Project 2 - Source code (Phase 1)

Due May 14 by 11pm **Points** 180 **Submitting** on paper

CS 165 - Project 2 - Testing Solution Quality (Phase 1)

180 points

Project 2 involves testing various bin packing algorithms experimentally to determine the quality of the solutions they produce. The specific bin-packing algorithms you should implement are the following five algorithms (please see the course notes for details on these algorithms):

1. Next Fit (NF)
2. First Fit (FF)
3. Best Fit (BF)
4. First Fit Decreasing (FFD)
5. Best Fit Decreasing (BFD)

For Phase 1 of Project 2, you will need to submit your source code, implementing each of the above algorithms, electronically to Gradescope using the course Gradescope link

Please use the supplied files, **requirements.py (https://canvas.eee.uci.edu/courses/55229/files/22534826?wrap=1)** **(https://canvas.eee.uci.edu/courses/55229/files/22534826/download?download_frd=1)** and **zipzip_tree.py (https://canvas.eee.uci.edu/courses/55229/files/22430658?wrap=1)** **(https://canvas.eee.uci.edu/courses/55229/files/22430658/download?download_frd=1)** , to define your functions. **next_fit.py (https://canvas.eee.uci.edu/courses/55229/files/22430663?wrap=1)** **(https://canvas.eee.uci.edu/courses/55229/files/22430663/download?download_frd=1)** is provided as an example. Your source code will be auto-graded, both for the bin-packing algorithms and your zip-zip tree data structure. If you have any questions about these files, please ask the TA or the Reader. Please test your code using the file **project2_tests.py (https://canvas.eee.uci.edu/courses/55229/files/22430661?wrap=1)** **(https://canvas.eee.uci.edu/courses/55229/files/22430661/download?download_frd=1)** .

Note: For your implementations of First Fit and Best Fit (including the FFD and BFD versions), you must implement these methods to run in $\mathcal{O}(n \log n)$ expected time. Moreover, your solutions must use a balanced binary search tree known as a zip-zip tree (please see the course video and notes for details) and you must implement this data structure yourself, not from a library.