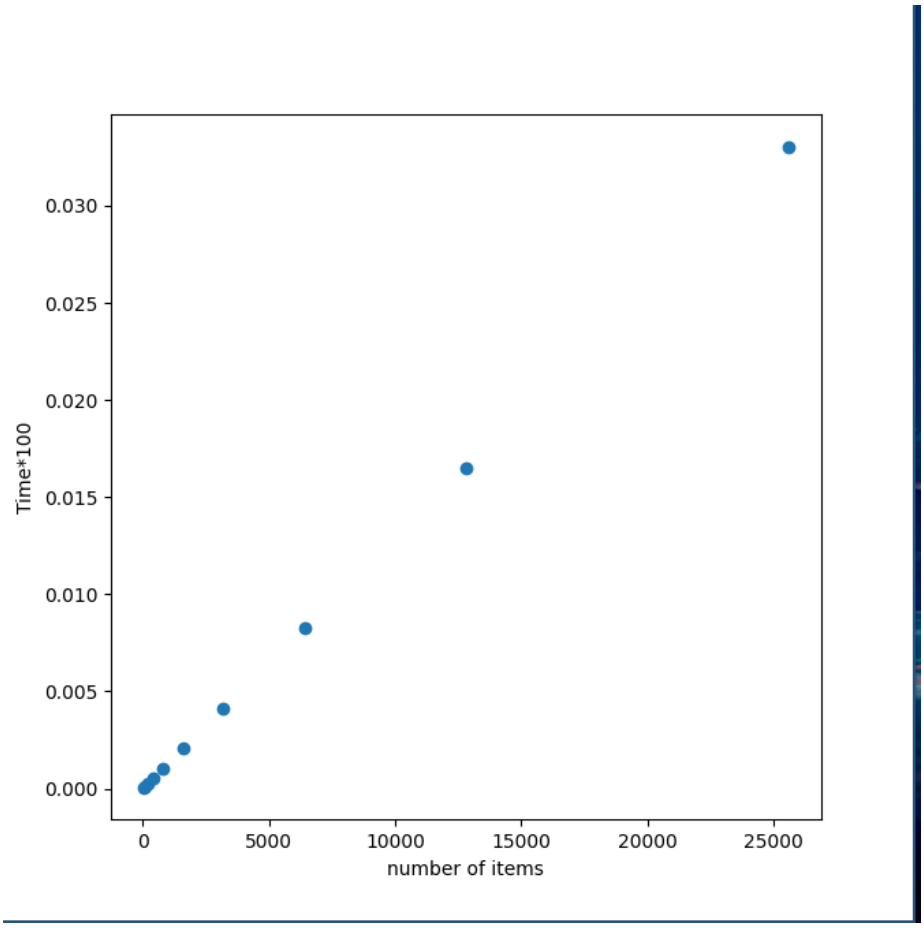
- Now plotting the data after some processing with the following code:


```
import matplotlib.pyplot as plt

fp = open('out', 'r')
lines = list(map(lambda x: [float(y)
                            for y in x.split(':')[1:]], fp.readlines()))
lines = list(map(lambda x: [x[0], x[1]*100], lines))
for line in lines:
    print(line)
# For found

ns = [lines[x][0] for x in range(0, len(lines), 2)]
```

```
times = [lines[x][1] for x in range(0, len(lines), 2)]
print(ns)
print(times)
plt.scatter(ns, times)
plt.xlabel("number of items")
plt.ylabel("Time*100")
plt.show()

# For not found
ns = [lines[x][0] for x in range(1, len(lines), 2)]
times = [lines[x][1] for x in range(1, len(lines), 2)]
print(ns)
print(times)
plt.scatter(ns, times)
plt.xlabel("number of items")
plt.ylabel("Time*100")
plt.show()
```

-First plot is for `FOUND`, and 2nd is for `NOT FOUND`


0.008
0.007
0.006
0.005
0.004
0.003
0.002
0.001
0.000
Time*100
0
5000
10000
15000
20000
25000
number of items


0.030
0.025
0.020
0.015
0.010
0.005
0.000
Time*100
0
5000
10000
15000
20000
25000
number of items