## horizontal line



Set Operations On Numbers

02.10.2017

**DS project for 3rd SEM - Section-B**

*52 Anurag Mishra*

*53 Aman Saxena*

*54 Rishit Ratna*

*55 Vyshnav Ayilliath*

*56 Anmol Nanchahal*

# Problem Statement

Write a C application to implement a complete library for Set Operations on numbers. The library must provide options for creating sets, you are free to use any data structure. Perform the following Operations:

i) Union of Sets v) Compute Power Set

ii) Intersection vi) Set Difference

iii) Set Membership vii) Set Inclusion

iv) Complement viii) Cartesian Product

# Data Structure Used:

Doubly Linked List without header node and not circularly linked.

*typedef struct* Node{

*int* number;

*struct* Node \*right\_ptr;

*struct* Node \*left\_ptr;

}Node;

# Algorithms

## Union of Sets

1. Step 1: Join Both the linked list
2. Step 2: Delete every Duplicate element

## Intersection of sets

1. Step 1: For each element in the first linked list, iterate through the 2nd list.
2. Step 2: Repeat until end of list 1.

## III. Set Membership

1. Step 1: Ask user for an element to compare
2. Step 2: Iterate through list and Check if it is present in list 1.
3. Step 3: Do the same for list 2.

## IV. Compliment

1. Step 1: Take list 1 as the Universal Set.
2. Step 2: Display all elements present in list one but not in Universal Set.

## V. Power Set

## Step 1: Find the number of elements(n) in the given List

1. Step 2: Run a for loop for 0 to n-1
2. Step 3: For each iteration a different Sequence of binary codes will be generated which selects whether an element is there in that set or not.
3. Step 4: Display each of combination.

## VI. Set Difference

## Step 1: Assume list 1 - list 2

1. Step 2: For every element in list 2 run a loop
2. Step 3: Check if the Corresponding number is present in list 1 and delete it.

## VI. Set Inclusion

1. Step 1: Check if every element of list 2 is present in list 1
2. Step 2: If condition is satisfied, then list 2 is a subset of list 1.

## 

## 

## VI. Cartesian Product

## Step 1: For an element is list 1, make a pair with every element of list 2

## Step 2: Do this for every element in list 1.

# Functions Used

## I. Ins\_node : Inserts nodes into a given Linked List.

## II. Delete: Delete an element from a node

## III. Display: Display a linked list

## IV. UnionList: Union of two lists

## V. IntersectionList: Intersection of two Lists

## VII. SetMemberShip : Check the membership of an element in a list

## VIII. Complement: Find the complement of a Set

## VII. PowerSet: Find the PowerSet of a Set

## VII. SetDifference : Difference between Two sets

## VII. SetContainment: Checks if one set is a subset of another

## VIII. CartesianProduct: Finds cartesian Product of two sets

## IX. CreateSet: Creates a Set

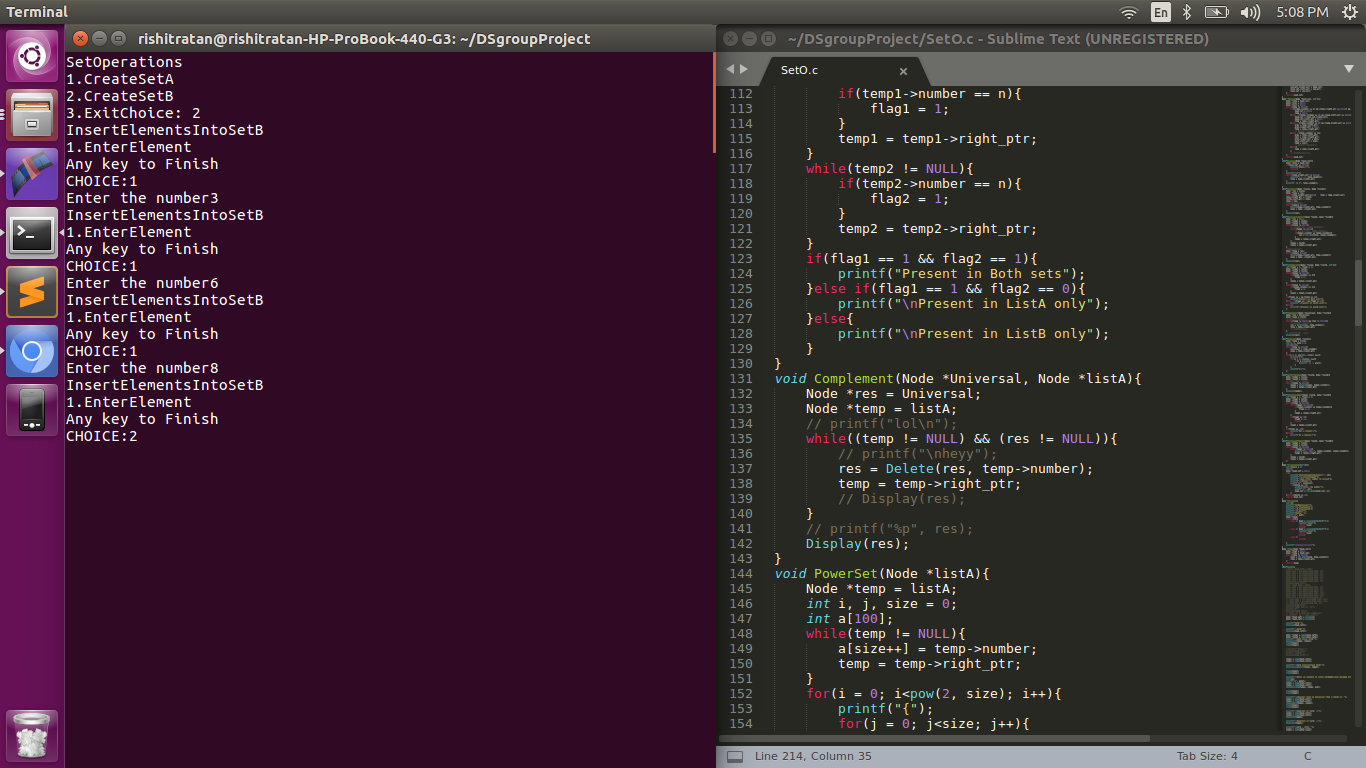
## X. Screen: Displays the required Screen

## XI. Copy: Copies one Linked List to the other

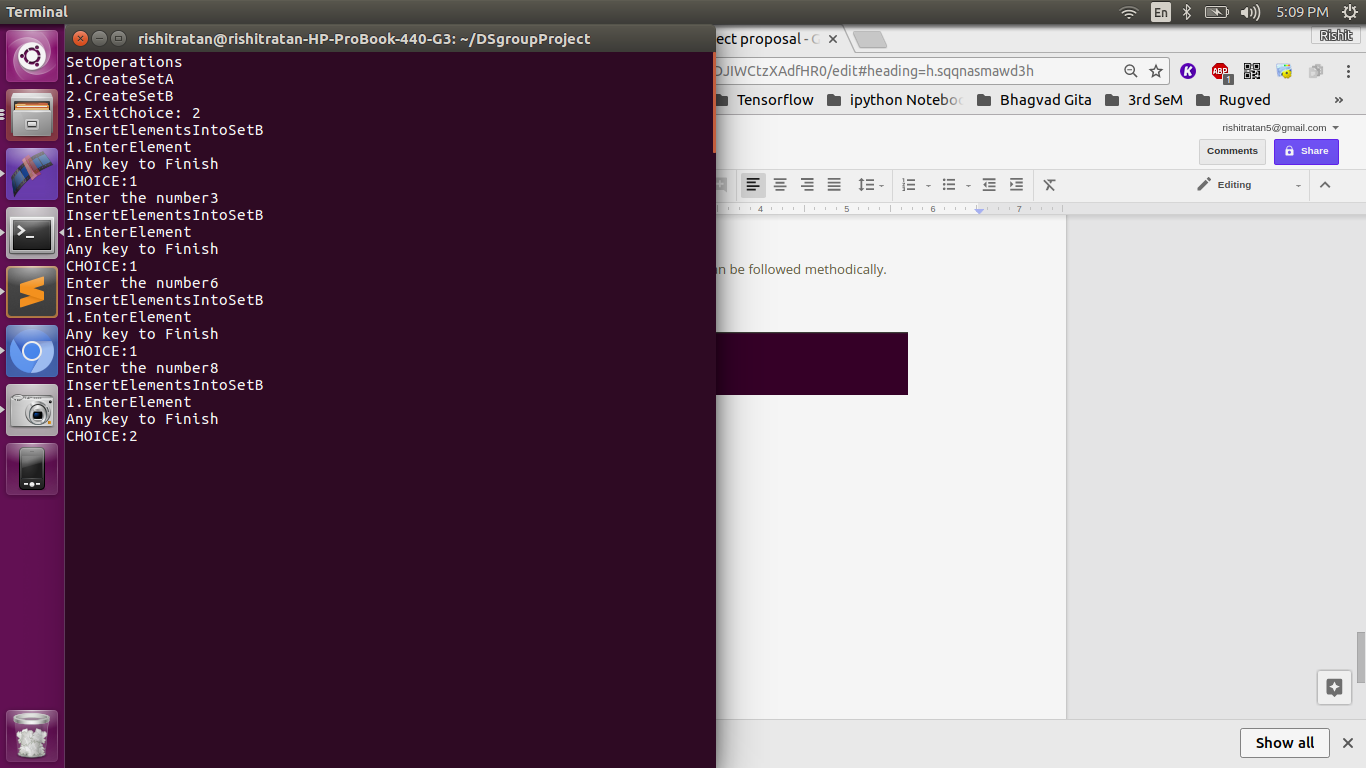
# Input/Output

It is a menu driven Program. Hence Every step can be followed methodically.

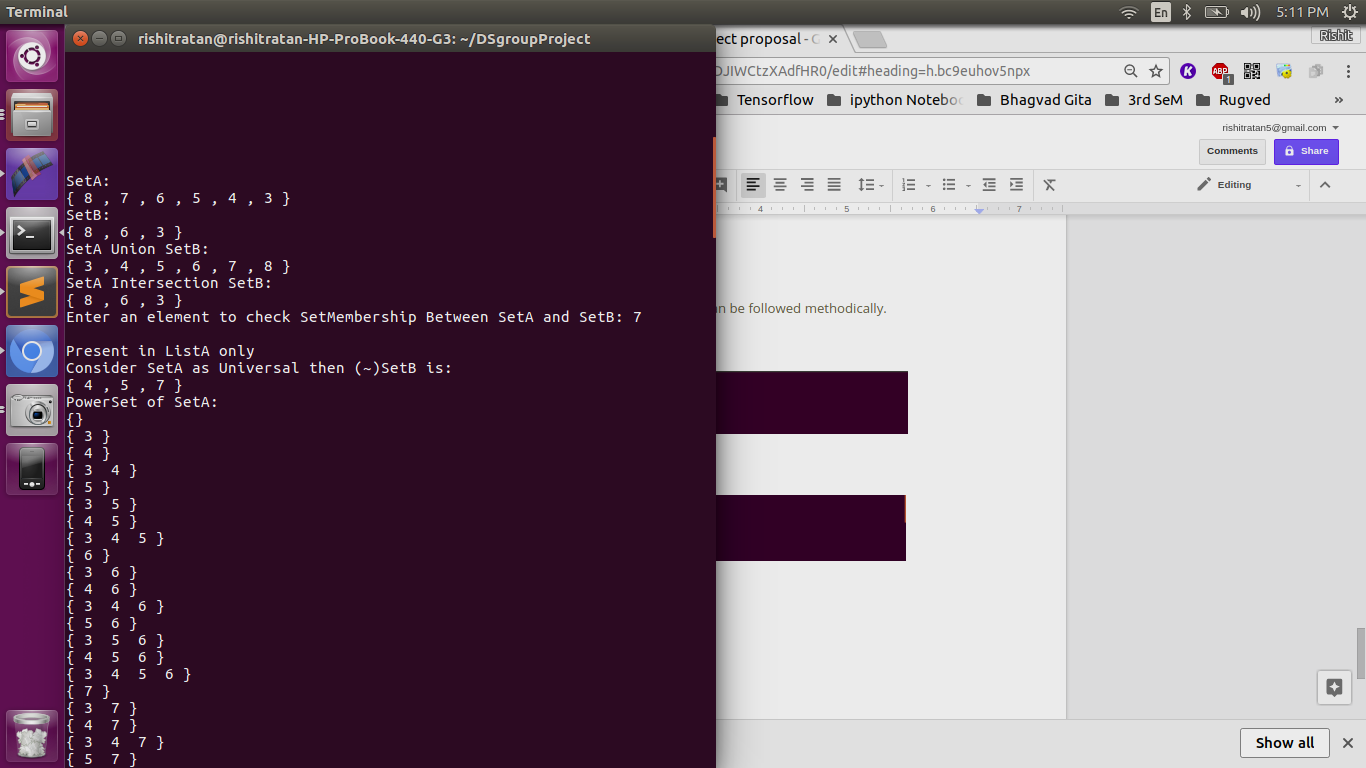
## I. Creates a Set



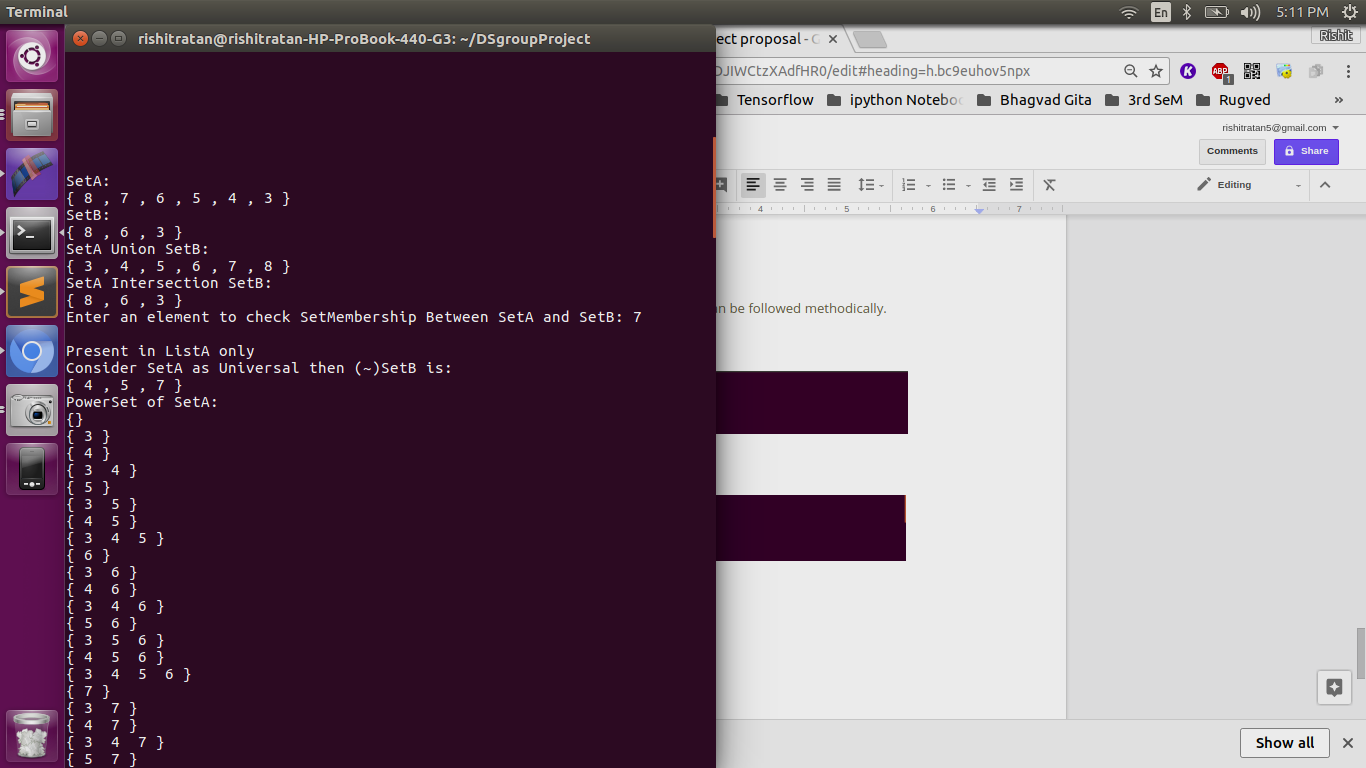
## II. Enter Elements



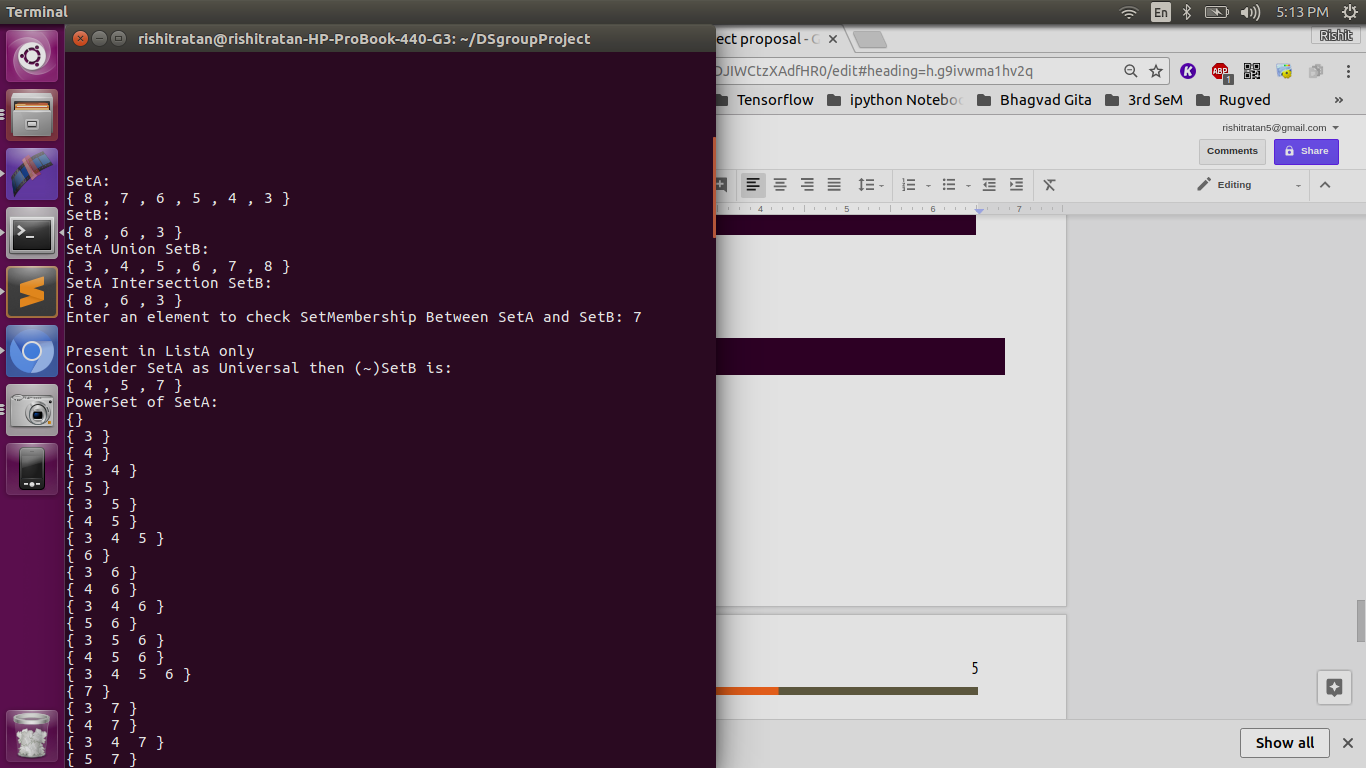
## III. Output:



## IV. Union:

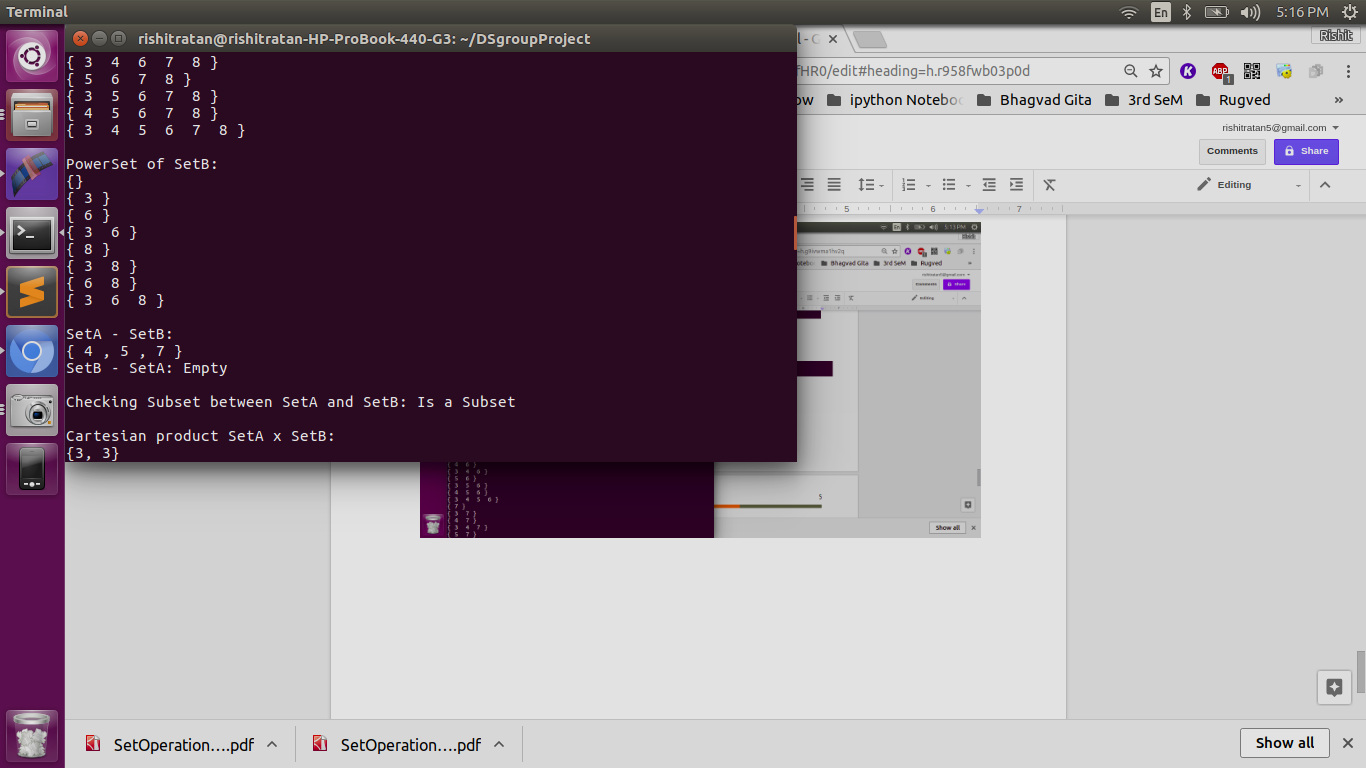


## V. Intersection



## 

## VI. Power Sets



## VII. Cartesian Product

