Em primeiro lugar, o código do nosso projeto foi elaborado pensando na implementação dos métodos que aprendemos na sala de aula, como deixar mais rápido a leitura do array e encontrar as triplas que forem encontradas dentro do array pelos valores que o usuário colocará, nesse caso, temos várias funções onde a desenvolvemos na pasta “threesum.c”. Sendo assim, na pasta “threesum.c” fizemos a implementação “treeSumForcaBruta” e “treeSumMelhorado“ onde mostra a análise feita de forma normal no “treeSumForcaBruta” e da implementação tanto do MergeSort quanto ao do Busca Binaria, que facilitará o tempo de execução do nosso “treeSumMelhorado“, também calculamos a quantidade de operações da “treeSumForcaBruta” e do “treeSumMelhorado“ através da função “ImprimeQtdOperacoes”. Posto isso, implementamos elas para a pasta “main” , compilamos e executamos o código que na saída irá aparecer os resultados dos cálculos feitos pela “treeSumForcaBruta” e pelo “treeSumMelhorado“ aparecendo o array informado e ordenado que o usuário colocou e logo em seguida as quantidades de operações.

Destarte, também vimos que o tempo de execução (T(n)) para se achar as triplas encontradas utilizando o método força bruta demora mais tempo do que o método utilizando a forma melhorada. O motivo disso é a variação do código onde na força bruta achamos o N (número do array) percorrendo I, J e K tendo assim o nosso N cúbico; já no método melhorado, utilizamos a implementação do MergeSort e da Busca Binaria, com isso o tempo de execução do nosso código diminui pois temos (n.lgn) representando o MergeSort somados mais a linha da Busca Binaria ao invés de ser N cúbico como o exemplo da força bruta, teremos (n^2.lgn).