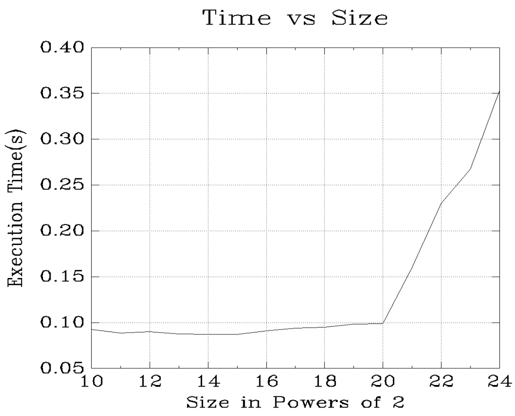
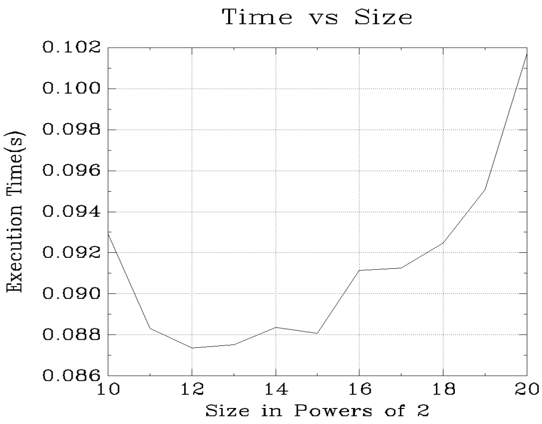
Task 1: Calculating the sizes of the cache for my Computer

- **The Idea v1: Access different sizes of arrays, note the time taken, and perform analysis**
    - **Errors**: Segmentation fault while allocating large-sized arrays, limitations of int variables.
    - **Analysis**: To overcome this, we must somehow overcome the limitations of the allocation limits.
- **The Idea v2: Statically allocate a large-sized array and perform iteration with limits. For the int variable, use unsigned long int**
    - **Errors**: Nothing meaningful data was gathered after running the program. The plot is just a simple exponential graph.
    - **Analysis**: Since we are accessing different-sized arrays linearly, the time taken increases as we increase the array size. We can’t compare data like this.
- **The Idea v3: Access different sized arrays a fixed number of times. So that we can compare the time required with other arrays. (Works!!)**
    - **Description**: By default, the program allocates arrays of 1 KB to 16 MB, which can be altered by passing command-line parameters. The program makes an equal number of accesses to each array element at every cache line. The procedure is performed ten times for all the arrays, and the time spent is recorded each time. The average time spent is then calculated and displayed. The time is the total time required to access all the elements from the different caches. (including hits and misses)
    - **Observations:**
        - **Measured Values:**
          L1 Cache: about 64 KB
          L2 Cache: about 512 KB
          L3 Cache: about 2 MB
        - **Real Values:**
          L1 Cache: 64 KB
          L2 Cache: 512 KB
          L3 Cache: 2 MB





The plots show a sudden increase in execution time at 16, 19, and 21. Since the time has increased suddenly, we can conclude that the previous array size is the cache size.