

**Smt. Indira Gandhi College of Engineering**

**Ghansoli – Navi Mumbai**

**Computer Engineering Department**

**Academic Year 2022-23 (Even Sem)**

**Student Name:** Pramod Bandgar **Roll No:** 02 **Class:** TE **Sem:** VI

**Course Name:** Artificial Intelligence Lab

**Course Code:** CSL604

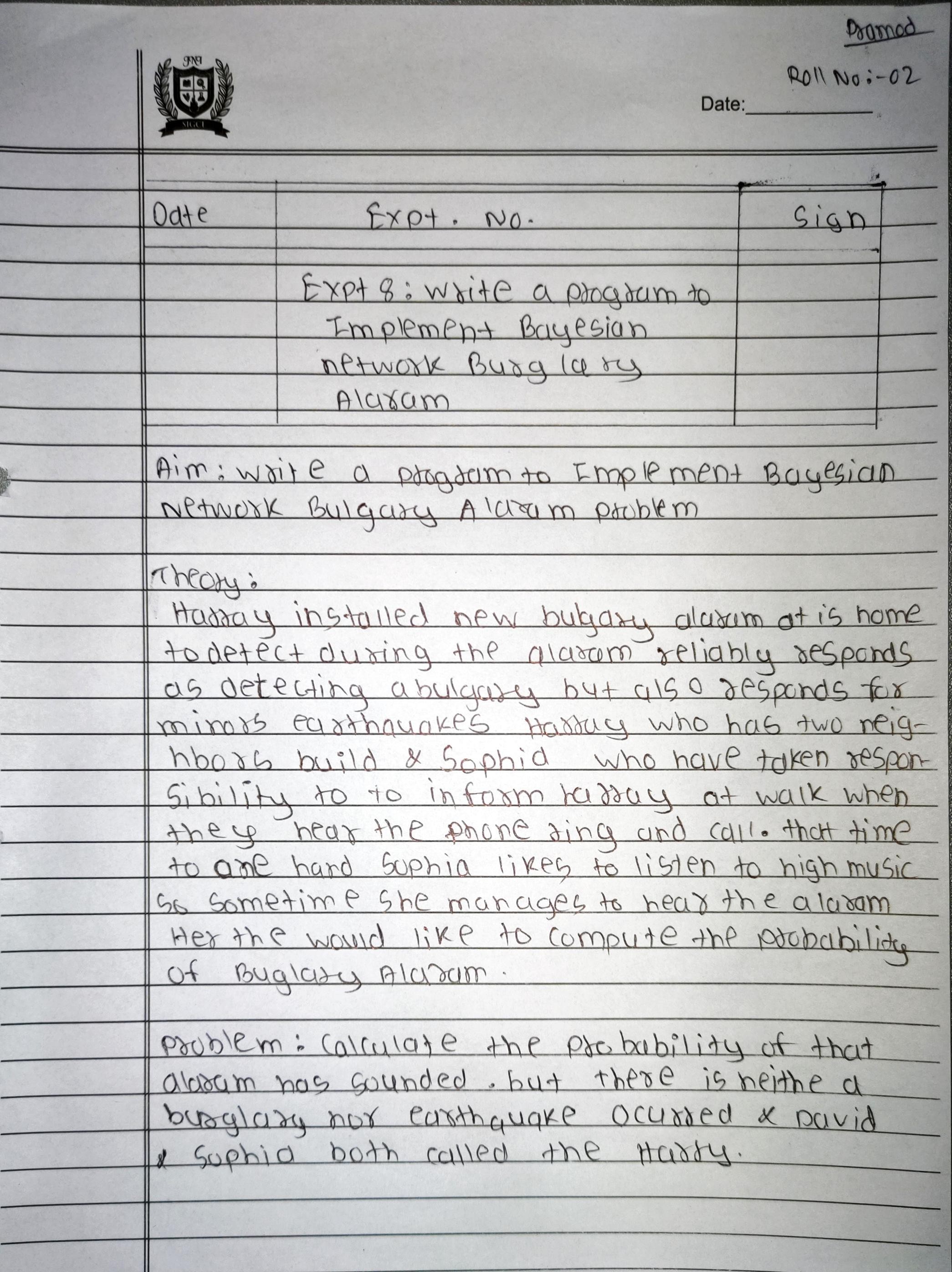
Experiment No. 08

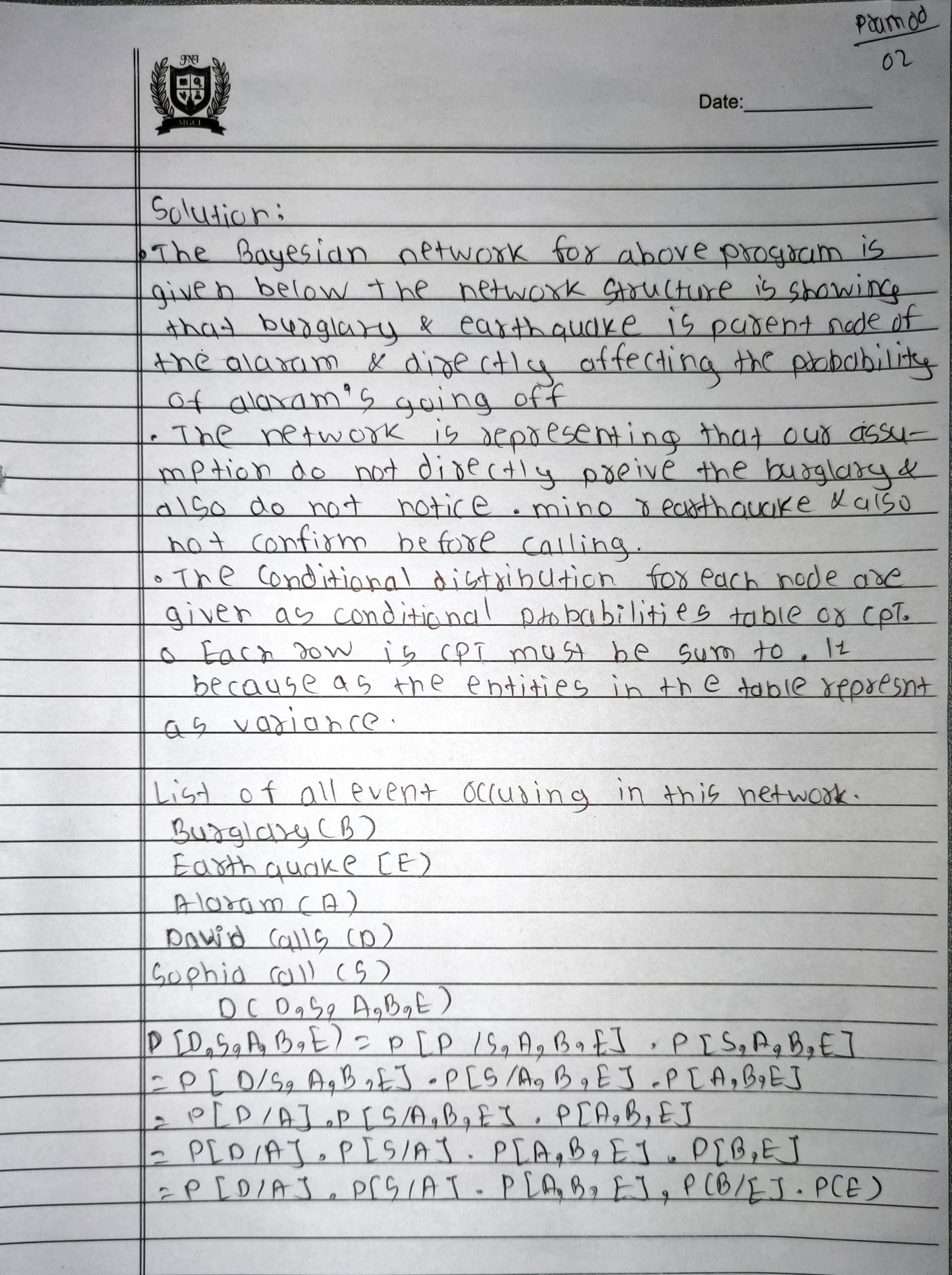
**Experiment Title:** Implementing Bayesian Network: Burglary Alarm Problem

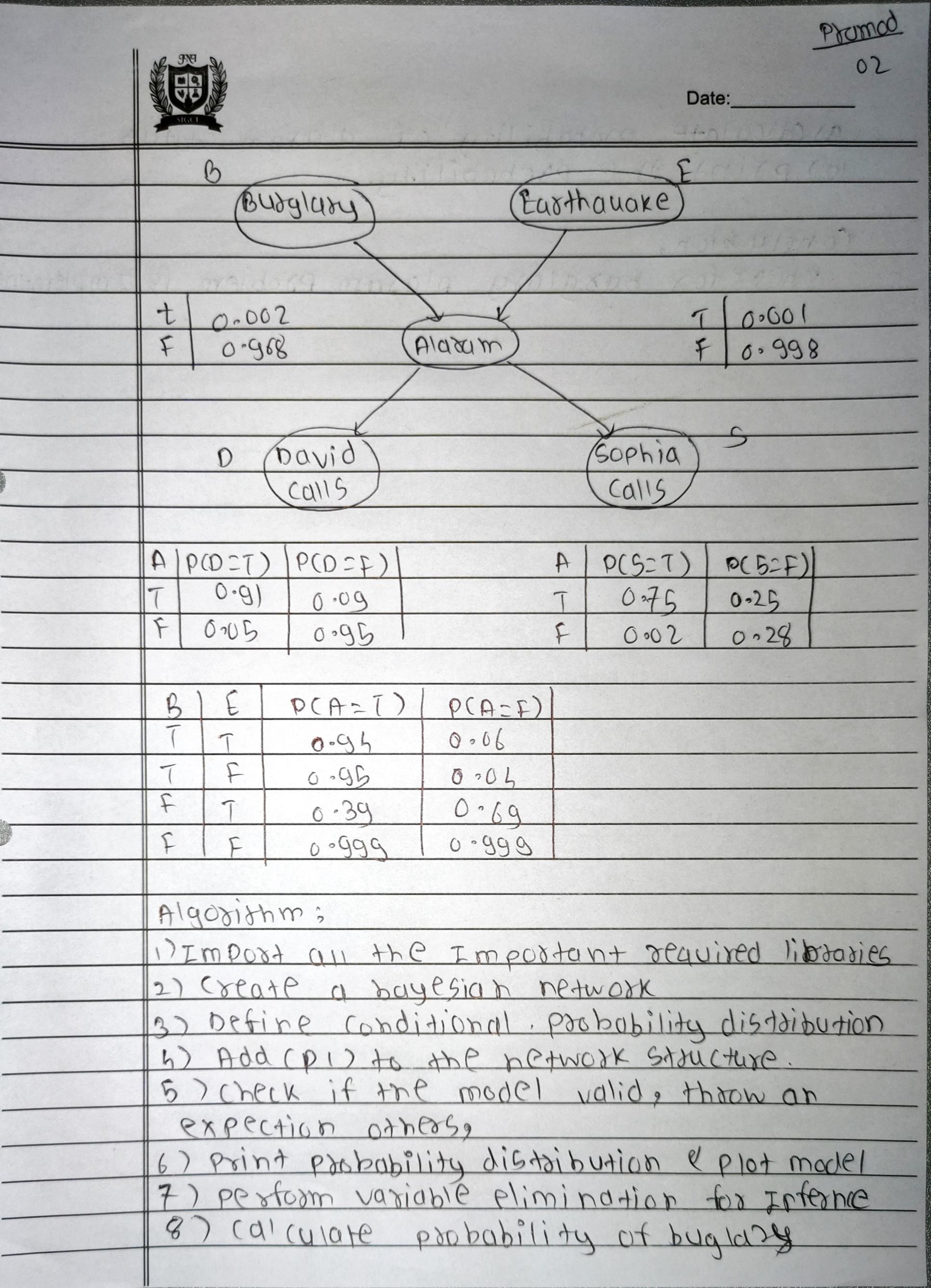
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Date of Performance** | **Date of Submission** | **Marks (10)** | | | | | **Sign / Remark** |
| **A** | **B** | **C** | **D** | **E** |
| **2** | **3** | **2** | **2** | **1** |
|  |  |  |  |  |  |  |  |
| **Total Marks** | | | | |
|  | | | | |

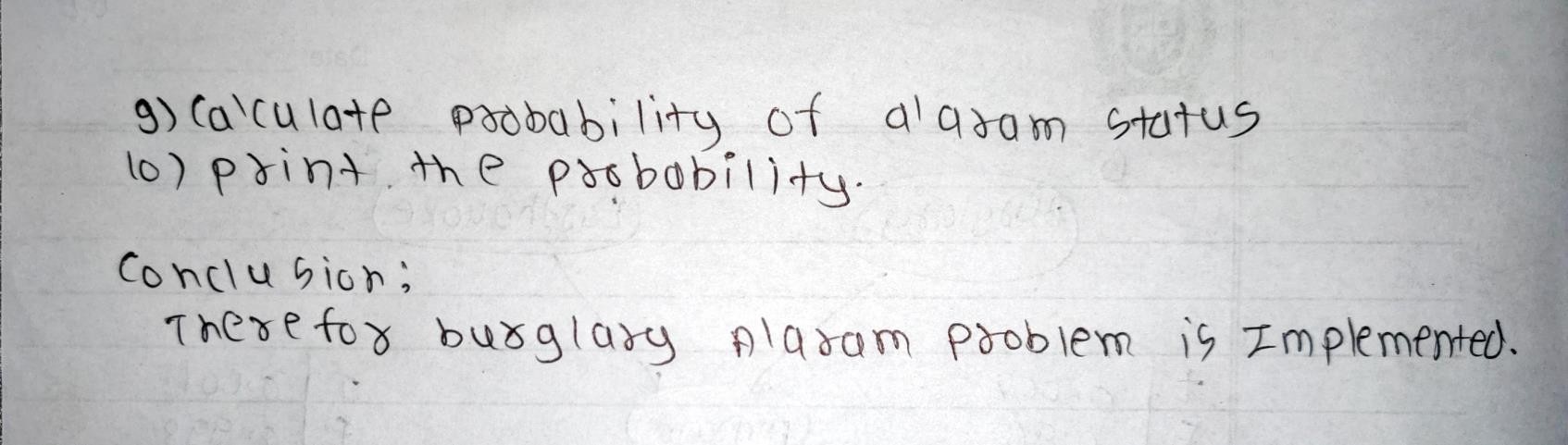
**A:** On Time Submission **B:** Understanding **C:** Analytical Skill

**D:** Critical Thinking **E:** Presentation









# Experiment No. 08

**Code:**

import pgmpy.models import pgmpy.inference

model = pgmpy.models.BayesianNetwork([('Burglary', 'Alarm'), ('Earthquake', 'Alarm'),

('Alarm', 'JohnCalls'),

('Alarm', 'MaryCalls')])

cpd\_burglary = pgmpy.factors.discrete.TabularCPD('Burglary', 2, [[0.001], [0.999]])

cpd\_earthquake = pgmpy.factors.discrete.TabularCPD('Earthquake', 2, [[0.002], [0.998]])

cpd\_alarm = pgmpy.factors.discrete.TabularCPD('Alarm', 2, [[0.95, 0.94, 0.29, 0.001],

[0.05, 0.06, 0.71, 0.999]],

evidence=['Burglary', 'Earthquake'], evidence\_card=[2, 2])

cpd\_john = pgmpy.factors.discrete.TabularCPD('JohnCalls', 2, [[0.90, 0.05],

[0.10, 0.95]],

evidence=['Alarm'], evidence\_card=[2])

cpd\_mary = pgmpy.factors.discrete.TabularCPD('MaryCalls', 2, [[0.70, 0.01],

[0.30, 0.99]],

evidence=['Alarm'], evidence\_card=[2])

model.add\_cpds(cpd\_burglary, cpd\_earthquake, cpd\_alarm, cpd\_john, cpd\_mary)

model.check\_model() print('\n')

infer = pgmpy.inference.VariableElimination(model)

# Calculate the probability of a burglary if only John calls evidence = {'JohnCalls': 1, 'MaryCalls': 0}

posterior\_probability = infer.query(['Burglary'], evidence=evidence) # Print posterior probability

print(f'Posterior probability of Burglary if JohnCalls({bool(evidence["JohnCalls"])}) and MaryCalls({bool(evidence["MaryCalls"])})')

print(posterior\_probability) print()

# Calculate the probability of a burglary if only Mary calls evidence = {'JohnCalls': 0, 'MaryCalls': 1}

posterior\_probability = infer.query(['Burglary'], evidence=evidence) # Print posterior probability

print(f'Posterior probability of Burglary if JohnCalls({bool(evidence["JohnCalls"])}) and MaryCalls({bool(evidence["MaryCalls"])})')

print(posterior\_probability) print()

# Calculate the probability of a burglary if John and Mary calls evidence = {'JohnCalls': 1, 'MaryCalls': 1}

posterior\_probability = infer.query(['Burglary'], evidence=evidence) # Print posterior probability

print(f'Posterior probability of Burglary if JohnCalls({bool(evidence["JohnCalls"])}) and MaryCalls({bool(evidence["MaryCalls"])})')

print(posterior\_probability)

print()

# Calculate the probability of a Burglary if no one calls evidence = {'JohnCalls': 0, 'MaryCalls': 0}

posterior\_probability = infer.query(['Burglary'], evidence=evidence) # Print posterior probability

print(f'Posterior probability of Burglary if JohnCalls({bool(evidence["JohnCalls"])}) and MaryCalls({bool(evidence["MaryCalls"])})')

print(posterior\_probability) print()

# Output:

