SPARSE MATRIX

↳ Having many number of non-zero elements

Methods for storing sparse matrix

1. Coordinate list / Three column representation
2. Compressed sparse row

1. Coordinate list / Three column representation

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

8 x 9 — 72 elements

72 x 2 = 144 bytes

row	column	element
8	9	8
1	8	3
2	3	8
2	6	10
4	1	4
6	3	2
7	4	6
8	2	9
8	5	5

2. Compressed sparse row

A[3, 8, 10, 4, 2, 6, 9, 5] → Non Zero elements

IA[0, 1, 3, 3, 4, 4, 5, 6, 8] (0 1 2 3 4 5 6 7 8 → Row Number) → Cummulative sum of number of non zero elements in each row

JA[8, 3, 6, 1, 3, 4, 2, 5] → No of column in which each non zero element is located

8 + 9 + 8 = 25 x 2 = 50 bytes

1. Coordinate List / Three column representation (ADDITION)

$A =$

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

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

$B =$

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

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

$C =$

	1	2	3	4	5	6
1	0	0	0	6	0	0
2	0	10	0	0	5	0
3	0	2	2	5	0	7
4	0	0	0	9	0	0
5	12	0	0	0	0	0

0	1	2	3	4	5	6	7	8	9
5	1	2	2	3	3	3	3	4	5
6	4	2	5	2	3	4	6	4	1
	6	10	5	2	2	5	7	9	12

Q

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

4 x 5 mxn

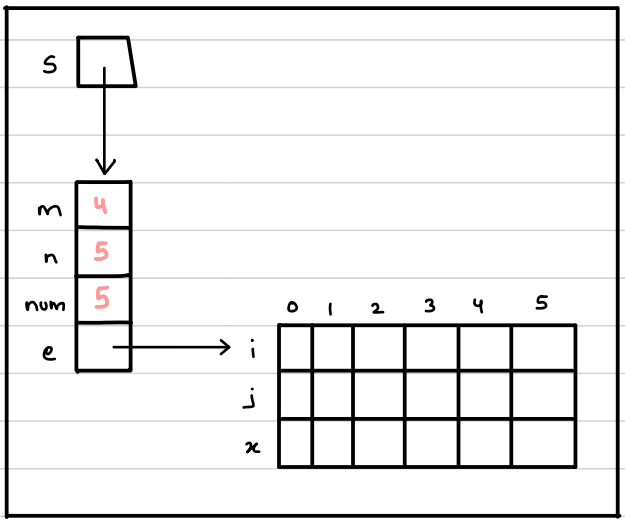
	0	1	2	3	4	5
i	4	1	2	2	3	4
j	5	3	1	4	1	5
k	5	7	2	5	9	4

CREATING SPARSE MATRIX PROGRAM

```
Struct Element
{
        int i;
        int j;
        int x;
};
Struct sparse
{
        int m;
        int n;
        int num;
        Struct Element *e;
};
void main()
{
        Struct sparse s;
        create (&s);
}
```

```
void create ( struct sparse *s)
{
    int i;
    printf(" Enter dimensions");
    scanf (" %d %d", &s->m, &s->n);
    printf(" Enter number of non-zero elements");
    scanf (" %d", &s->num);
    S->e = new Elements [s->num];
    printf(" Enter all elements ");

    for (i=0; i< s->num; i++)
            scanf(" %d %d %d", &s->e[i].i,
                               &s->e[i].j,
                               &s->e[i].x);
}
```

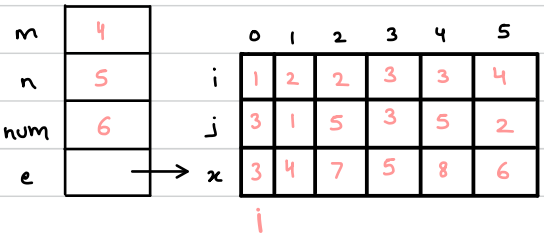
S->e = (Struct Element *) malloc
(s->num* sizeof(Struct Element))

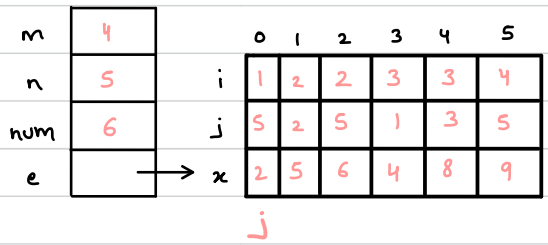


ADDING SPARSE MATRIX PROGRAM

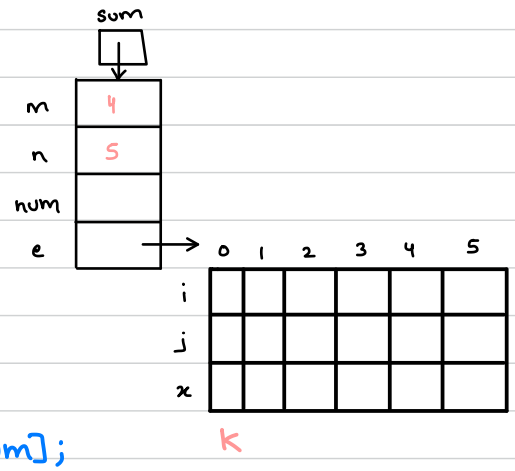
	1	2	3	4	5
1	0	0	3	0	0
2	4	0	0	0	7
3	0	0	5	0	8
4	0	6	0	0	0

	1	2	3	4	5
1	0	0	0	0	2
2	0	5	0	0	6
3	4	0	8	0	0
4	0	0	0	0	9






```
add ( struct sparse *s1 , struct sparse s2)
{
        struct Sparse *sum;

        if ( s1 -> m != s2-> m || s1->n != s2->n)
                return 0 ;
        sum = new sparse;
        sum -> m = s1 ->m;
        sum -> n = s1 ->n;
        sum -> e = new Element [s1 -> num + s2 -> num];

        while (i < s1 -> num && j < s2 ->num)
        {
                if ( s1. e[i].i < s2. e [j].i )
                        sum ->e[k++] = s1 ->e [i++];

                else if ( s1 -> e[i] .i > s2 ->e [j] .i)
                        sum -> e[k++] = s2 -> e[j++] ;

                else
                {
                        if ( s1-> e[i] .j < s2 -> e [j] .j)
                                sum -> e [k++] = s1 -> e[i++] ;
                        else if ( s1 -> e[i].j > s2 -> e [j] .j )
                                sum -> e[k++] = s2 -> e[j++];
                        else
                        {
                                sum -> e[k] = s1 -> e [i++] ;
                                sum -> e [k++] .x += s2 -> e [j++] .x ;
                        }
                }
        }
```



Rows are Compared

Column compared

POLYNOMIAL REPRESENTATION

1. Polynomial Representation
2. Evaluation of Polynomial
3. Addition of two Polynomials

$p(x) = 3x^5 + 2x^4 + 5x^2 + 2x + 7$

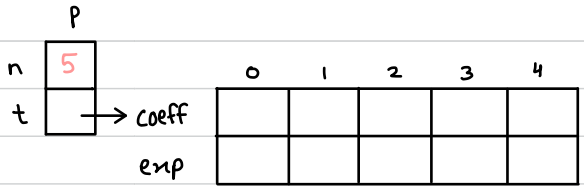
coeff	3	2	5	2	7
exp	5	4	2	1	0

$n = 5$

```
Struct Term
{
    int coeff;
    int Exp;
}
```

```
Struct Poly
{
    int n;
    Struct Term *t;
}
```

```
Struct Poly P;
printf(" No of non-zero terms");
scanf (" %d", &p.n);
p.t = new Term[p.n];
printf(" Enter polynomial terms ");
for(i=0; i<p.n; i++)
{
    printf(" Term No : %d ", i+1);
    scanf("%d %d", &p.t[i].coeff, &p.t[i].Exp);
}
```



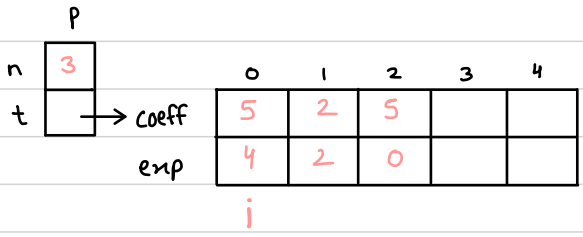
EVALUATION

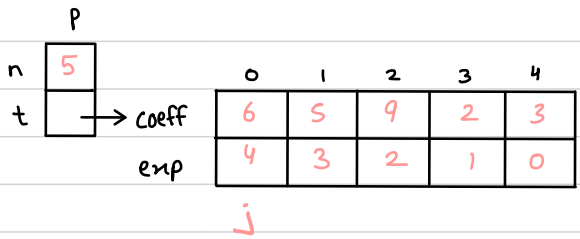
```
Struct Poly P;
x = 5; sum = 0;

for(i=0; i<p.n; i++)
    sum += p.t[i].coeff * pow(x, p.t[i].Exp);
```

POLYNOMIAL ADDITION

$p1(x) = 5x^4 + 2x^2 + 5$

$p2(x) = 6x^4 + 5x^3 + 9x^2 + 2x + 3$






```
while ( i < p1.n && j < p2.n)
{
        if ( p1.t[i].Exp > p2.t[j].Exp)
                p3.t[k++] = p1.t[i++];

        else if ( p2.t[j].Exp > p1.t[i].Exp)
                p3.t[k++] = p2.t[j++];

        else
        {
                p3.t[k].Exp = p1.t[i].Exp;
                p3.t[k++].coeff = p1.t[i++].coeff + p2.t[j++].coeff;
        }
}
```