1. Write a C program that receives a number N as a command line argument. The main process creates N groups of 3 threads each. After creating one group of threads, the main process waits for them to terminate before starting the next group. The main process starts all three threads of each group without any delays. One thread from each group will populate the elements of an array with 10 positions with random values between 0 and 100. After the array is fully populated, the other two threads each iterate through the elements of the array. Whichever thread encounters a non-zero value, it adds it to a local sum and sets that array value to 0, then sleeps for 10 milliseconds. Once the threads finish iterating through the array, they print their respective local sums and terminate.

2. (Optional) Write a C program that calculates the sum of elements on each column of a matrix of integers. The implementation will calculate the required sums sequentially and then use threads (one thread per column). Measure the execution time for both cases. Use the following matrix to test: https://www.cs.ubbcluj.ro/~horea.muresan/os/sol-c/bigmat
   To measure the execution time in seconds you can use:

   ```
   struct timeval tv1, tv2;
   gettimeofday(&tv1, NULL);
   // CODE HERE
   gettimeofday(&tv2, NULL);
   printf("Total time = %f seconds\n", (double)(tv2.tv_usec - 
   tv1.tv_usec) / 1000000 + (double) (tv2.tv_sec - tv1.tv_sec));
   ```

   Which will print the execution time of the code placed between the two calls "gettimeofday".