SPARSE MATRIX

Exp. No.:
AIM:

ALGORITHM:

PROGRAM:

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>

int i, j;

struct Node
{
   int value;
   int row;
   int column;
   struct Node *next;
} *head = NULL;
typedef struct Node node;

node *create_node(int data, int row, int column)
{
   node *temp;
   temp = (node *)malloc(sizeof(node));
   temp->value = data;
   temp->row = row;
   temp->column = column;
   temp->next = NULL;
   return temp;
}

void insert_node(int data, int row, int column)
{
   node *temp = head;
   if (temp == NULL)
   {
      head = create_node(data, row, column);
   }
   else
   {
      while (temp->next != NULL)
```

```
      {
         temp = temp->next;
      }
      temp->next = create_node(data, row, column);
   }
}

void create_matrix(int **array, int rows, int columns)
{
   for (i = 0; i < rows; i++)
   {
      for (j = 0; j < columns; j++)
      {
         if (array[i][j] != 0)
         {
            insert_node(array[i][j], i + 1, j + 1);
         }
      }
   }
}

void printMatrix()
{
   node *temp = head;
   while (temp != NULL)
   {
      printf("\nRow: %d, Column: %d, Data: %d", temp->row, temp->column, temp-
>value);
      temp = temp->next;
   }
}

void checkSparse(int **array, int rows, int columns)
{
   int count = 0;
   int total = rows * columns;
   for (i = 0; i < rows; i++)
   {
      for (j = 0; j < columns; j++)
```

```
        {
            if (array[i][j] == 0)
            {
                count++;
            }
        }
    }
    if (count >= (total / 2) + 1)
    {
        printf("\nSparse Matrix");
    }
    else
    {
        printf("\nNot a Sparse Matrix");
    }
}

int main()
{
    int **array;
    int row, column;
    printf("Enter Number of Rows and Columns:");
    scanf("%d %d", &row, &column);
    array = (int **)malloc(row * sizeof(int));
    for (i = 0; i < row; i++)
    {
        array[i] = (int *)malloc(column * sizeof(int));
    }
    printf("Enter Elements of Array:");
    for (i = 0; i < row; i++)
    {
        for (j = 0; j < column; j++)
        {
            scanf("%d", &array[i][j]);
        }
    }
    create_matrix(array, row, column);
    checkSparse(array, row, column);
    printMatrix();
```

```
    getch();
    clrscr();
    return 0;
}
```

OUTPUT:

```
Enter Number of Rows and Columns:4 5
Enter Elements of Matrix:12 5 0 98 0 0 0 0 47 0 0 -9 0 0 0 3 0 0 -1 0

Given Matrix is a Sparse Matrix
Compact Form:

Row: 1, Column: 1, Data: 12
Row: 1, Column: 2, Data: 5
Row: 1, Column: 4, Data: 98
Row: 2, Column: 4, Data: 47
Row: 3, Column: 2, Data: -9
Row: 4, Column: 1, Data: 3
Row: 4, Column: 4, Data: -1
```

RESULT: