**Experiment 3**

**AIM:** Write a program to add 2 sparse matrices

**Algo:**

define structure sparse:

int r,c,v

Initialize sparse a[100],b[100],c[100]

define bool compare(sparse a,sparse b):

if a.r != b.r then:

return a.r < b.r

return a.c < b.c

read a[0].r, a[0].c, a[0].v

for i=1 to a[0].v do:

read a[i].r, a[i].c, a[i].v

endfor

read b[0].r, b[0].c, b[0].v

for i=1 to b[0].v do:

read b[i].r, b[i].c, b[i].v

endfor

Initialize i=1,j=1,k=1

while i<=a[0].v and j<=b[0].v do:

if compare(a[i],b[j]) then:

c[k]=a[i]

i=i+1

else:

c[k]=b[j]

j=j+1

endif

k=k+1

endwhile

while i<=a[0].v do:

c[k]=a[i]

i=i+1

k=k+1

endwhile

while j<=b[0].v do:

c[k]=b[j]

j=j+1

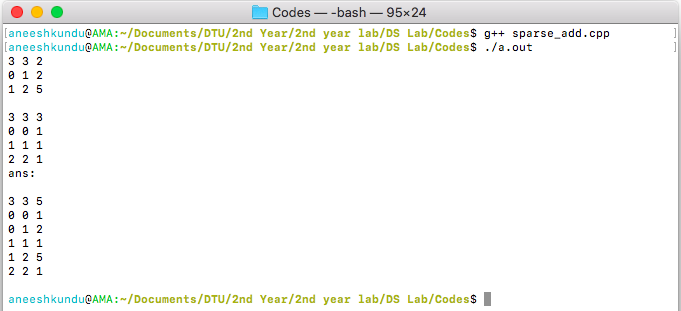
k=k+1

endwhile

**Code:**

1. #include < iostream >
2. using namespace std;
3. struct sparse {
4. int row, col, val;
5. sparse() {}
6. sparse(int r, int c, int v) {
7. row = r;
8. col = c;
9. val = v;
10. }
11. };
12. sparse a[20], b[20], ans[20];
13. void read\_sparse(sparse \* s) {
14. cin >> s[0].row >> s[0].col >> s[0].val;
15. for (int i = 1; i <= s[0].val; i++)
16. cin >> s[i].row >> s[i].col >> s[i].val;
17. }
18. void print\_sparse(sparse \* s) {
19. cout << endl;
20. int k = s[0].val;
21. for (int i = 0; i <= k; i++)
22. cout << s[i].row << " " << s[i].col << " " << s[i].val << endl;
23. cout << endl;
24. }
25. void add\_sparse(sparse \* a, sparse \* b, sparse \* c) {
26. int i = 1, j = 1, k = 1;
27. int i\_max = a[0].val, j\_max = b[0].val;
28. while (i <= i\_max && j <= j\_max) {
29. if (a[i].row < b[j].row)
30. c[k++] = a[i++];
31. else if (a[i].row == b[j].row) {
32. if (a[i].col < b[j].col)
33. c[k++] = a[i++];
34. else if (a[i].col == b[j].col) {
35. c[k] = sparse(a[i].row, a[i].col, a[i].val + b[j].val);
36. i++;
37. j++;
38. k++;
39. } else
40. c[k++] = b[j++];
41. } else
42. c[k++] = b[j++];
43. }
44. while (i <= i\_max)
45. c[k++] = a[i++];
46. while (j <= j\_max)
47. c[k++] = b[j++];
48. c[0] = sparse(a[0].row, a[0].col, k - 1);
49. }
50. int main() {
51. read\_sparse(a);
52. read\_sparse(b);
53. add\_sparse(a, b, ans);
54. cout << "ans:\n";
55. print\_sparse(ans);
56. return 0;
57. }

**Output:**



**Discussion:**

In this program, we maintain 2 pointers I and j for the 2 sparse matrices. If the row and col don’t match then we place the value with the smaller index in c and increment its pointer, if the row and col matches we add the values and advance both I and j, When either of the 2 pointers reaches its max value we copy the remaining of the other sparse matrix in c.

**Conclusion:**

The program adds 2 sparse matrices, with n and m entries, with time complexity .