Based on my experimental observations as well as reference from C++11 standards, the fundamental difference between cout in the manner where multiple objects are being concatenated (threadID + “Hello World” + endl), is that although each of these individual writes to output stream is atomic, but other threads can interleave between these three individual writes.
Thus we can expect one thread writing threadID to output stream, then another thread writing its threadID and first thread writing Hello World etc. Thus output is interleaved between these three elements of multiple threads.

For printf, it constructs a buffer of the formatted string, and then it writes to the output stream. Since write to output stream is thread safe and atomic, it does not have an interleaving issue. Although any concurrent thread can jump in and print its output first, but printf ensures that whole output is printed in atomic fashion before another thread jumps in.

My graphing solution is tested to work with Python 3.6 and Python 3.10, along with pandas and matplotlib libraries. The graph output is stored in png file in same directory named output.png. It can be invoked through the commandline and it needs the csv file path as an argument. For example:
python3 graphing.py ./../results.csv
would generate output.png in the script directory.