Experiment No 4 **Roll no :**

Aim

Theory :

Flajolet–Martin Algorithm (FM Algorithm)

1. Introduction The Flajolet–Martin algorithm

It is a probabilistic algorithm used in data stream processing to estimate the number of distinct elements (cardinality) in a large dataset using very little memory. It is widely applied in Big Data because storing and processing all elements in memory is often impractical. It was introduced by Philippe Flajolet and G. Nigel Martin in 1985.

2. Core Idea

- Instead of storing all elements, hash each element into a bitstring.
- Use the position of the rightmost 1-bit in the binary representation to estimate cardinality.
- The probability of getting a certain number of trailing zeros in a random hash follows a predictable pattern, which can be used to estimate the count.

3. Advantages

- Low Memory Usage – Works with fixed, small memory space.
- Scalable – Handles huge data streams efficiently.
- Fast – Single-pass algorithm (O(n) time, O(1) space).
- Probabilistic Accuracy – Acceptable error margin for Big Data analytics.
- Parallelizable – Can be run on distributed systems like Hadoop or Spark

4. Disadvantages

- Approximation Only – Produces estimates, not exact counts.
- Hash Function Dependency – Accuracy depends on the quality of the hash function.
- Collision Risk – Poor hash functions may produce skewed results.
- Not Good for Small Data – Overhead may be unnecessary for small datasets.

5. Applications of Flajolet–Martin Algorithm

- **Search Engines –** Estimate the number of unique search queries daily without storing complete query logs (Google, Bing, Baidu).
- **Network Traffic Analysis –** Count distinct IP addresses or devices connecting to a network for analytics and intrusion detection.
- **E-Commerce Analytics** – Track unique shoppers, product viewers, or cart creators during sales events.

• **Social Media Analytics** – Estimate unique active users posting, liking, or sharing a topic in real time.

• **Database Query Optimization** – Provide quick cardinality estimates for query planners (PostgreSQL, Oracle).

• **IoT Monitoring** – Track unique sensor IDs transmitting data from smart devices in large-scale IoT environments.

• **Financial Systems** – Estimate unique accounts involved in daily transactions to detect anomalies or fraud.

6. Examples of Flajolet–Martin Algorithm

• **Example 1 –** Twitter Hashtag Tracking Twitter uses FM to estimate how many unique users tweeted #WorldCup2025 in the last 24 hours without storing each tweet.

• **Example 2** – ISP Network Analysis An Internet Service Provider estimates the number of distinct IPs visiting their network monthly using FM instead of maintaining massive logs.

• **Example 3** – Amazon Flash Sale Analytics Amazon monitors unique customers viewing a product during Prime Day using FM for real-time dashboards.

Source Code

```
: from prettytable import PrettyTable  # For nice table output

def hash_func(val):
    return (3 * val + 1) % 5

def count_trailing_zero(binstring):
    count = 0
    for char in reversed(binstring):
        if char == '0':
            count += 1
        else:
            break
    return count

def binconver(hash_val):
    binary_repr = format(hash_val, '04b')
    return binary_repr, count_trailing_zero(binary_repr)

# Input data
input_list = [1, 3, 2, 1, 2, 3, 4, 3, 1]

# Table setup
table = PrettyTable(["Element", "Hash Value", "Binary", "Trailing Zeros"])
```

```
max_trailing_zeros = 0

for val in input_list:
    hv = hash_func(val)
    binary_repr, tz = binconver(hv)
    max_trailing_zeros = max(max_trailing_zeros, tz)
    table.add_row([val, hv, binary_repr, tz])

estimate = 2 ** max_trailing_zeros

# Print results
print(table)
print("Max trailing zeros:", max_trailing_zeros)
print("Estimated distinct elements (FM):", estimate)
```

Output :

```
PS D:\PrasadB1-713> python -u "d:\PrasadB1-713\flajolet.py"
+---------+------------+--------+-----------------+
| Element | Hash Value | Binary | Trailing Zeros |
+---------+------------+--------+-----------------+
|    1    |     4      |  0100  |        2        |
|    3    |     0      |  0000  |        4        |
|    2    |     2      |  0010  |        1        |
|    1    |     4      |  0100  |        2        |
|    2    |     2      |  0010  |        1        |
|    3    |     0      |  0000  |        4        |
|    4    |     3      |  0011  |        0        |
|    3    |     0      |  0000  |        4        |
|    1    |     4      |  0100  |        2        |
+---------+------------+--------+-----------------+
Max trailing zeros: 4
Estimated distinct elements (FM): 16
PS D:\PrasadB1-713>
```

Conclusion : We have studied and successfully implemented Flajolet Martin algorithm.