

Code

sort_test.py ×

```
1  import random
2  import time
3  import matplotlib.pyplot as plt
4  # all lists below will be used to plot the algorithm's running time
5  # store input size for bubble sort
6  bubble_sort_input_size: list[int] = []
7  # store elapsed time for bubble sort
8  bubble_sort_elapsed_time: list[int] = []
9  # store input size for insertion sort
10 insertion_sort_input_size: list[int] = []
11 # store elapsed time for insertion sort
12 insertion_sort_elapsed_time: list[int] = []
13
14 def bubble_sort(a: list[int]) -> None: 1 usage
15     """Sorts in-place using Bubble Sort from class slides
16
17     Sort by starting at the end of the list and working toward
18     the beginning. The outer loop will consist of the elements
19     that are already sorted. The inner loop is responsible for
20     sorting by swapping elements to the left.
21
22     Parameters
23     -----
24     a : list of integers
25
```

```

14  def bubble_sort(a: list[int]) -> None: 1 usage
25
26  Returns
27  -----
28  None
29  """
30
31  start: int = time.perf_counter_ns()
32  for i in range(0, len(a)):
33      for j in range(len(a) - 1, i, -1):
34          if a[j] < a[j - 1]:
35              # swap elements if current element
36              # is less than element on left
37              temp = a[j]
38              a[j] = a[j - 1]
39              a[j - 1] = temp
40  stop: int = time.perf_counter_ns()
41  # do not time anything other than sort logic
42  elapsed_time: int = stop - start
43  # print results
44  print(f'Elapsed time for input size {len(a)}: {elapsed_time}')
45  # append the data to be used for plot
46  bubble_sort_input_size.append(len(a))
47  bubble_sort_elapsed_time.append(elapsed_time)
48

```

sort_test.py ×

```
48
49
50 def insertion_sort(a: list[int]) -> None: 1 usage
51     """Sorts in-place using Insertion Sort from class slides
52
53     Sorts items by comparing the key element with values
54     considered to be already sorted. The outer loop will consist
55     of the sorted sequence while the inner loop is responsible
56     for comparing items with the key (item to be added to sorted
57     list)
58
59     Parameters
60     -----
61     a : list of integers
62
63     Returns
64     -----
65     None
66     """
67
68     start: int = time.perf_counter_ns()
69     for j in range(1, len(a)):
70         key = a[j]
71         # Insert a[j] into sorted sequence a[0...j-1]
72         i = j - 1
73         while i >= 0 and a[i] > key:
```

sort_test.py ×

```
50 def insertion_sort(a: list[int]) -> None: 1 usage
72     i = len(a) - 1
73     while i >= 0 and a[i] > key:
74         a[i + 1] = a[i]
75         i = i - 1
76     a[i + 1] = key
77     stop: int = time.perf_counter_ns()
78     # Do not time anything other than sort logic
79     elapsed_time = stop - start
80     # print results
81     print(f'Elapsed time for input size {len(a)}: {elapsed_time}')
82     # append the data to be used for plot
83     insertion_sort_input_size.append(len(a))
84     insertion_sort_elapsed_time.append(elapsed_time)
85
86 def plot_time(x: list[int], y: list[int]) -> None: 2 usages
87     r"""Plot the timing of an algorithm using Matplotlib
88
89     Parameters
90     -----
91     x: list[int]
92         This axis represents the input size sent to the algorithm
93     y: list[int]
94         This axis represents the elapsed time (in nanoseconds)
95
96     Returns
```
























sort_test.py ×

```
86 def plot_time(x: list[int], y: list[int]) -> None: 2 usages
87
88     -----
89     None
90     """
91
92     plt.plot(*args: x, y)
93     plt.xlabel('Input Size')
94     plt.ylabel('Time (ns)')
95     plt.show()
96
97
98
99
100
101
102
103
104
105
106
107 if __name__ == "__main__":
108     print("Beginning Bubble Sort on 10 lists")
109     # generate lists for bubble sort
110     # note these will be sorted in-place so do not use the same lists for insertion-sort
111     for i in range(1, 11):
112         # we want to run the sorts on 10k, 20k... 100k elements
113         x: int = i * 10000
114         input_list: list[int] = [random.randrange(start=1, stop=1000, step=1) for i in range(x)]
115         bubble_sort(input_list)
116     # Once the sort has finished plot its graph (should be quadratic)
117     plot_time(bubble_sort_input_size, bubble_sort_elapsed_time)
118
119     print("Beginning Insertion Sort on 10 lists")
120     for i in range(1, 11):
```

sort_test.py ×

```
107 > if __name__ == "__main__":
108     print("Beginning Bubble Sort on 10 lists")
109     # generate lists for bubble sort
110     # note these will be sorted in-place so do not use the same lists for insertion-sort
111     for i in range(1, 11):
112         # we want to run the sorts on 10k, 20k... 100k elements
113         x: int = i * 10000
114         input_list: list[int] = [random.randrange( start: 1, stop: 1000, step: 1) for i in range(x)]
115         bubble_sort(input_list)
116     # Once the sort has finished plot its graph (should be quadratic)
117     plot_time(bubble_sort_input_size, bubble_sort_elapsed_time)
118
119     print("Beginning Insertion Sort on 10 lists")
120     for i in range(1, 11):
121         # we want to run the sorts on 10k, 20k... 100k elements
122         x: int = i * 10000
123         input_list: list[int] = [random.randrange( start: 1, stop: 1000, step: 1) for i in range(x)]
124         insertion_sort(input_list)
125     plot_time(insertion_sort_input_size, insertion_sort_elapsed_time)
126
```

IDE showing python file and venv

▼  project-1-sorts ~/Documents/nerdgasm/grad-school/data-structu	86
▼  venv	93
>  bin	94
>  include	95
>  lib	96
>  share	97
 pyvenv.cfg	98
	99
 bubble-sort-output.png	100
 bubble-sort-time-graph.png	101
 code-1-25.png	102
 code-25-48.png	103
 code-49-73.png	104
 code-73-96.png	105
 code-97-120.png	106
 code-end.png	107 
 insertion-sort-graph.png	108
 output-insertion-sort.png	109
 p1.docx	110
 project-1-pdf.pdf	111
 sort_test.py	112
>  External Libraries	113
>  Scratches and Consoles	114
	115
	116

Output of experiment For Bubble Sort

```
Run sort_test x
Beginning Bubble Sort on 10 lists
Elapsed time for input size 10000: 2287958125
Elapsed time for input size 20000: 8713757167
Elapsed time for input size 30000: 19597288750
Elapsed time for input size 40000: 36577694542
Elapsed time for input size 50000: 57497697208
Elapsed time for input size 60000: 79294952667
Elapsed time for input size 70000: 112796306625
Elapsed time for input size 80000: 140671398917
Elapsed time for input size 90000: 185358264333
Elapsed time for input size 100000: 228805673875
```

Output of experiment for Insertion Sort

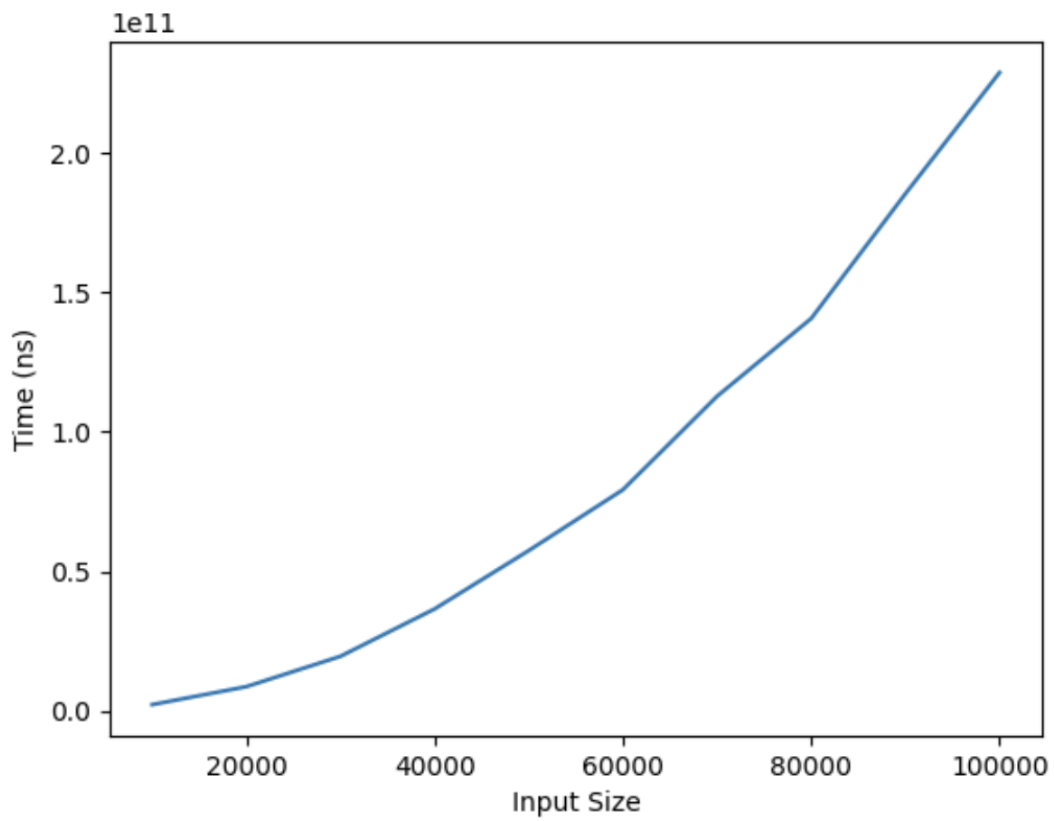
```
Beginning Insertion Sort on 10 lists
Elapsed time for input size 10000: 812344541
Elapsed time for input size 20000: 3255235292
Elapsed time for input size 30000: 7442371583
Elapsed time for input size 40000: 13504252958
Elapsed time for input size 50000: 20386126792
Elapsed time for input size 60000: 30666097125
Elapsed time for input size 70000: 40006711500
Elapsed time for input size 80000: 52390396250
Elapsed time for input size 90000: 66243761625
Elapsed time for input size 100000: 83835616167
```


Graph for Bubble Sort

Plots

🔍 📏 ⊕ ⊖ 1:1 🖼️ ✎

640x480 PNG (32-bit color) 17.01 kB



Graph for Insertion Sort

