There is a lot kinds of matrices here 8 of important ones

1- Digonal matrix

$$M = \begin{bmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 7 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 0 & 6 \end{bmatrix}$$

Row columen
5 x 5
only digonal not zero

To be digonal matrix
M[i][j] = 0 if $i \neq j$

We can save space in memoreg since most of it is zero, we make it liner array

A =

0	1	2	3	4
3	7	4	9	6

since non-zero elements are only dignoal and dignoal index alway i = j we make it liner

Let's code it

```
Void set(int A[], int i, int j, int x)
{
    if(i==j) A[i-1]=x; // if they are not equal than its not dignoal
}
int get(int A[], int i, int j)
{
    if(i==j) return A[i-1];
    return 0;
}
```

C++ class for Dignoal matrix

```
class Digonal
{
  private:
  int n;
  int *A;
  public:
  Digonal(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();
~Digonal();
};
```

```
Void Digonal::set(int i, int j,int x)
{
   if(i==j) A[i-1]=x;
}
int Digonal::get(int i,int j)
{
  if(i==j) return A[i-1];
    return 0;
}
Void Digonal::display()
{ for(int i=0; i<n; i++)
    for(int j=0;j<n; j++)
   { if(i==j) cout<<A[i]<<' ';
     else cout<<0<<' ';
   }
     cout<<endl;
  }
}
~Digonal() { delete[] A; }
```

lower Triangular Matrix Row-major mapping

	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}

5x5

the valued elements such index always $i \geq j$

therefore if $i < j$ element = 0

For square matrix number of elements = $\frac{n(n+1)}{2}$

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

$5^2 - \frac{5(5+1)}{2} = 25 - 15 = 10$ → elements

In Program we need to save memory, so we don't store zero elements

lets optimize it

A

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
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}

Row major method

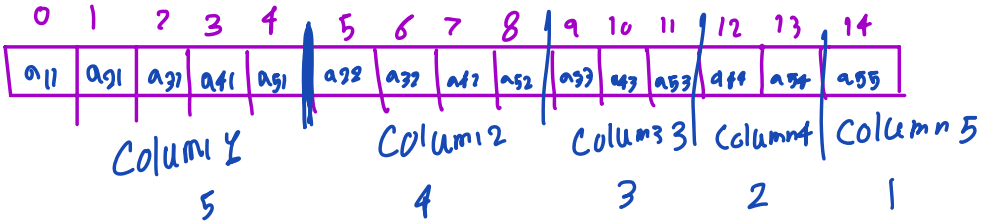
formula for mapping $A[i][j] = \frac{i(i-1)}{2} + j - 1$

$\frac{i(i-1)}{2}$: row pass

$j-1$: element in desired row pass

now column major method for mapping

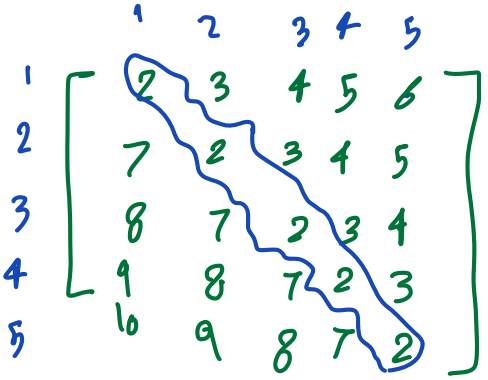
A



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

Upper triangular same concepts

Toeplitz matrix



matrix at 1 dignol are same

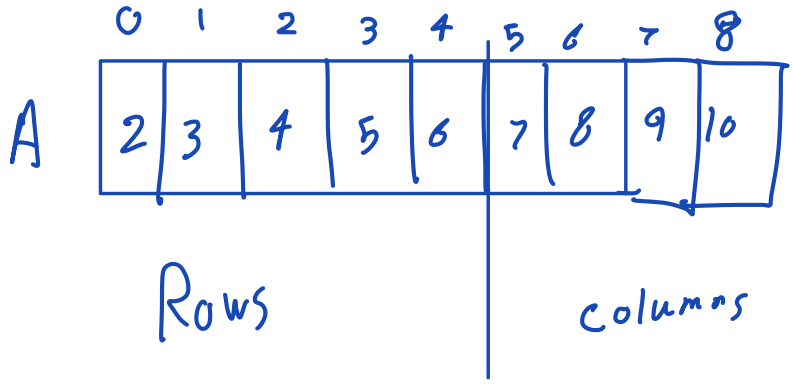
$$m[i][j] = m[i-1, j-1]$$

No. of elements = $n+n-1$

$5+5-1 = 9$

No. of rows & columns

let's store them in 1 dimension Array



formula

$A[i,j]$

3, 4, j-1

case $j \geq i$ upper digonal

$A[i,j] = A[j-i]$

case $j < i$ lower digonal

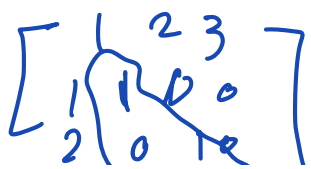
$A[i,j] = n+i-j-1$

move to colums in single dimension array

---

menu driven Program for matrices (digonal)

```
int main()
int *A, n, choice, x;
```



```
cout << "enter dimensions ";
cin >> n ;
A = new int[ n ];
do
{      // display menu -> you do it
 switch(ch)
 {
    case 1 : for i=0; to n; i++
               cin >> A[i];
             break;
 case 2 : cout << "enter indices ";
          cin >> i, j;
          if i=j cout << A[i-1];
          else cout << 0;
 case 3 : cout << enter row, col, element
          cin >> 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) cout<<A[i-1]_
else cout<< cout 0;
break;
while(true);
```