Assignment 2

```
import random
import string

def generate_data(email):
  """
  This function generates network data (nodes and edges) based on a student's email ID.

  Args:
      email: The student's email ID (used as a seed for random number generation).

  Returns:
      A dictionary containing two keys:
          "Nodes": A list of unique node names (uppercase two characters).
          "Edges": A list of dictionaries representing edges with "from", "to", and "cost" keys.
  """
  # Set a random seed based on the email hash.
  random.seed(hash(email))

  # Generate number of nodes (between 4 and 10).
  num_nodes = random.randint(4, 10)

  # Generate unique node names.
  nodes = [''.join(random.choices(string.ascii_uppercase, k=2)) for _ in range(num_nodes)]

  # Probability of generating same data for different emails.
  same_data_probability = 0.2

  # Check if random value falls within the probability threshold for same data.
  generate_same_data = random.random() < same_data_probability

  # If generating same data, use a predefined set (modify as needed).
  if generate_same_data:
    return {
      "Nodes": ["AA", "BC", "CE", "DM"],
      "Edges": [
        {"from": "AA", "to": "BC", "cost": 1},
        # ... (rest of the edges from the sample output)
      ]
    }

  # Generate random edges (avoiding duplicates).
  edges = []
  while len(edges) < num_nodes * (num_nodes - 1) // 2:
    from_node = random.choice(nodes)
    to_node = random.choice(nodes)
    cost = round(random.uniform(1, 5), 1)  # Cost between 1.0 and 5.0

    # Ensure no duplicate edges (same from-to pair).
    if not any(edge["from"] == from_node and edge["to"] == to_node for edge in edges):
      edges.append({"from": from_node, "to": to_node, "cost": cost})

  return {"Nodes": nodes, "Edges": edges}

# Example usage
email = "abc@email.com"
data = generate_data(email)

print("Nodes:", data["Nodes"])
print("Edges:", data["Edges"])
```