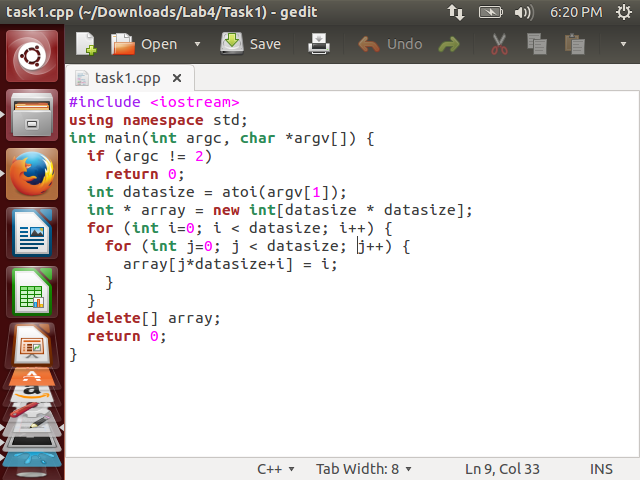
LAB-4

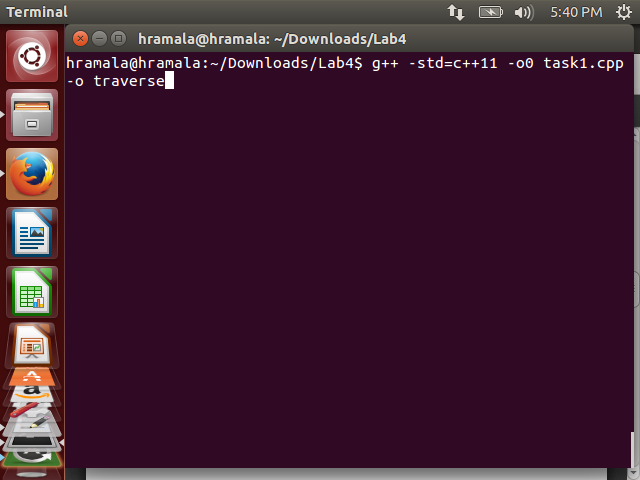
Task1: Matrix Traverse

Solution:

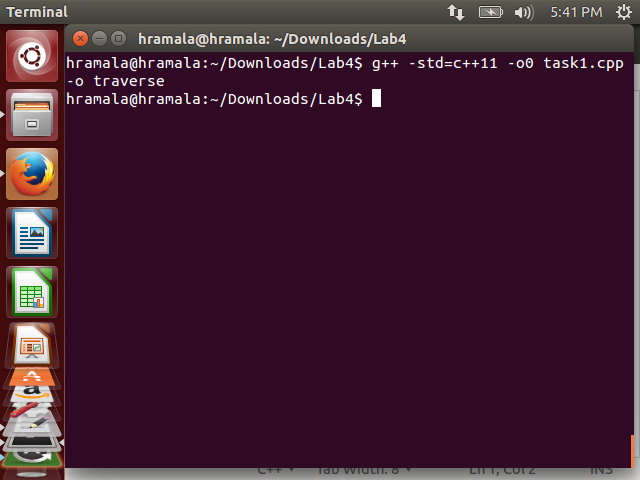
Task 1 Code



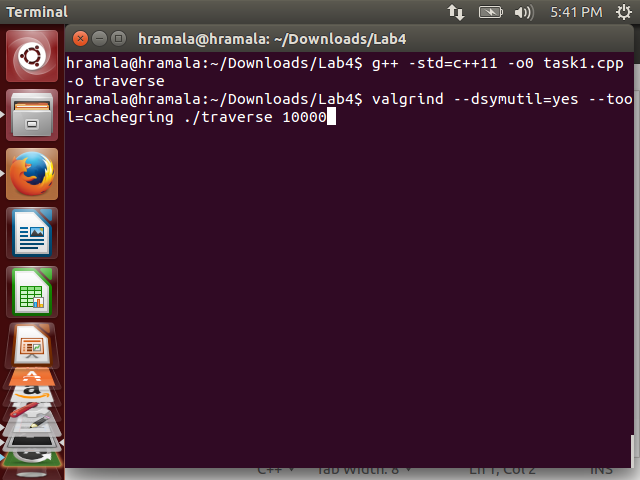
Compiling task 1



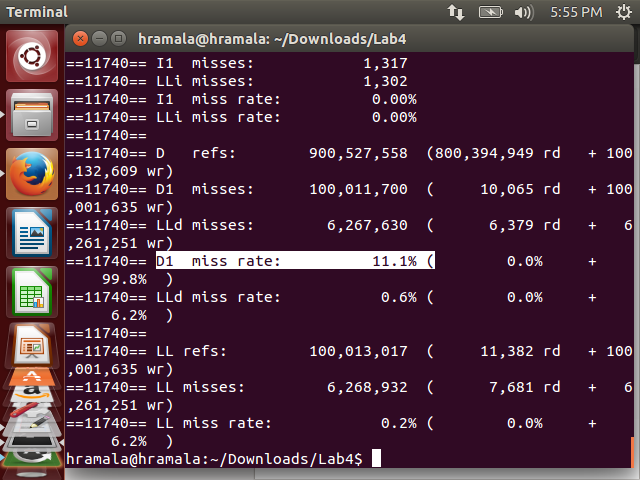
Compilation done



Running Task 1

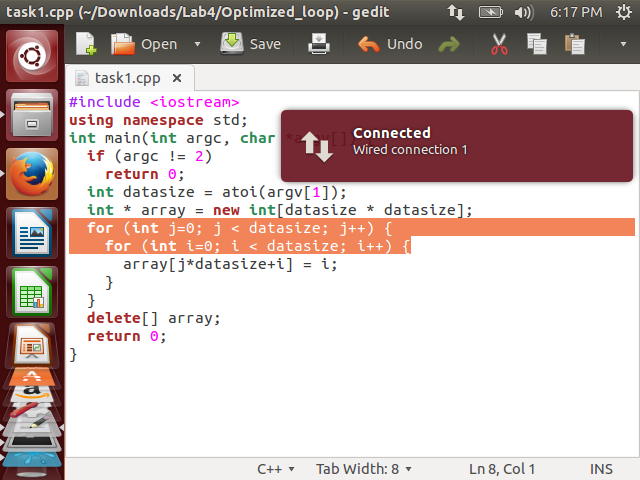


1. **Original D1 Miss Rate:**

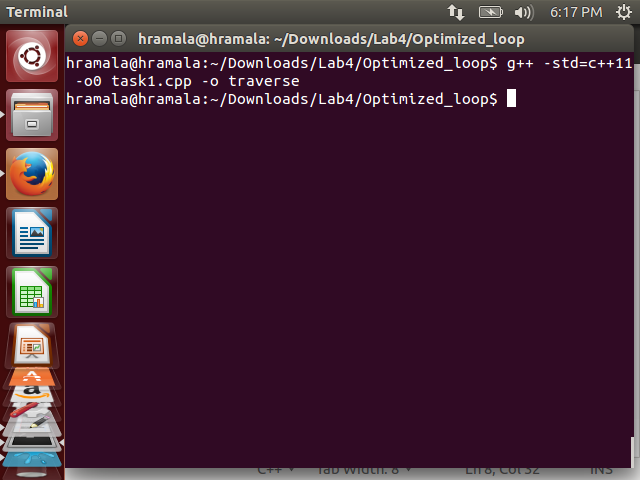


­­

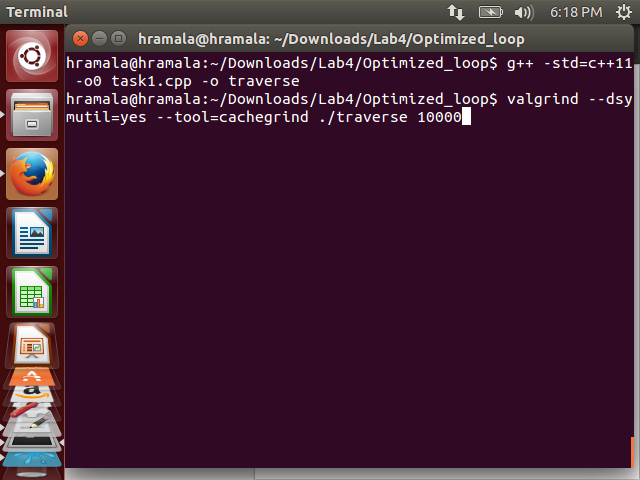
1. **Optimized Task 1 Program:**



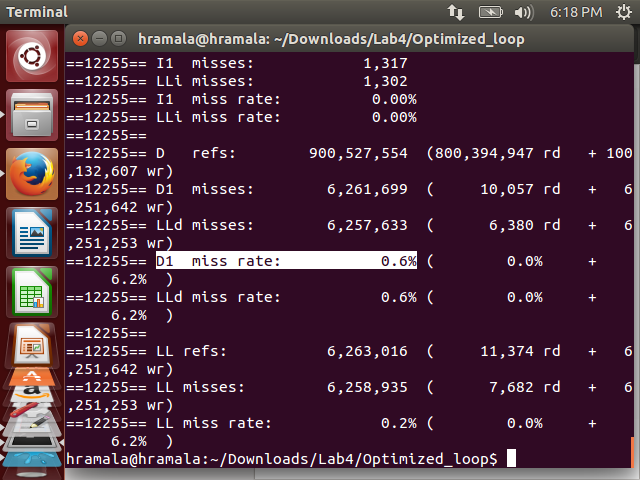
Compiling optimized code:



Running Optimized program



1. **D1 Miss Rate of Optimized program:**



1. **Explanation:**

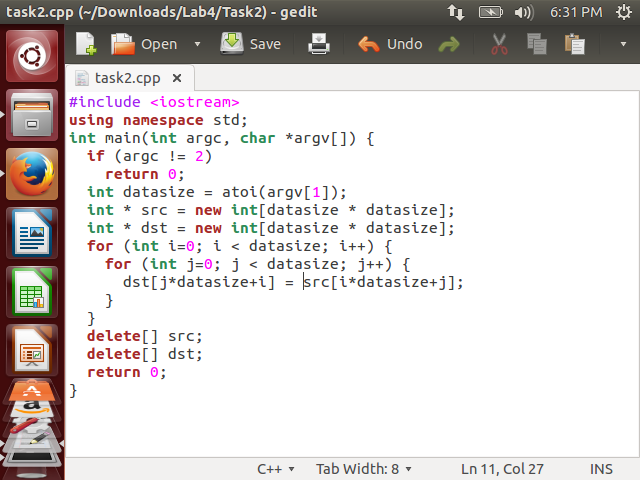
The technique used to optimize this program is **"Loop Interchange".**

By switching the order in which loops execute, misses can be reduced due to improvements in spatial locality. These loops cause a miss on each memory access because of the long stride given by index j in the inner loop. By switching the order of the loops, the stride is changed to 1, allowing the elements to be accessed in sequential order. This reduces the D1 Miss rate as shown in the above image.

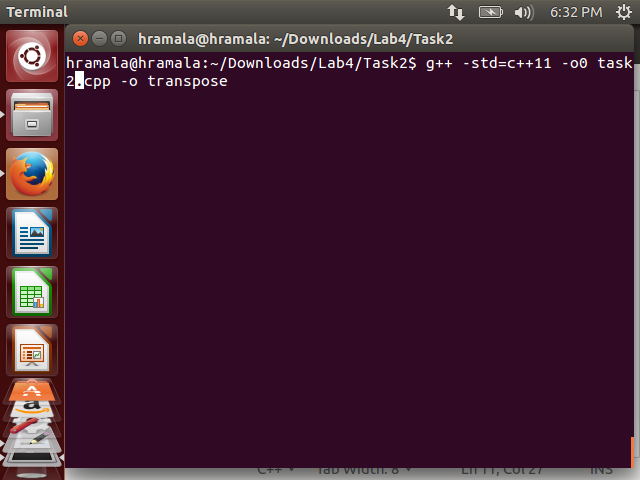
Task2: Matrix Transpose

Solution:

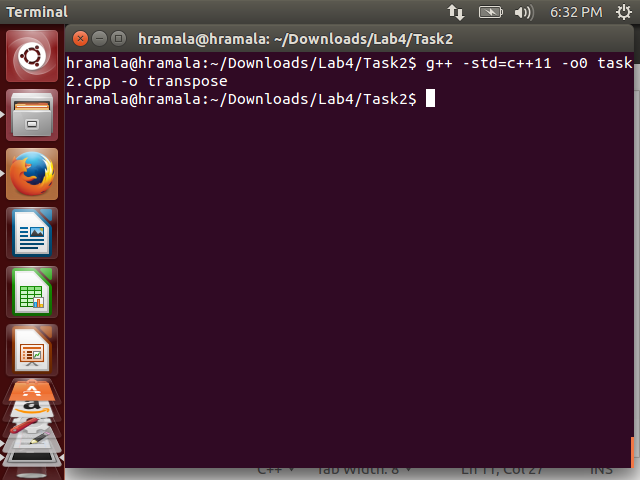
Task 2 Code



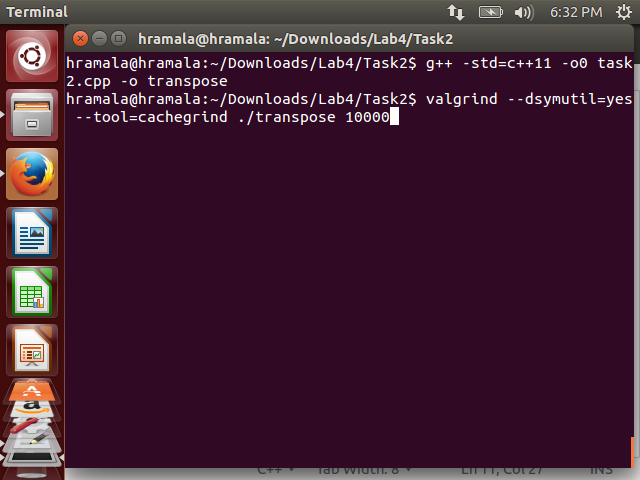
Compiling Task 2



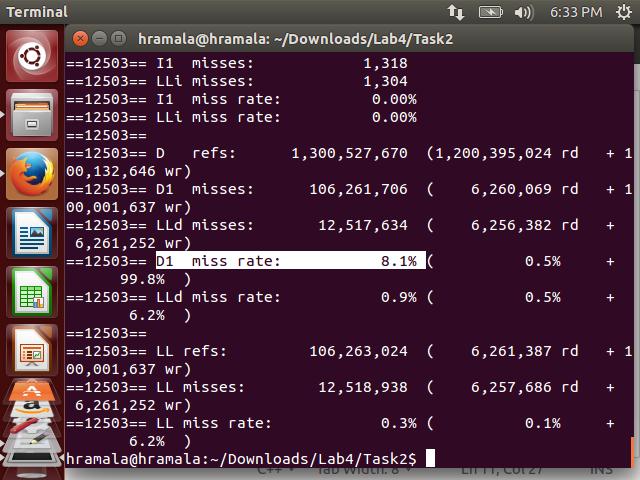
Compilation done



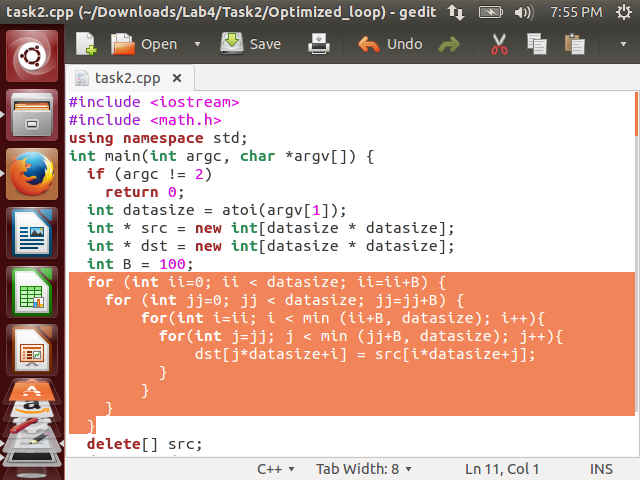
Running Task 2



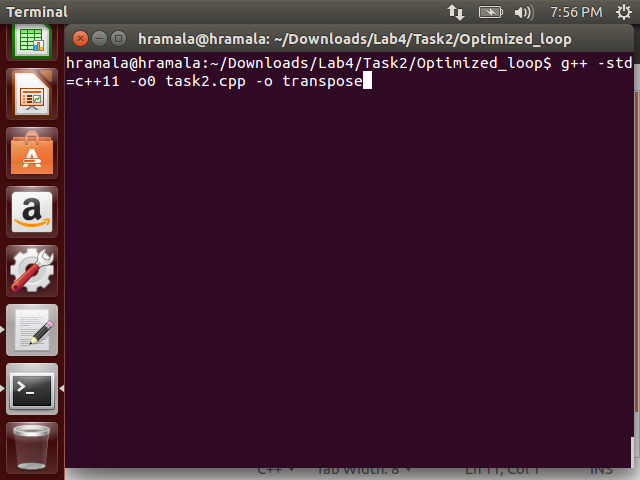
1. **Original D1 Miss Rate:**



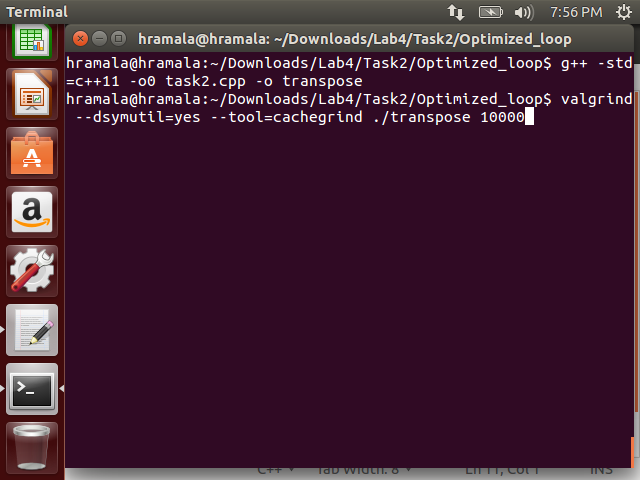
1. **Optimized Task 2 Program:**



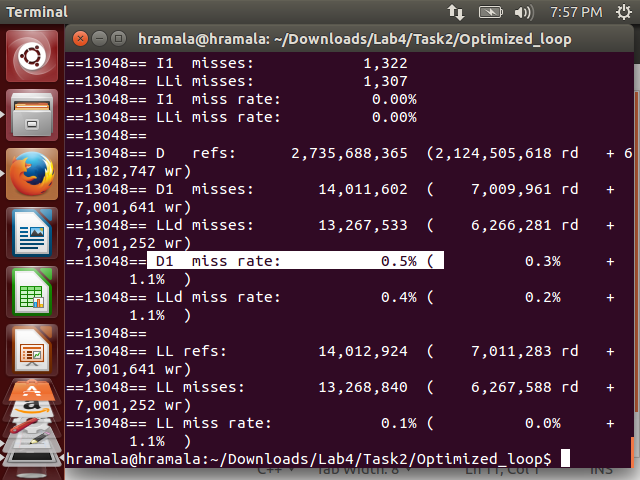
Compiling Optimized Task 2



Running Optimized Task 2 Program



1. **D1 Miss Rate of Optimized Program:**

****

1. **Explanation:**

The technique used for optimizing Task 2 is called **"Blocking".**

Optimized matrix transpose requires the cache to hold the minimum elements required for the transpose. Capacity misses can occur for large matrices since it may not be possible to store all the elements required to do transpose in the cache. Blocking operates on blocks (submatrices) and reduces the total number of memory words (elements) accessed by a factor of B (the blocking factor). This reduces the D1 Miss Rate as shown in the above image.