Project #1 Report

CSC 8980: Distributed Systems

(Group #3 -** Pooja Baba, Ishu Goyal, Dinesh Kumar Pullepally, Samvidha Reddy Mannem**)

Problem Statement

Using a programming language of your choice, implement an algorithm that **generates the sets** required for **Maekawa's mutual exclusion algorithm**. You must implement a simulated distributed computing environment in which nodes join and leave the distributed system. Your simulated system must be able to deal with any number of nodes, i.e., you may have to compute **degenerate sets**. Further, your simulation should **allow nodes to be added and/or fail**. This will require the **dynamic recalculation of the sets**. You may assume that all nodes can be informed before a node fails or is added.

Algorithm:

The code is implemented using **python** language. It can be executed in the terminal using the command –

```
python project1-group3.py
```

1. Conditions for set creation

The following conditions are considered for the creation of sets –

i. $\forall i \forall j: i \neq j \quad 1 <= i, j <= N :: R_i \cap R_j \neq Ø$
ii. $\forall i: 1 <= i <= N :: S_i \in Ri$
iii. $\forall i: 1 <= i <= N :: |R_i| = K$
iv. Any site S_i is contained in exactly K sets

2. Choice of S_i's

- N: number of nodes in the system
- K: number of items in each set
- N can be expressed in the form of K as $N = K(K-1)+1$

3. Algorithm for set creation

To create the sets, we will perform the following steps –

```
1)  User inputs a number N, which indicates the number of nodes we wish to have in the system.
2)  Check if N can be expressed as K(K-1)+1 using brute force method as below-
      a. For i in range (round(N/2) + 1):
              if N == i*i – i + 1
Using the above snippet, we get the value of K, i.e., we get the cardinality of each set
3)  If N = K(K-1)+1:
      a. allSets = Create a list of sets with all the possible combinations for numbers from 1 to  N
      b. allCouplings = Create all possible sets of cardinality 2 for numbers from 1 to N
      c. For each set in allSets (step 3.a), create all possible sets of cardinality 2 from the number available
         in the set – subSetCouplings
             Count=0
            i. For each subSetCoupling in subSetCouplings
                  1. If subSetCoupling present in allCouplings
                  2. Count++
           ii. If count==subsetCouplings.length():
                  1. allSets.remove(set)
```

4. Algorithm for set creation if N cannot be expressed as *K(K-1)+1*

i. If N cannot be expressed as K(K-1)+1, then we need to find another number M such that M is slightly larger than N and can be expressed as K(K-1)+1
ii. While calculating the number M we can calculate K as in step 2 of #3 section
iii. For this we increment M (N=M) as M += 1 till we find a K such that M = K(K-1)+1
iv. Once M is found that satisfies the condition, we start following the steps 3 in #3 section
v. Once all the sets following the conditions for set creation are satisfied, we fill perform pruning of the numbers in the sets that are greater than N (original input entered by the user)
vi. For subset in setsNew:

```
For element in subset:
    for el in element:
        If el == newN:
            el = N
        if el > N:
            # perform relevant replacements
            # eg. If N=5, newN=7, el=6 then el is replaced with 4
            El = N-(newN-el)
```

vii. After the replacements have been done, we will delete the redundant lists, i.e., sets with the same values.

viii. We will all make each set free of any duplicates, if any. This may even lead to some sets not having the cardinality same as others.

ix. Further, for more filtering, we will perform steps from 3.a to 3.c. This will lead to the final degenerate sets being created.

5. Addition and deletion of number of nodes in the system

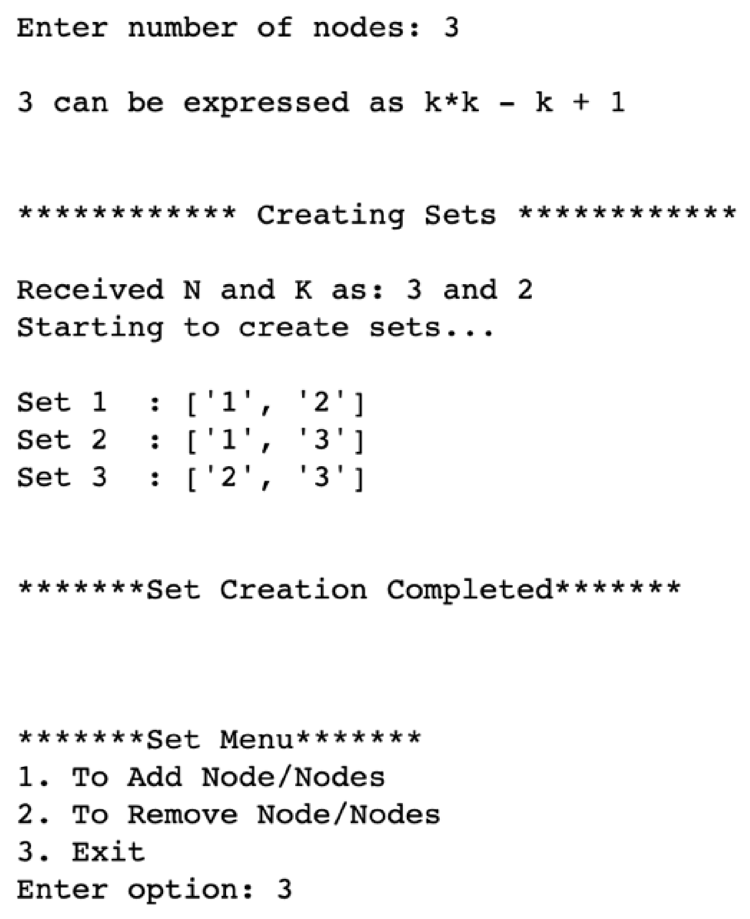
Algorithms #3 and #4 continue till the user wishes to exit the system. The user can add or remove the nodes by entering the number of nodes to be added or deleted in the system. When the user enters the option to add x number of nodes in the system, we recalculate the original N as (N+x)that was entered earlier. And perform steps in algorithms 3 and 4. Similarly, if the user enters the option to delete x number of nodes, we recalculate the original N as (N-x) and perform steps in algorithms 3 and 4.

6. Config file generation

Once the code is executed and the user selects option #3 to exit the system, the config file will be generated with the list of sets and other config parameters for Project #2.

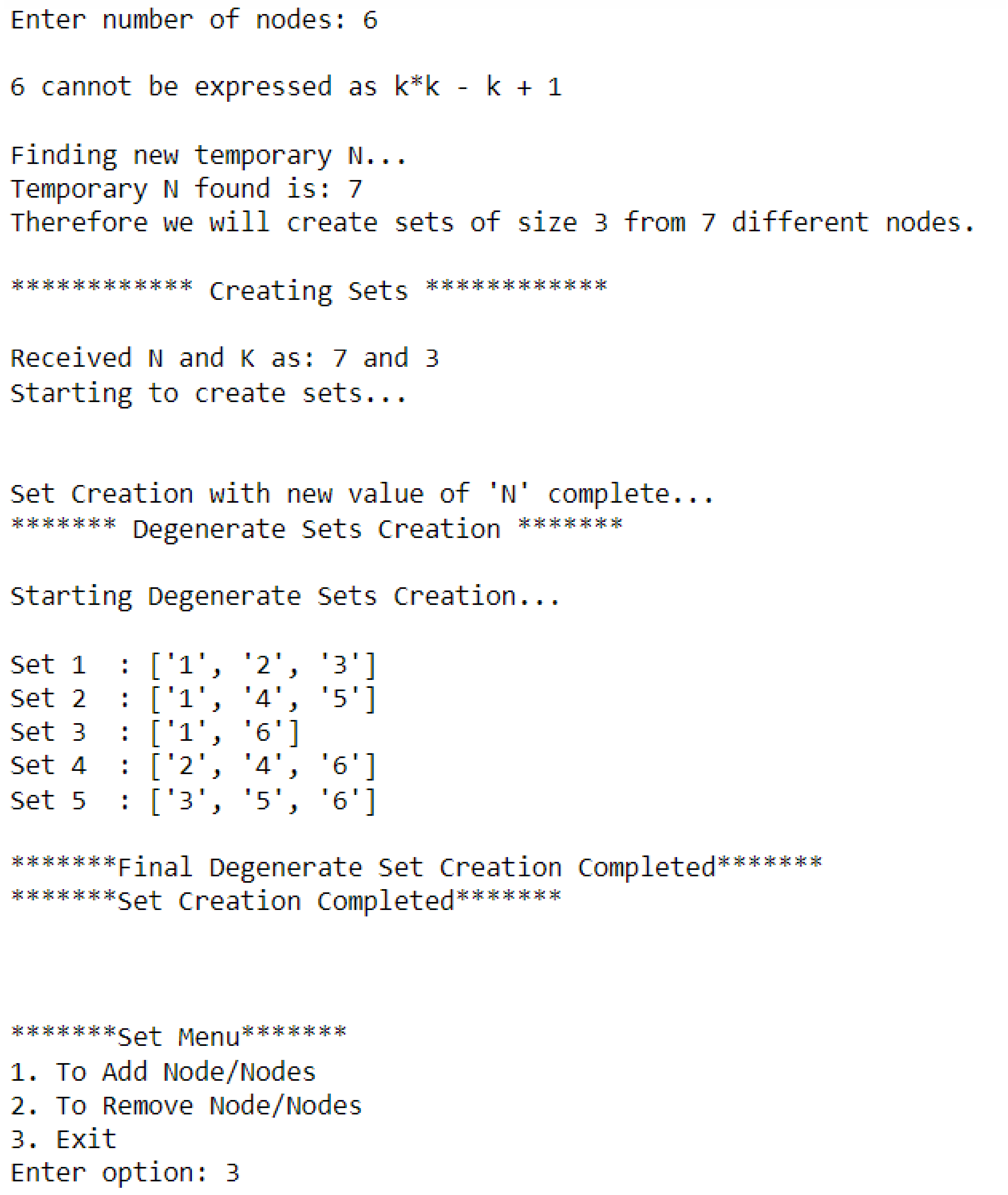
Algorithm implementation output:

1. When N can be expressed as K(K-1)+1



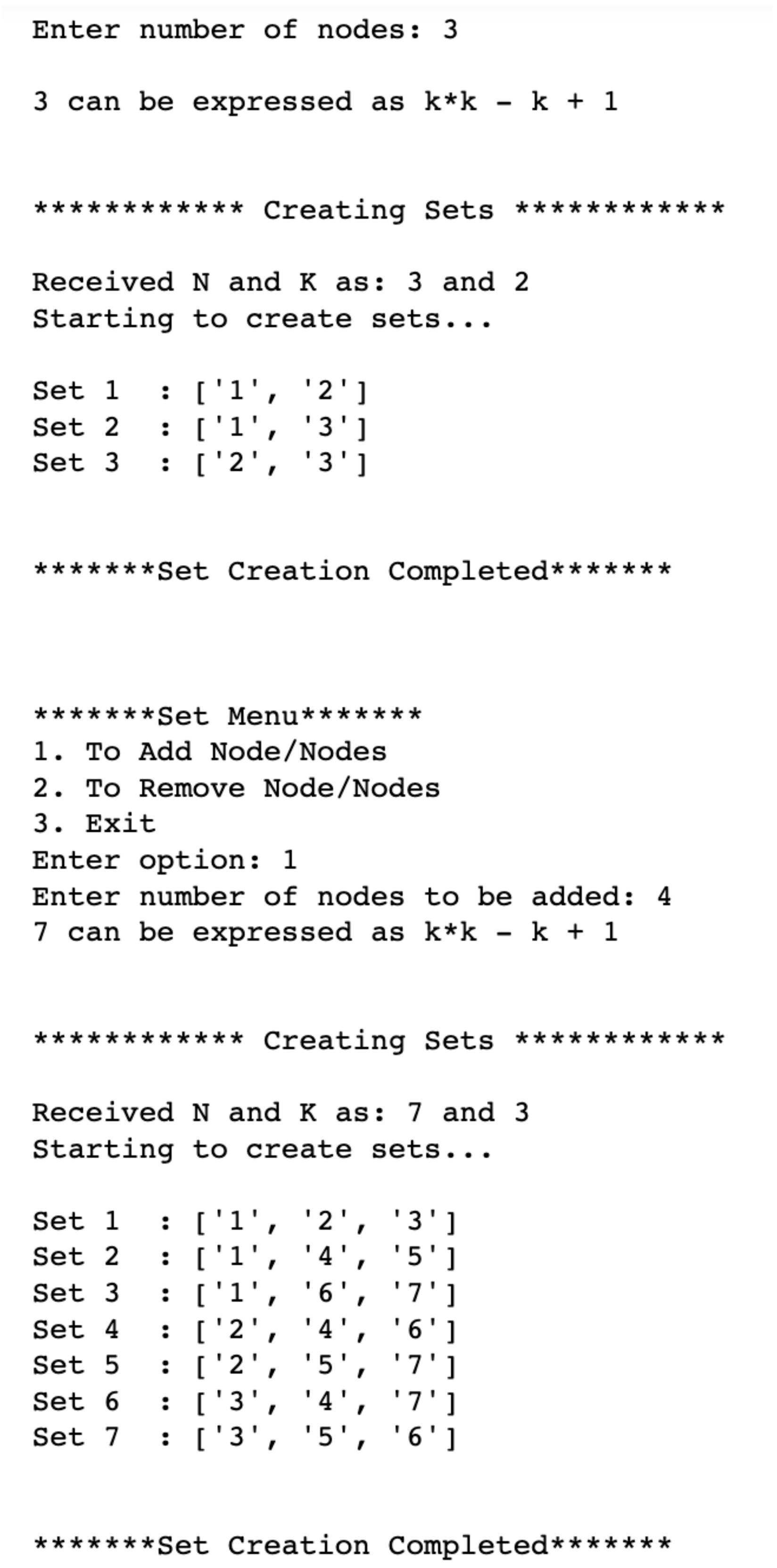
```
Enter number of nodes: 3

3 can be expressed as k*k - k + 1


************ Creating Sets ************

Received N and K as: 3 and 2
Starting to create sets...

Set 1  : ['1', '2']
Set 2  : ['1', '3']
Set 3  : ['2', '3']


*******Set Creation Completed*******



*******Set Menu*******
1. To Add Node/Nodes
2. To Remove Node/Nodes
3. Exit
Enter option: 3
```

2. When N cannot be expressed as K(K-1)+1



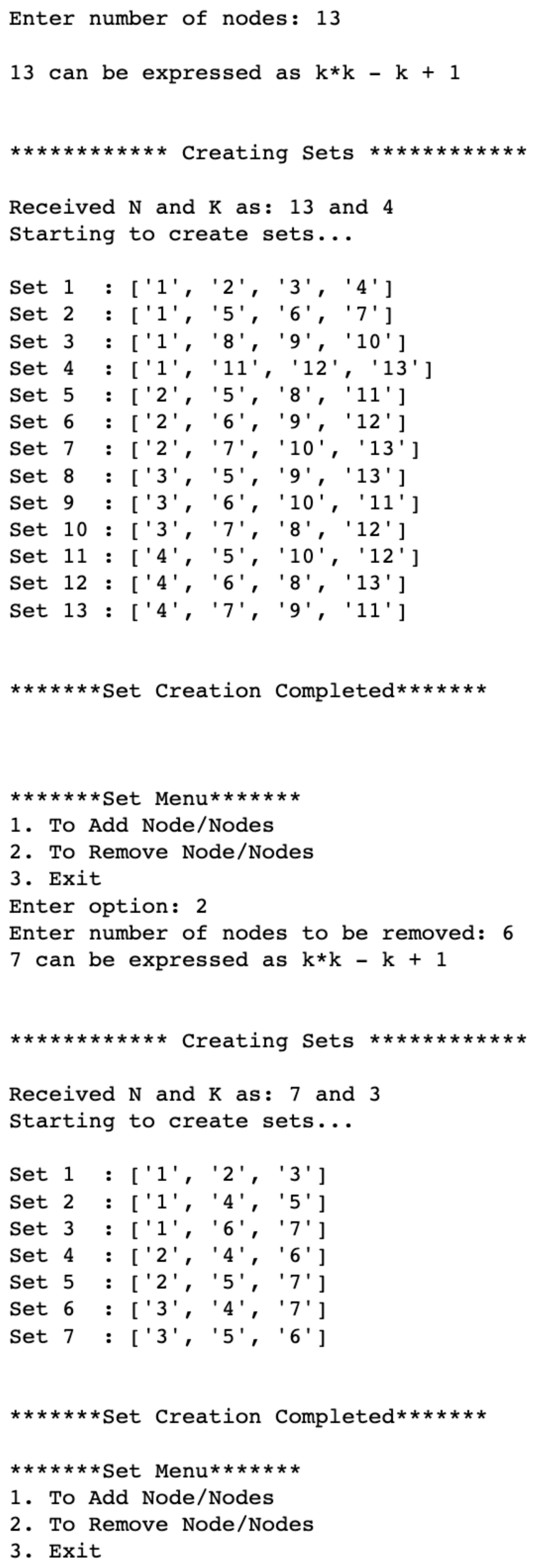
```
Enter number of nodes: 6

6 cannot be expressed as k*k - k + 1

Finding new temporary N...
Temporary N found is: 7
Therefore we will create sets of size 3 from 7 different nodes.

************ Creating Sets ************

Received N and K as: 7 and 3
Starting to create sets...


Set Creation with new value of 'N' complete...
******* Degenerate Sets Creation *******

Starting Degenerate Sets Creation...

Set 1  : ['1', '2', '3']
Set 2  : ['1', '4', '5']
Set 3  : ['1', '6']
Set 4  : ['2', '4', '6']
Set 5  : ['3', '5', '6']

*******Final Degenerate Set Creation Completed*******
*******Set Creation Completed*******



*******Set Menu*******
1. To Add Node/Nodes
2. To Remove Node/Nodes
3. Exit
Enter option: 3
```

3. User wishes to add x number of nodes in the system



```
Enter number of nodes: 3

3 can be expressed as k*k - k + 1


************ Creating Sets ************

Received N and K as: 3 and 2
Starting to create sets...

Set 1  : ['1', '2']
Set 2  : ['1', '3']
Set 3  : ['2', '3']


*******Set Creation Completed*******



*******Set Menu*******
1. To Add Node/Nodes
2. To Remove Node/Nodes
3. Exit
Enter option: 1
Enter number of nodes to be added: 4
7 can be expressed as k*k - k + 1


************ Creating Sets ************

Received N and K as: 7 and 3
Starting to create sets...

Set 1  : ['1', '2', '3']
Set 2  : ['1', '4', '5']
Set 3  : ['1', '6', '7']
Set 4  : ['2', '4', '6']
Set 5  : ['2', '5', '7']
Set 6  : ['3', '4', '7']
Set 7  : ['3', '5', '6']


*******Set Creation Completed*******
```

4. User wishes to delete x number of nodes in the system



```
Enter number of nodes: 13

13 can be expressed as k*k - k + 1


************ Creating Sets ************

Received N and K as: 13 and 4
Starting to create sets...

Set 1  : ['1', '2', '3', '4']
Set 2  : ['1', '5', '6', '7']
Set 3  : ['1', '8', '9', '10']
Set 4  : ['1', '11', '12', '13']
Set 5  : ['2', '5', '8', '11']
Set 6  : ['2', '6', '9', '12']
Set 7  : ['2', '7', '10', '13']
Set 8  : ['3', '5', '9', '13']
Set 9  : ['3', '6', '10', '11']
Set 10 : ['3', '7', '8', '12']
Set 11 : ['4', '5', '10', '12']
Set 12 : ['4', '6', '8', '13']
Set 13 : ['4', '7', '9', '11']


*******Set Creation Completed*******



*******Set Menu*******
1. To Add Node/Nodes
2. To Remove Node/Nodes
3. Exit
Enter option: 2
Enter number of nodes to be removed: 6
7 can be expressed as k*k - k + 1


************ Creating Sets ************

Received N and K as: 7 and 3
Starting to create sets...

Set 1  : ['1', '2', '3']
Set 2  : ['1', '4', '5']
Set 3  : ['1', '6', '7']
Set 4  : ['2', '4', '6']
Set 5  : ['2', '5', '7']
Set 6  : ['3', '4', '7']
Set 7  : ['3', '5', '6']


*******Set Creation Completed*******

*******Set Menu*******
1. To Add Node/Nodes
2. To Remove Node/Nodes
3. Exit
```