CSCI 2500 — Computer Organization
Lab 11 (document version 1.0)

- This lab is due by the end of your lab session on Wednesday, November 20, 2019.
- This lab is to be completed **individually**. Do not share your code with anyone else.
- You **must** show your code and your solutions to a TA or mentor to receive credit for each checkpoint.
- Labs are available on Mondays before your lab sessions. Plan to start each lab early and ask questions during office hours, in the discussion forum on Submitty, and during your lab session.

1. **Checkpoint 1:** This lab focuses on material from Chapter 5 regarding cache memory. For the first checkpoint, download the `lab11.c` code. Walk through the code to understand what is expected overall.

   Figuring out how many blocks (also known as slots or lines) exist in your direct-mapped cache is rather straightforward. To do so, determine how many possible index bit combinations exist. For example, with two bits, we have binary values `00`, `01`, `10`, and `11`, which is simply 2^2.

   Implement this in the `block_count()` function. Test with the cases shown in `main()`, but also be sure to add more test cases to ensure your code works.

2. **Checkpoint 2:** For the second checkpoint, you will determine the index of a would-be cache entry via the `get_index()` function. More specifically, given an address, a starting point for your index, and the ending point of your index, implement logic in the `get_index()` function to find the appropriate index in a direct-mapped cache.

   As with the previous checkpoint, test with the cases shown in `main()`, but also be sure to add more test cases to ensure your code works.

3. **Checkpoint 3:** For the third checkpoint, you will determine the size of a cache block via the `get_cache_block_size()` function. More specifically, given the starting and ending point of your index as well as the block offset bits (BOBs), determine the total cache block size.

   As with the previous checkpoints, test with the cases shown in `main()`, but also be sure to add more test cases to ensure your code works.