8/1/21

D D M M Y Y

* Section 9 : Matrices

142. Section Introduction

* Types of Matrices (Special Matrix)

1) Diagonal matrix

2) Lower Triangular Matrix

3) Upper Triangular Matrix

4) Symmetric matrix

5) Tridiagonal Matrix nxn 5x5

6) Triband Matrix

7) Toeplitz matrix

8) Sparse Matrix

143. Diagonal Matrix

M=

	1	2	3	4	5
1	3	0	0	0	0
2	0	7	0	0	0
3	0	0	4	0	0
4	0	0	0	9	0
5	0	0	0	0	6

5×5

All should be zero except diagonal

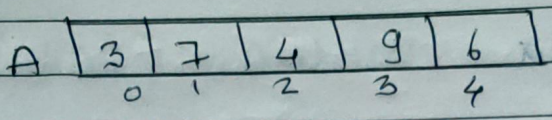
$m[i,j] = 0 \quad \text{if } i \neq j$

A

	0	1	2	3	4
0	3	0	0	0	0
1	0	7	0	0	0
2	0	0	4	0	0
3	0	0	0	9	0
4	0	0	0	0	6

5×5 = 25

25 × 2 = 50 bytes




```
int A[5];
void set (int A[], int i, int j, int x) {
    if (i==j) {
        A[j-1]=x;
    }
}
int get (int A[], int i, int j) {
    if (i==j)
        return A[i-1];
    else
        return (0);
}
```

145 C++ class for Diagonal Matrix

```
class Diagonal {
private:
    int n;
    int *A;
public:
    Diagonal (int n) {
        this->n=n;
        A=new int[n];
    }
    void set (int i, int j, int x);
    int get (int i, int j);
    void Display();
```

```
    ~Diagonal();
};

void Diagonal::Set(int i,int j,int x){
    if(i==j)
        A[i-1]=x;
}
void Diagonal::get(int i,int j){
    if(i==j)
        return A[i-1];
    else
        return 0;
}
void Diagonal::Display(){
    for(i=0;i<n;i++){
        for(j=0;j<=n;j++){
            if(i==j)
                cout<<A[i-1];
            else
                cout<<"0";
        }
        cout<<endl;
    }
}
~Diagonal()
Diagonal::~Diagonal(){
    delete[]A;
}
```

147. Lower Triangular Matrix Row-Major Mapping

$j \rightarrow$

i ↓	1	2	3	4	5
1	a_{11}	0	0	0	0
2	a_{21}	a_{22}	0	0	0
3	a_{31}	a_{32}	a_{33}	0	0
4	a_{41}	a_{42}	a_{43}	a_{44}	0
5	a_{51}	a_{52}	a_{53}	a_{54}	a_{55}

M = 5x5

$m[i,j] = 0$ if $i < j$

$m[i,j] = non\text{-}zero$ if $i >= j$

Non-Zero $= 1+2+3+4+5$

$= 1+2+\ldots+n = \frac{n(n+1)}{2}$

Zero $= n^2 - \frac{n(n+1)}{2}$

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

* Row-Major

A	a_{11}	a_{21}	a_{22}	a_{31}	a_{32}	a_{33}	a_{41}	a_{42}	a_{43}	a_{44}	a_{51}	a_{52}	a_{53}	a_{54}	a_{55}
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	row1			row2			row3				row4				row5

Index (A[4][3]) = [1+2+3] + 2 = 8

Index (A[5][4]) = [1+2+3+4] + 3 = 13

* Formula

$$Index(A[i][j]) = \left[\frac{i(i-1)}{2}\right] + (j-1)$$

D D M M Y Y Y Y

148. Lower Triangular matrix Column-Major Mapping

* Column - Major

A	a_{11}	a_{21}	a_{31}	a_{41}	a_{51}	a_{22}	a_{32}	a_{42}	a_{52}	a_{33}	a_{43}	a_{53}	a_{44}	a_{54}	a_{55}
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	col 1					col 2				col 3			col 4		col 5

Index (A[4][4]) = [5+4+3] + 0 = 12
Index (A[5][4]) = [5+4+3] + 1 = 13
Index (A[5][3]) = [5+4] + 2 = 11

* Formula

$$\text{Index}(A[i][j]) = [n + n-1 + n-2 \ldots + n-(j-2)] + (i-j)$$

$$\text{Index}(A[i][j]) = [n(j-1) - [1+2 \ldots + j-2]] + (i-j)$$

$$\therefore \text{Index}(A[i][j]) = \left[n(j-1) - \frac{(j-2)(j-1)}{2} \right] + (i-j)$$

15. Upper Triangular Matrix Row-Major Mapping

$j \rightarrow$

$i \downarrow$

$$M = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ 0 & a_{22} & a_{23} & a_{24} & a_{25} \\ 0 & 0 & a_{33} & a_{34} & a_{35} \\ 0 & 0 & 0 & a_{44} & a_{45} \\ 0 & 0 & 0 & 0 & a_{55} \end{bmatrix}_{5 \times 5}$$

$m[i,j] = 0 \quad \text{if } (i > j)$

$m[i,j] = \text{non-zero} \quad \text{if } i <= j$

$$\text{non-zero} = 5+4+3+2+1$$
$$= n+n-1+\dots+3+2+1$$
$$= \frac{n(n+1)}{2}$$

$$\text{zero} = \frac{n(n-1)}{2}$$

* Row-Major

A	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{22}	a_{23}	a_{24}	a_{25}	a_{33}	a_{34}	a_{35}	a_{44}	a_{45}	a_{55}
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	row 1					row 2				row 3			row 4		row 5

Index $(A[4][5]) = [5+4+3]+1 = 13$

Index $(A[i][j]) = [n+n-1+\dots+n-(i-2)]+(j-1)$

$$\text{Index}(A[i][j]) = \left[(i-1)n - \frac{(i-2)(i-1)}{2}\right] + (j-i)$$

152. Upper Triangular Matrix Column-Major Mapping

* Column Major

A	a_{11}	a_{12}	a_{22}	a_{13}	a_{23}	a_{33}	a_{14}	a_{24}	a_{34}	a_{44}	a_{15}	a_{25}	a_{35}	a_{45}	a_{55}
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
	col 1		col 2		col 3		col 4				col 5				

$$\text{Index}(A[4][5]) = [1+2+3+4]+3 = 13$$

$$\text{Index}(A[i][j]) = [1+2+\cdots+j-1]+i-1$$

$$\boxed{\text{Index}(A[i][j]) = \left[\frac{j(j-1)}{2}\right] + i - 1}$$

153. Symmetric Matrix

$$M = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 2 & 2 & 2 & 2 & 2 \\ 2 & 2 & 3 & 3 & 3 & 3 \\ 3 & 2 & 3 & 4 & 4 & 4 \\ 4 & 2 & 3 & 4 & 5 & 5 \\ 5 & 2 & 3 & 4 & 5 & 6 \end{array}$$

(i → row, j → column)

Sufficient to store upper triangle or lower triangle

if $M[i,j] = M[j,i]$

⇒ Lower Triangular

⇒ Upper Triangular

154. Tri-diagonal and Tri-band matrix

* Tri-diagonal

M =	1	2	3	4	5
1	a_{11}	a_{12}	0	0	0
2	a_{21}	a_{22}	a_{23}	0	0
3	0	a_{32}	a_{33}	a_{34}	0
4	0	0	a_{43}	a_{44}	a_{45}
5	0	0	0	a_{54}	a_{55}

Main Diagonal $i-j=0$

Lower Diagonal $i-j=1$

Upper Diagonal $i-j=-1$

Non-zero $|i-j| \leq 1$

$m[i,j] \neq$ non-zero if $|i-j| \leq 1$

$m[i,j] = 0$ if $|i-j| > 1$

Non zero $= 5+4+4$

$= n+n-1+n-1$

$= 3n-2$

A	a_{21}	a_{32}	a_{43}	a_{54}	a_{11}	a_{22}	a_{33}	a_{44}	a_{55}	a_{12}	a_{23}	a_{34}	a_{45}
	Lower Diag				main Diag					upper Diag			

Index (A[i][j])

case 1 if $i-j=1$ index $= i-2$

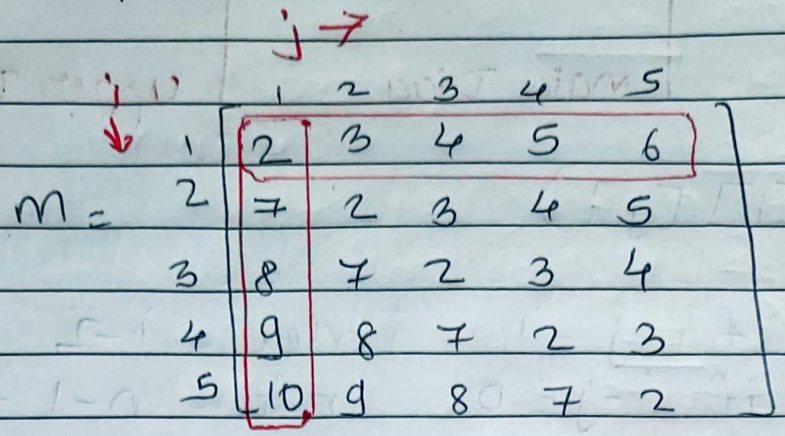
case 2 if $i-j=0$ index $= n-1+i-1$

case 3 if $i-j=-1$ index $= 2n-1+i-1$

* Square Band Matrix

	1	2	3	4	5	6	7	8
1					0	0	0	0
2						0	0	0
3							0	0
4								0
5	0							
6	0	0						
7	0	0	0					
8	0	0	0	0				

155. Toeplitz Matrix



$$m[i,j] = m[i-1, j-1]$$

elements - $n+n-1$

A	2	3	4	5	6	7	8	9	10
	0	1	2	3	4	5	6	7	8

row | col

Index (A[i][j])

Case 1 if $i \leq j$ // upper triangle

A[2][4] - 4-2 = 2

Index = j - i

Case 2 if $i \geq j$ // Lower triangle

Index = n + i - j - 1

156. Menu Driven Program for Matrices menu

1. create
2. Get
3. Set
4. Display

```
void main() {
    int *A, n, ch, x, i, j;
    printf("Enter dimension: ");
    scanf("%d", &n);
    A = (int*) malloc(n * sizeof(int));

    do
    {
        // Display menu
        switch(ch) {
            case 1:
                for(i=0; i<=n; i++)
                    printf("%d", &A[i-1]);
                break;

            case 2:
                printf("Enter indices: ");
                scanf("%d %d", &i, &j);
                if(i==j)
                    printf("%d", A[i-1]);
                else
                    printf("0");

            case 3:
                printf("Enter row, col and ele");
                scanf("%d %d %d", &i, &j, &x);
```

```
        if (i == j)
            A[i-1] = x;
        break;
    case 4:
        for(i=1; i<=n; i++){
            for(j=1; j<=n; j++){
                if (i == j)
                    printf("%d", A[i-1]);
                else
                    printf("0");
            }
            printf("\n");
        }
```

15. Menu Driven Program for Matrices using functions

```
void create(int A[], int n){
}
int get(int x[], int i, int j){
}
void set(int x[], int i, int j, int x){
}
void display(int x[], int n){
}
```

158. How to write C++ classes for All matrices

```
class Diagonal {
    private:
        int *A, n;
    Public:
        Diagonal (int n);
        void create ();
        int Get(int i, int j);
        void Set (int i, int j, int x);
        void Display ();
        ~Diagonal ();
};

Diagonal::Diagonal (int n) {
    this->n = n;
    A = new int [n];
}

Diagonal::~Diagonal () {
    delete [] A;
}

void Diagonal::create () {
    int x;
    for(i=1; i<=n; i++) {
        for(j=1; j<=n; j++) {
            cin >> x;
            if(i==j)
                A[i-1] = x;
        }
    }
}
```

```
void Diagonal::Display() {
    for (i=1; i<=n; i++) {
        for (j=1; j<=n; j++) {
            if (i==j)
                cout << A[i-1] << " ";
            else
                cout << "0 ";
        }
        cout << endl;
    }
}
```