# Self-Test & Grading

You have experienced how your assignments have been graded for a couple of months. Now, I ask you to grade your assignment by yourself.

* **"timeit**" means that you measure the elapsed time with 1 Million nodes if the time complexity of the function is O(n), and 100,000 nodes for O(n^2) in your computer.
* **Mark X if you have not implement or if it does not work.**
* You may expand this form as needed.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Steps** | **pnts** | **Functions** | **Self-Grade**  **or Mark X** | **Use this column to show your test result or comments about your test results.** |
| Step 1 | 1.0 | push\*() |  | Include test case: 1st and last node, large list |
| Step 2 | 1.0 | pop\*() |  | Include test code: pop all, 1st and last node large list |
| Step 3 | 1.0 | show() |  | test case: small or large n |
| Step 4 | 1.0 | "F", "B",  "Y", "P" |  | timeit:  F  P:  B:  Y: |
| Step 5 | 2.0 | push\_N(),  pop\_N() |  | Modify driver file from this step  timeit:  F  P:  B:  Y: |
| Step 6 | 2.0 | reverse\_ in\_stack() |  | timeit: |
| Step 7 | 2.0 | reverse\_ in\_place |  | timeit:  this must be much faster than step 6 |
| Step 8 | 2.0 | reverse\_ odd2() |  | timeit:  include test cases: all odds, all evens |
| Step 9 | 3.0 | reverse\_ oddn() |  | timeit:  include test cases: all odds, all evens |

Your comments overall if necessary:

Analysis:

1. Estimate the elapsed time of push\_backN and pop\_backN for running 1 million of nodes.  
   Use the elapsed time for 100,000 nodes you are getting from your machine, respectively.
2. Estimate the elapsed time of reverse\_odd2() running for 1 million of nodes.  
   Use the elapsed time for 100,000 nodes you are getting from your machine, respectively.
3. Analyze the elapsed time of running reverse\_using\_stack() and reverse\_in\_place() with a million nodes.