ASSIGNMENT 2

Data Structures Laboratory



**Rishi Chordia**

18118052

BTech CSE

# 

# 

# Problem Statement 1:

# A Problem to implement a simple transposition cipher, where this cipher encrypts and decrypts a sequence of characters by dividing the sequence into blocks of size n, where n is specified by the encryption key. If the input text has a length that is not a multiple of n, the last block is padded with null characters ('\0'). In addition to n, the key also specifies two parameters a and b. For each block, the i-th output character, starting from 0 as usual, is set to the j-th input character, where j = (ai + b) mod n. For appropriate choices of a and b, this will reorder the characters in the block in a way that can be reversed by choosing a corresponding decryption key (n, a′, b′).Write a program transpose.c that takes n, a, b, inputfile.txt in argv[1], argv[2], argv[3], and argv[4], respectively, applies the above encryption; and writes the result to outputfile.txt. Further, write a program inverseTranspose.c that decrypt the outputfile.txt and result in a new file named decryptedOutputfile.txt. Finally, write a program compareFiles.c to find the equivalence between the inputfile.txt and decryptedOutputfile.txt files.

# Algorithms and Implementation:

# With the use of simple character arrays ,strings have been stored and operated on

# Simple file reading and writing techniques have been used

# GDB used for debugging

# Problem Statement 2:

# Write a C program, MAT.c to represent any region (in image array representation), into its quadtree form. region can be represented either by its interior or by its boundary. Here we represent the region by its interior using one of the most common methods called image array. In this case we have a collection of pixels. Since the number of elements in the array can be quite large,the main objective is to reduce its size by aggregating equal-valued pixels.A general approach is to treat the region as a quadtree, where the region is represented as a union of maximal non-overlapping square blocks whose sides are in power of 2. The quadtree can be generated by successive subdivision of the image array into four equal sized quadrants. If the sub-array does not consist entirely of 1s or entirely of 0s, it is then further subdivided into quadrants and sub-quadrants, etc.

# Algorithms and Implementation:

# Recursion was used to read the matrix and correspondingly store the related values

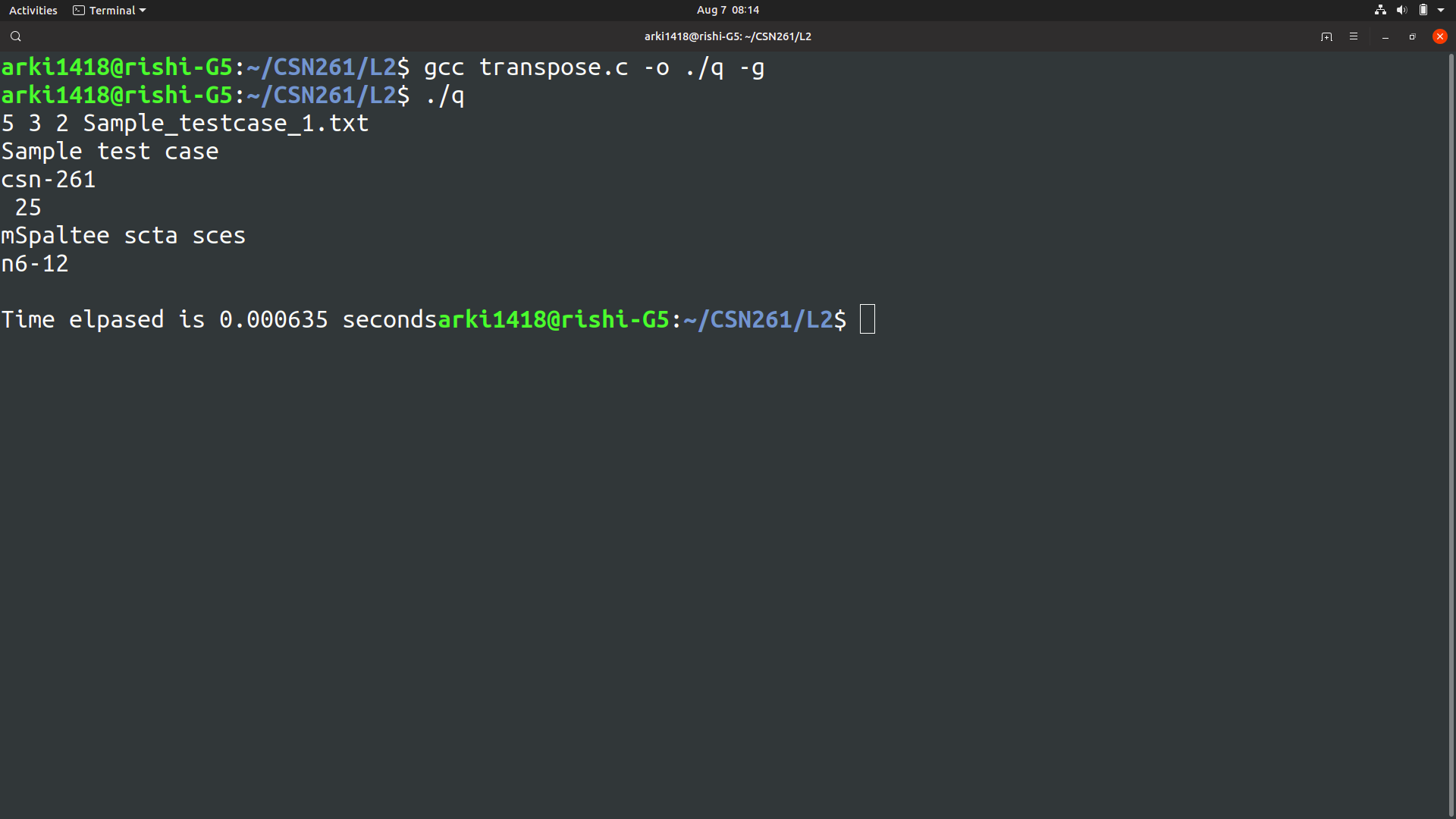
# A simple tree data structure was used to represent the matrix as given in the question

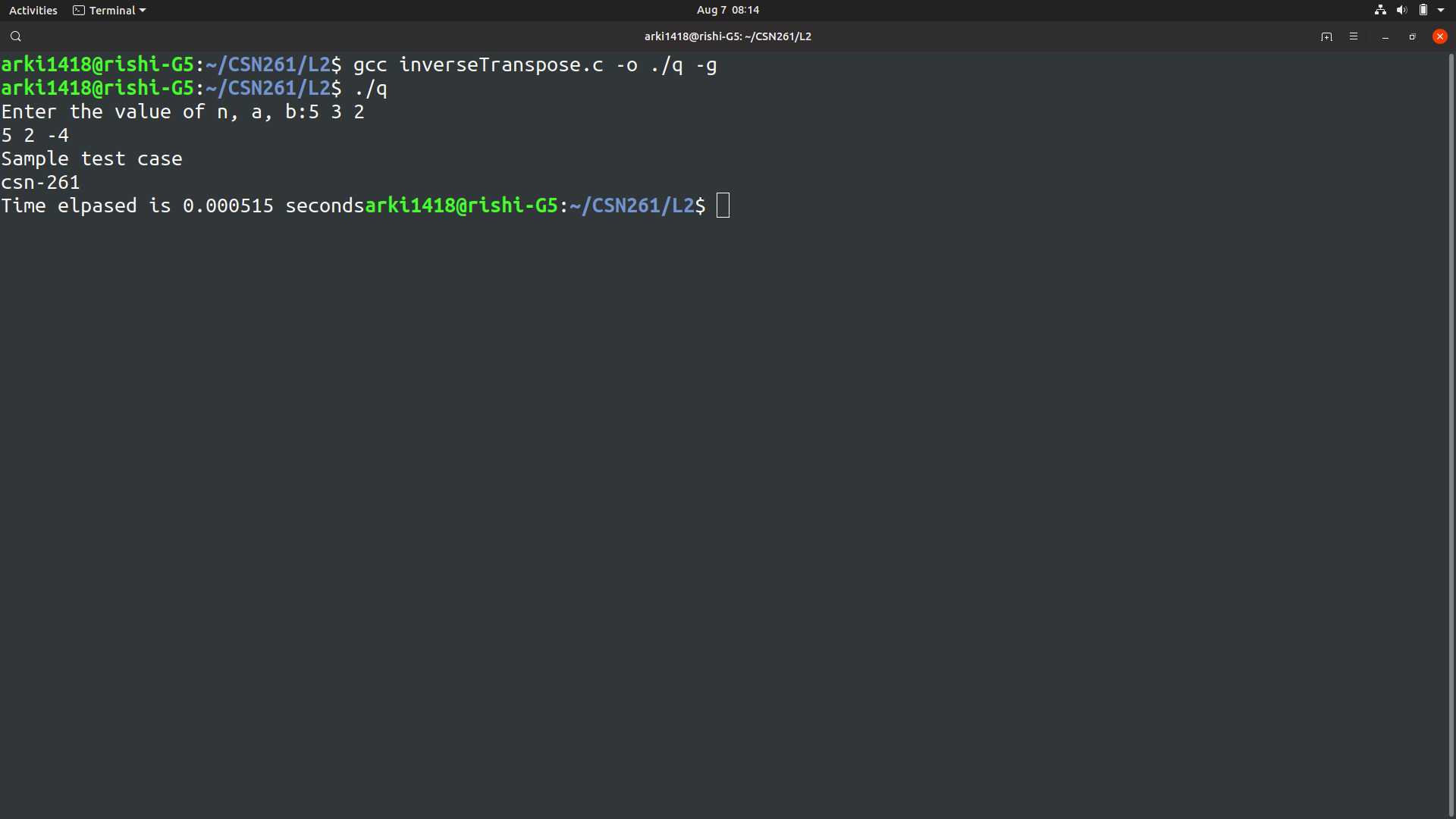
# GDB was used for debugging

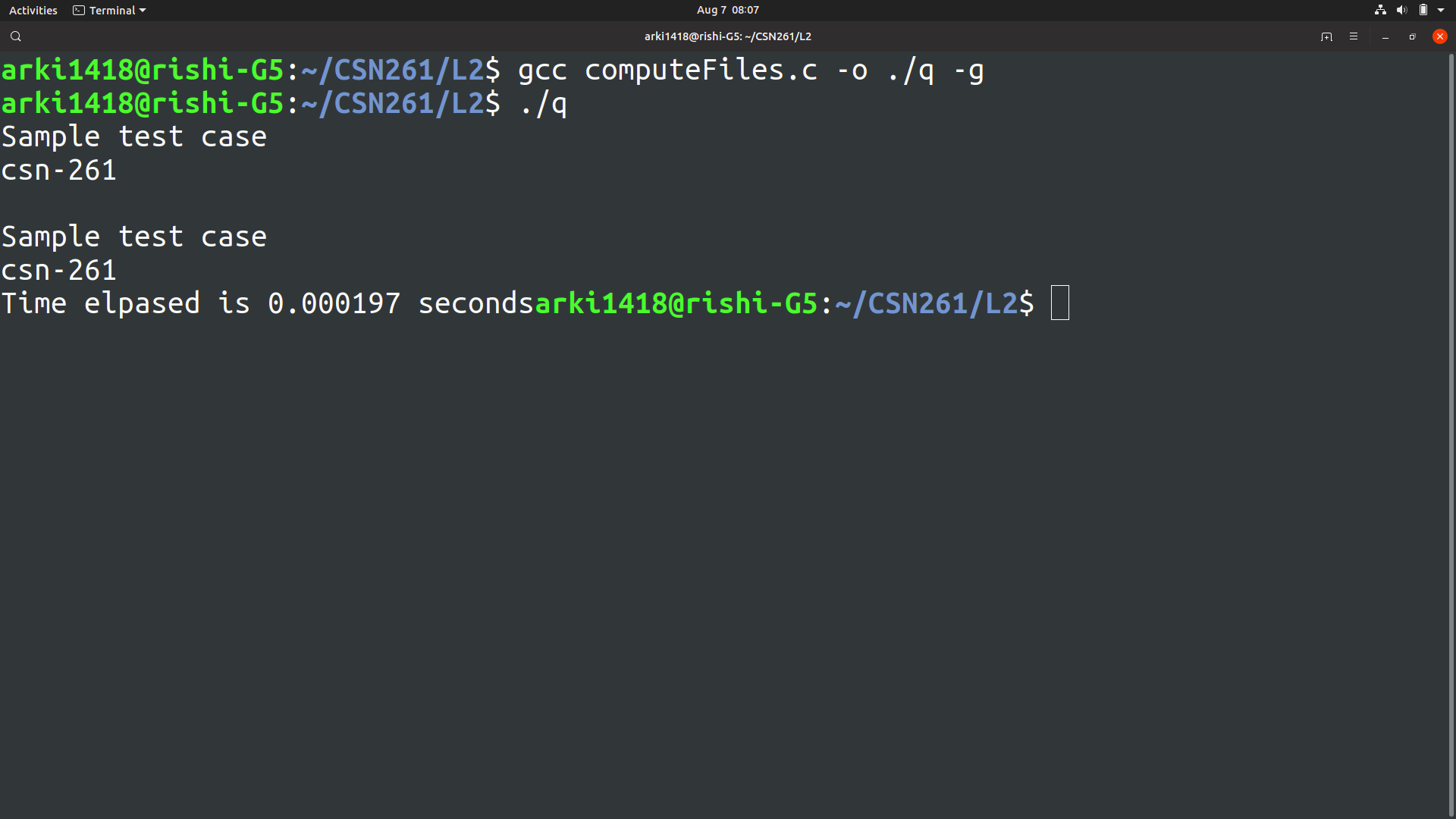
# 

# Snapshots and Computation Time

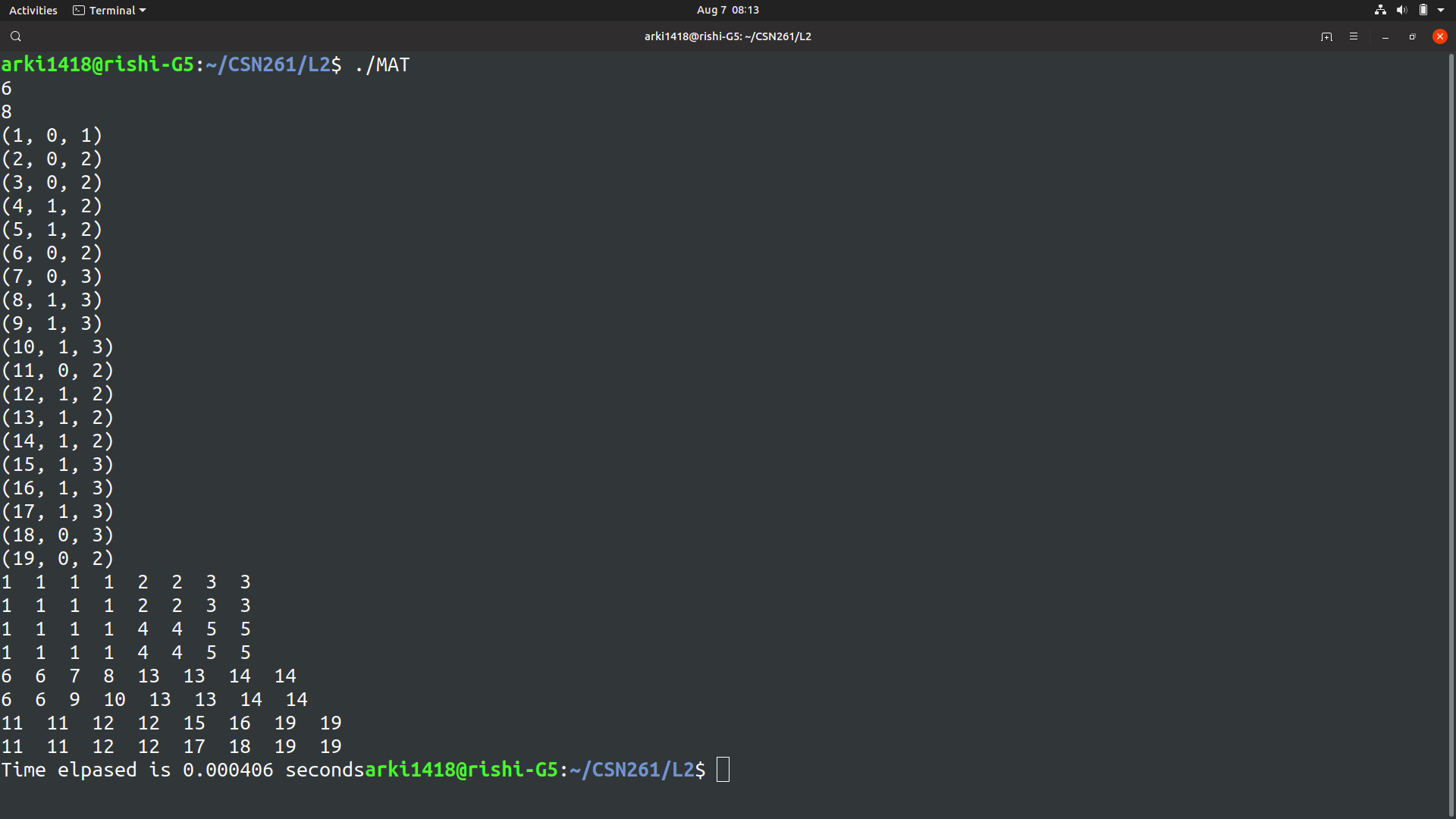
Question 1:







Question 2:



￼