CT Week 4 : Nested iterations, Birthday paradox, Binning

Week 4 Tutorial 4.1

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.PartOfSpeech == "Noun" OR X.PartOfSpeech == "Verb") {
                count = count + 1
        }
        Move X to Pile 2
}
```

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.PartOfSpeech == "Noun") {
                count = count + 1
        }
        if (X.PartOfSpeech == "Verb") {
                count = count + 1
        }
        Move X to Pile 2
}
```

Both are equivalent

ONLINE DEGR

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.Mathematics > 90 OR X.Physics > 90) {
                count = count + 1
        }
        Move X to Pile 2
}
```

If Mathematics is 93 and Physics is 96, count is incremented once.

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.Mathematics > 90) {
                count = count + 1
        }
        if (X.Physics > 90) {
                count = count + 1
        }
        Move X to Pile 2
}
```

If Mathematics is 93 and Physics is 96, count is incremented twice.

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.PartOfSpeech == "Noun" OR X.PartOfSpeech == "Verb") {
                count = count + 1
        }
        Move X to Pile 2
}
```

The conditions cannot both be true simulateneously. They are mutually exclusive.

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.PartOfSpeech == "Noun") {
                count = count + 1
        }
        if (X.PartOfSpeech == "Verb") {
                count = count + 1
        }
        Move X to Pile 2
}
```

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.Mathematics > 90 OR X.Physics > 90) {
                count = count + 1
        }
        Move X to Pile 2
}
```

The conditions can both be true simulateneously. They are not mutually exclusive.

```
count = 0
while (Pile 1 has more cards) {
        Read the top card X in Pile 1
        if (X.Mathematics > 90) {
                count = count + 1
        }
        if (X.Physics > 90) {
                count = count + 1
        }
        Move X to Pile 2
}
```

```
Ncount = 0
Vcount = 0
while (Pile 1 has more cards) {
    Read the top card X in Pile 1
    if (X.PartOfSpeech == "Noun") {
        Ncount = Ncount + 1
    }
    if (X.PartOfSpeech == "Verb") {
        Vcount = Vcount + 1
    }
    Move X to Pile 2
}
```

```
Ncount = 0
Vcount = 0
while (Pile 1 has more cards) {
    Read the top card X in Pile 1
    if (X.PartOfSpeech == "Noun" OR X.PartOfSpeech == "Verb") {
        Ncount = Ncount + 1
        Vcount = Vcount + 1
    }
    Move X to Pile 2
}
```

Both are different

Lecture 1 : Concept of nested iterations using the birthday paradox (Naive approach)

L. 4.1 — Concept of Nested Iterations Using the birthday Paradox (Naive Approach)

- Outer Iteration → ☐ ☐ and Inner Iteration)

Birthday Paradox

(Repr people — 25 Then there is a sub chance of 2 people have a same month and Day of birth. —)

↳ 25 student → high probability they have same birthday.

\# Pick 1 card

\# Check birthday and then Compare it to all other Cards.

Pile 1 Pile 2

(1) (n-1)

→ Improvement — Separate DOB in month-wise check.

QN : 1,3

Lecture 2 : Concept of nested iterations using the birthday paradox (Using binning)

- Separates Card into months (bin)

* Nested Iteration

Iteration {
  Interation {
  }
}

- Total ← Jan ... Dec

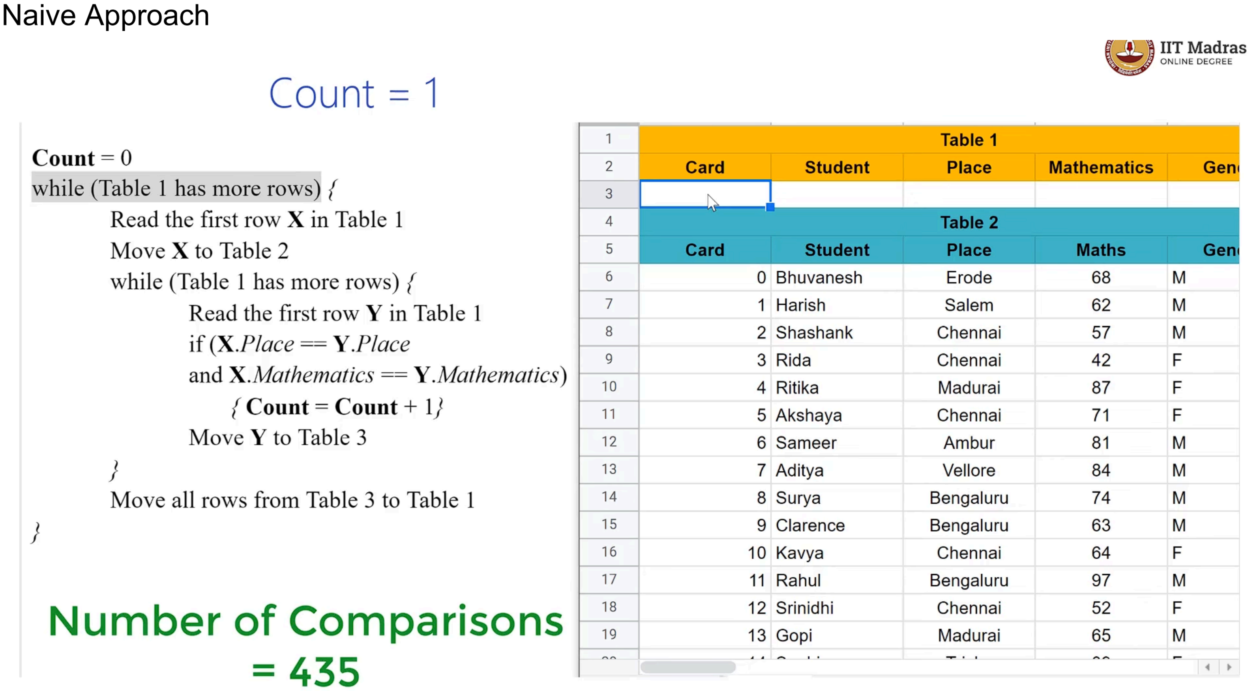
$\frac{n(n-1)}{2}$

Then use 1st naive approach on this after using binning.

QN : 1,3

Week 4 Tutorial 4.2

Naive Approach



	Card	Student	Place	Mathematics	Gen
1	Table 1				
2	Card	Student	Place	Mathematics	Gen
3					
4	Table 2				
5	Card	Student	Place	Maths	Gen
6	0	Bhuvanesh	Erode	68	M
7	1	Harish	Salem	62	M
8	2	Shashank	Chennai	57	M
9	3	Rida	Chennai	42	F
10	4	Ritika	Madurai	87	F
11	5	Akshaya	Chennai	71	F
12	6	Sameer	Ambur	81	M
13	7	Aditya	Vellore	84	M
14	8	Surya	Bengaluru	74	M
15	9	Clarence	Bengaluru	63	M
16	10	Kavya	Chennai	64	F
17	11	Rahul	Bengaluru	97	M
18	12	Srinidhi	Chennai	52	F
19	13	Gopi	Madurai	65	M

Binning Approach

```
while (Table 1 has more rows) {
        Read the first row X in Table 1
        if (X.Place == "Ambur"){
                Move X to Table-Ambur
           }
        if (X.Place == "Bengaluru"){
                Move X to Table-Bengaluru
           }
        if (X.Place == "Chennai"){
                Move X to Table-Chennai
           }
        if (X.Place == "Erode"){
                Move X to Table-Erode
           }
        if (X.Place == "Madurai"){
                Move X to Table-Madurai
           }
        if (X.Place == "Salem"){
                Move X to Table-Salem
           }
```

```
                Move X to Table-Madurai
           }
        if (X.Place == "Salem"){
                Move X to Table-Salem
           }
        if (X.Place == "Trichy"){
                Move X to Table-Trichy
           }
        if (X.Place == "Vellore"){
                Move X to Table-Vellore
           }
        if (X.Place == "Theni"){
                Move X to Table-Theni
           }
        if (X.Place == "Nagercoil"){
                Move X to Table-Nagercoil
           }
}
```

Table 1

Card	Student	Place	Mathematics	Gender
0	Bhuvanesh	Erode	68	M
1	Harish	Salem	62	M
2	Shashank	Chennai	57	M
3	Rida	Chennai	42	F
4	Ritika	Madurai	87	F
5	Akshaya	Chennai	71	F
6	Sameer	Ambur	81	M
7	Aditya	Vellore	84	M
8	Surya	Bengaluru	74	M
9	Clarence	Bengaluru	63	M
10	Kavya	Chennai	64	F
11	Rahul	Bengaluru	97	M
12	Srinidhi	Chennai	52	F
13	Gopi	Madurai	65	M
14	Sophia	Trichy	89	F
15	Goutami	Theni	76	F
16	Tauseef	Trichy	87	M
17	Arshad	Chennai	62	M
18	Abirami	Erode	72	F
19	Vetrivel	Trichy	56	M

Table-Ambur

Card	Student	Place	Maths	Gender
6	Sameer	Ambur	81	M

Table-Bengaluru

Card	Student	Place	Maths	Gender
8	Surya	Bengaluru	74	M
9	Clarence	Bengaluru	63	M
11	Rahul	Bengaluru	97	M
21	Monika	Bengaluru	78	F
23	Deepika	Bengaluru	97	F

Table-Chennai

Card	Student	Place	Maths	Gender
2	Shashank	Chennai	57	M
3	Rida	Chennai	42	F
5	Akshaya	Chennai	71	F
10	Kavya	Chennai	64	F
12	Srinidhi	Chennai	52	F

```
Count = 0
while (Table-City has more rows) {
    Read the first row X in Table-City
    Move X to Table 2
    while (Table-City has more rows) {
        Read the first row Y in Table-City
        if (X.Mathematics == Y.Mathematics)
            { Count = Count + 1}
        Move Y to Table 3
    }
    Move all rows from Table 3 to Table-City
}
```

	Card	Student	Place	Maths	Gender
3					
4	**Table-Ambur**				
5	**Card**	**Student**	**Place**	**Maths**	**Gender**
6	6	Sameer	Ambur	81	M
7					
8	**Table-Bengaluru**				
9	**Card**	**Student**	**Place**	**Maths**	**Gender**
10	8	Surya	Bengaluru	74	M
11	9	Clarence	Bengaluru	63	M
12	11	Rahul	Bengaluru	97	M
13	21	Monika	Bengaluru	78	F
14	23	Deepika	Bengaluru	97	F
15					
16	**Table-Chennai**				
17	**Card**	**Student**	**Place**	**Maths**	**Gender**
18	2	Shashank	Chennai	57	M
19	3	Rida	Chennai	42	F
20	5	Akshaya	Chennai	71	F
21	10	Kavya	Chennai	64	F
22	12	Srinidhi	Chennai	52	F
23	17	Arshad	Chennai	62	M
24	25	Geeta	Chennai	87	F

IIT Madras
ONLINE DEGREE

Card	Student	Gender	D.O.B	Place	Math	Physics	Chem	Total	Number of Comparisons
6	Sameer	M	23 Mar	Ambur	81	82	87	250	0
8	Surya	M	28 Feb	Bengaluru	74	64	51	189	10
9	Clarence	M	6 Dec	Bengaluru	63	88	73	224	
11	Rahul	M	30 Apr	Bengaluru	97	92	92	281	
21	Monika	F	15 Mar	Bengaluru	78	69	74	221	
23	Deepika	F	13 May	Bengaluru	97	91	88	276	
2	Shashank	M	4 Jan	Chennai	57	54	77	188	28
[illegible]	[illegible]	[illegible]	5 May	Chennai	42	53	78	173	
[illegible]	[illegible]	[illegible]	8 Feb	Chennai	71	92	84	247	
[illegible]	[illegible]	[illegible]	2 Jan	Chennai	64	72	68	204	
[illegible]	[illegible]	[illegible]	4 Jan	Chennai	52	64	71	187	
[illegible]	[illegible]	[illegible]	4 Dec	Chennai	62	81	67	210	
25	Geeta	F	16 May	Chennai	87	75	92	254	
26	JK	M	22 Jul	Chennai	74	71	82	227	

$$n \times (n-1)/2$$

n is the number of rows in that table.

Card	Student	Gender	D.O.B	Place	Math	Physics	Chem	Total	Number of Comparisons
27	Jagan	M	4 Mar	Madurai	81	76	52	209	
28	Nisha	F	10 Sep	Madurai	74	83	83	240	
22	Priya	F	17 Jul	Nagercoil	62	62	57	181	0
1	Harish	M	3 Jun	Salem	62	45	91	198	0
15	Goutami	F	22 Sep	Theni	76	58	90	224	0
14	Sophia	F	23 July	Trichy	89	62	93	244	3
16	Tauseef	M	30 Dec	Trichy	87	86	43	216	
19	Vetrivel	M	30 Aug	Trichy	56	78	62	196	
7	Aditya	M	15 Mar	Vellore	84	92	76	252	3
20	Kalyan	M	17 Sep	Vellore	93	68	91	252	
29	Naveen	M	13 Oct	Vellore	72	66	81	219	
									55

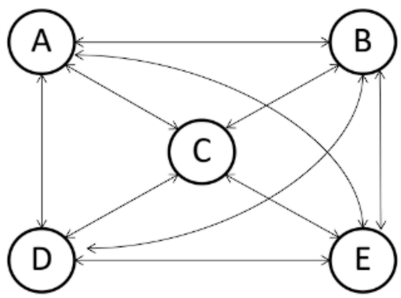
Lecture 3 : Importance of binning to reduce number of comparisons in nested iterations

Reducing number of comparisons

Reducing comparisons: what we observed

- Some computations seem to require comparisons of each card with all the other cards in the pile
  - for example, choosing a study partner for each student
  - the number of comparisons required can be very large
- We observed that if we can organise the cards into bins based on some heuristic:
  - then we only need to compare cards within one bin
  - this seems to significantly reduce the number of comparisons required
- Is there a formal way of determining the reduction in comparisons?
  - Calculate the number of comparisons without binning
  - Calculate the number of comparisons with binning
  - Use these calculations to determine the reduction factor

Comparing each element with all other elements



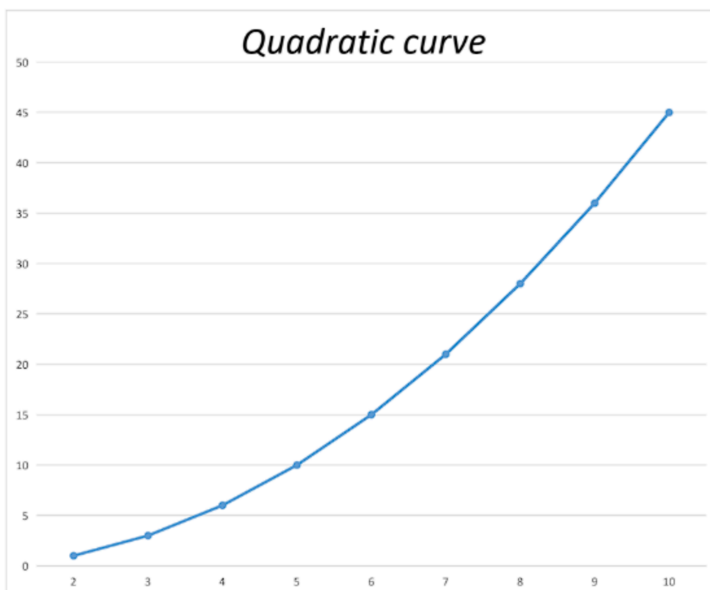
For 5 elements A, B, C, D, E:
The comparisons required are:
A with B, A with C, A with D, A with E (4)
B with C, B with D, B with E (3)
C with D, C with E (2)
D with E (1)
Number of comparisons: 4 + 3 + 2 + 1 = 10

- For N objects, the number of comparisons required will be:
  - (N - 1) + (N - 2) + ... + 1
  - which is = $\frac{N \times (N - 1)}{2}$
- This is the same as the number of ways of choosing 2 objects from N objects:
  - ${}^{N}C_{2} = \frac{N \times (N - 1)}{2}$
- From first principles:
  - Total number of pairs is N × N
  - From this reduce self comparisons (e.g. A with A). So number is reduced to: N × N - N
  - which can be written as N × (N - 1)
  - Comparing A with B is the same as comparing B with A, so we are double counting this comparison
  - So, reduce the count by half = $\frac{N \times (N - 1)}{2}$

Number of comparisons can be written as: ½ × N × (N - 1)

The number of comparisons grows really fast

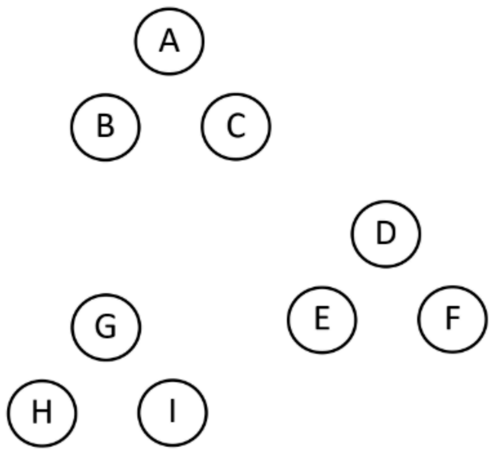
N	$\frac{N \times (N - 1)}{2}$
2	1
3	3
4	6
5	10
6	15
7	21
8	28
9	36
10	45
100	49,500
1000	4,99,500



How do we reduce the number of comparisons?

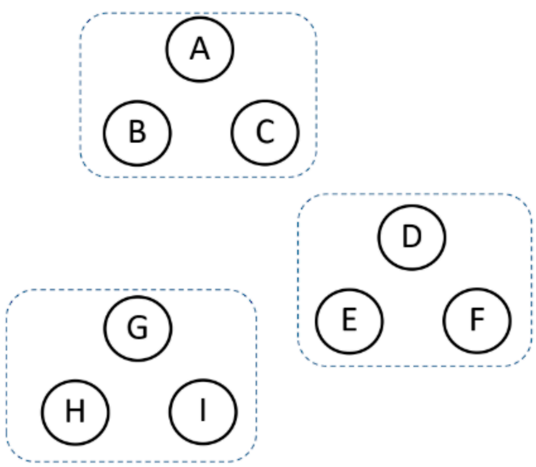
Use binning

Key idea: Use binning



- For 9 objects A,B,C,D,E,F,G,H,I:
  - The number of comparisons is ½ × 9 × (9 - 1) = ½ × 9 × 8 = 9 × 4 = 36

Key idea: Use binning



- For 9 objects A,B,C,D,E,F,G,H,I:
  - The number of comparisons is ½ × 9 × (9 - 1) = ½ × 9 × 8 = 9 × 4 = 36
- If the objects can be binned into 3 bins of 3 each:
  - The number of comparisons per bin is: ½ × 3 × (3 -1) = ½ × 3 × 2 = 3
  - Total number of comparisons for all 3 bins is: 3 × 3 = 9
- So, the number of comparisons reduces from 36 to 9 !
  - *Reduced by a factor of 4 times.*

Calculation of reduction due to binning

- For N items:
- Number of comparisons without binning is: $\frac{1}{2} \times N \times (N - 1)$

- If we use K bins of equal size, number of items in each bin is: N/K
- Number of comparisons per bin is: $\frac{1}{2} \times N/K \times (N/K - 1)$
- Total number of comparisons is:
  $K \times \frac{1}{2} \times N/K \times (N/K - 1) = \frac{1}{2} \times N \times (N/K - 1)$

- Factor of reduction is: $[\frac{1}{2} \times N \times (N - 1)] / [\frac{1}{2} \times N \times (N/K - 1)]$
  $= (N - 1) / (N/K - 1)$
- For N = 9 and K = 3, this is $(9 - 1) / (3 - 1) = 4$
  - So reduction is by a factor of 4 times.

Summary

- The number of comparisons between all pairs of items grows quadratically, i.e. quite fast
- The formula of number of comparisons for N items is: $\frac{1}{2} \times N \times (N - 1)$

- Sometimes, it is possible to find a heuristic that allows us to put the items into bins and compare only items within the bins
- If there are N items put into K bins each of equal size, then the number of comparisons reduces to: $\frac{1}{2} \times N \times (N/K - 1)$

- The factor of reduction is: $(N - 1) / (N/K - 1)$

QN : 5

Lecture 4 : Concept of binning to avoid the complexity of nested iterations

Bill ⟸ customer check

\# Shopping Data :- Comparing Price

\# Customer buying from many shopp -

\# Loyal customer

↳ Generating More bill from same shop.

↳ Only from same shop.

(1) Pick one customer and Iterate through all over dataset and check does customer have one or more shops bill or not.

(2) Bin each customer. and check does customer have bill from same shop or different shops.

↳ (Neerj 1)

QN - 3,4,5,6

Lecture 5 : Concept of fair teams

L.4.5 — Concept of fair team

Revise

2 student — 1 helps other 1, 1 better in one and 2nd better in 2nd subject.

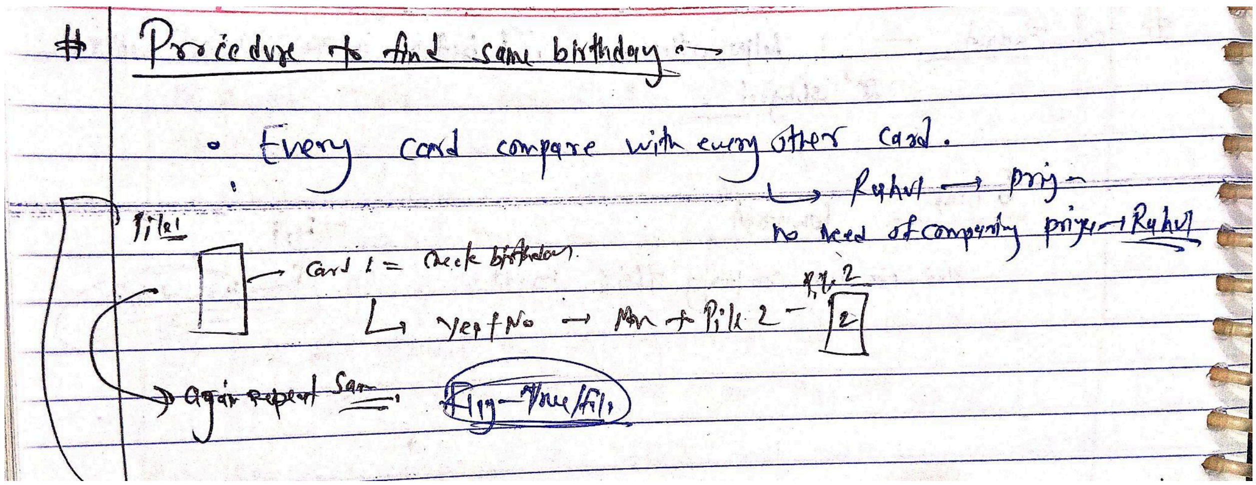
→ 6 bins   lowest — — — — highest

and pair them comparing their mark.

x → No need help

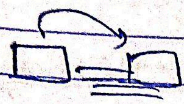
QN : 4

Lecture 6 : Procedure to find same date of birth for different students



Lecture 7 : Procedure to resolve pronoun with its equivalent matching noun

resolve pronoun with its equivalent matching noun

 → Found Pronoun Backtrack → Get noun

- Iterate through Dataset
- Getting a pronoun then Backtrack and check which was last noun.
- That noun → referring Pronoun

Tutorial – 4.3 → count vowel / consonant in one word.

Procedure Count Vowels and Consonants(X)

Week 4 Tutorial 4.3

```
Procedure CountVowelsAndConsonants (X)
        i = 1
        VCount = 0
        CCount = 0
        while (i ≤ X.LetterCount) {
                if (ith letter of X.Word is a vowel) {
                        VCount = VCount + 1
                }
                else {
                        CCount = CCount + 1
                }
                i = i + 1
        }
        return (VCount, CCount)
End CountVowelsAndConsonants
```

Note : Week 4

Nested Iterations

1. Concept of Nested Iterations

Nested iteration means having one loop inside another loop. In this case, the inner loop executes completely for every single iteration of the outer loop.

This is commonly used when working with multi-level data structures or repeated patterns, such as:

- Processing elements in a 2D array or matrix
- Generating combinations or pairs of elements
- Performing operations on grouped data

General structure:

```
foreach i in List1:              # Outer loop
    foreach j in List2:          # Inner loop
        # Code block executed length(List1) × length(List2) times
```

2. Difference Between Normal and Nested Iteration

Feature	Normal Iteration	Nested Iteration
Definition	A single loop executing repeatedly over a range or sequence	A loop inside another loop
Structure	One level of iteration	Two or more levels of iteration
Execution Count	Depends on one variable (e.g., N times)	Depends on product of loop counts (e.g., N × M times)
Use Case	Linear data (1D list, single range)	Multi-dimensional or pairwise operations (2D arrays, comparisons)
Example	Summing numbers in a list	Multiplying two matrices

Summary

- Normal Iteration: One loop for simple repeated tasks.
- Nested Iteration: Loops within loops for multi-level or relational tasks.
- Common Use: When one process must be repeated for every value of another.

Binning Method

Introduction

The binning method is used to smooth data or handle noisy data. In this method:

1. The data is first sorted.
2. The sorted values are then distributed into a number of buckets or bins.

Since binning methods consult the neighborhood of values, they perform local smoothing.

Binning example: When displaying students' marks in a subject, instead of listing each individual score, we group them into bins:
0–20 → Poor
21–40 → Average
41–60 → Good
61–80 → Very Good
81–100 → Excellent

Example: Comparing Each Element with All Other Elements

For N objects, the number of comparisons required will be:

```
(N - 1) + (N - 2) + ... + 1 = N × (N - 1) / 2
```

This is the same as the number of ways of choosing 2 objects from N:

```
NC2 = N × (N - 1) / 2
```

From First Principles

- Total number of pairs = N × N
- Remove self-comparisons (e.g., A with A): → N × N − N = N × (N − 1)
- Comparing A with B is the same as comparing B with A, so we are double-counting. → Divide by 2

Hence, the number of comparisons:

```
= 1/2 × N × (N - 1)
```

Example with 5 Elements: A, B, C, D, E

Element	Comparisons
A	with B, C, D, E → 4
B	with C, D, E → 3
C	with D, E → 2
D	with E → 1

Total comparisons = 4 + 3 + 2 + 1 = 10

Formula:

```
Number of comparisons = 1/2 × N × (N - 1)
```

Calculation of Reduction Due to Binning

Without Binning

For N items:

```
Number of comparisons = 1/2 × N × (N - 1)
```

With Binning

If we use **K bins** of equal size:

- Number of items per bin = N / K
- Number of comparisons per bin = 1/2 × (N / K) × ((N / K) – 1)

Total number of comparisons:

```
K × 1/2 × (N / K) × ((N / K) - 1)
= 1/2 × N × ((N / K) - 1)
```

Factor of Reduction

```
Factor of reduction = (1/2 × N × (N - 1)) / (1/2 × N × ((N / K) - 1))
                    = (N - 1) / ((N / K) - 1)
```

Example: If N = 9 and K = 3:

```
Factor of reduction = (9 - 1) / ((9 / 3) - 1)
                    = 8 / 2
                    = 4
```

Reduction is by a factor of 4 times

Key Idea: Use Binning

For **9 objects A, B, C, D, E, F, G, H, I**:

Without binning:

```
Number of comparisons = 1/2 × 9 × (9 - 1)
                        = 36
```

With binning into 3 bins of 3 each:

```
Comparisons per bin = 1/2 × 3 × (3 - 1) = 3
Total comparisons for all bins = 3 × 3 = 9
```

So, the number of comparisons reduces from 36 to 9! Reduced by a factor of 4 times.

PA : 2,3,4,6,7,9,10
GA : 1,4,6,7,9,10