ASSIGNMENT NO:7 **DATE:06/11/2014**

PROGRAM TITLE: Dynamically create a two dimensional matrix and write a user-defined function which takes this matrix as an argument and prints its individual row and column sums.

PROGRAM ALGORITHM:

```
algo main()
{
      input size of matrix
      dynamically allocate the matrix
      simultaneously check if memory corectly allocated or not
      input elements of matrix
      call  sum(arguments: matrix,rowsize,columnsize)
      free the allocated memory
}
algo sum(parameters: matrix,rowsize,columnsize)
{
      calculate individual row sum
      calculate individual column sum
      print matrix along with row and column sums
}
```

PROGRAM CODE:

indisum.c

```
/*C Program to Dynamically allocate a 2D Matrix and find its individual
Row and Column Sum*/
#include <stdio.h>
#include <stdlib.h>
int main()
{
      int i,j,r,c;

      /*Read the Size of the array*/
      printf("Enter size: ");
      scanf("%d %d",&r,&c);

      /*Dynamically(contiguous) allocate the 2D Matrix and Check if
memory correctly allocated or not*/
      int **mat=(int **)malloc(r*sizeof(int*));
      if(!mat)
            {
                  printf("Allocation failed\n");
                  exit(1);
            }
      mat[0]=(int *)malloc(r*c*sizeof(int));
      if(!mat[0])
      {
            printf("Allocation failed\n");
            exit(2);
      }
```

```
	for(i=1;i<r;i++)
	{
		mat[i]=mat[0]+(c*i);
		if(!mat[i])
		{
			printf("Allocation failed\n");
			exit(3);
		}
	}

	/*Read the elements of the Matrix*/
	printf("Enter elements\n");
		for(i=0;i<r;i++)
		{
			for(j=0;j<c;j++)
			{
				scanf("%d",&mat[i][j]);
			}
		}

	/*Call function to find Individual row and column sum*/
	sum(mat,r,c);

	/*Free the allocated memory*/
	free(mat);
	return 0;
}
```

rcsum.c

```
/*C Program to Find and Display Individual Row and Column Sum*/
#include <stdio.h>
int sum(int **mat,int r,int c)
{
	int i,j,sr[r],sc[c];
	printf("The given Matrix along with Individual Row and Column Sum
is:\n");

	/*Find Individual Row and Column Sum and Display*/
	for(i=0;i<r;i++)
	{
		sr[i]=0;
		for(j=0;j<c;j++)
		{
			sr[i]=sr[i]+mat[i][j];
		}
	}
	for(i=0;i<c;i++)
	{
		sc[i]=0;
		for(j=0;j<r;j++)
		{
			sc[i]=sc[i]+mat[j][i];
		}
	}
```

```
	for(i=0;i<r;i++)
	{
		for(j=0;j<c;j++)
		{
			printf("%d\t",mat[i][j]);
		}
		printf("|%d\n",sr[i]);
	}
	for(i=0;i<c;i++)
	{
		printf("_\t");
	}
	printf(".\n");
	for(i=0;i<c;i++)
	{
		printf("%d\t",sc[i]);
	}
	printf("\n");
	return 0;
}
```

OUTPUT:

Set 1:

```
Enter size: 3 2
Enter elements
1 2 3 4 5 6
The given Matrix along with Individual Row and Column Sum is:
1	2	|3
3	4	|7
5	6	|11
_	_	.
9	12
```

Set 2:

```
Enter size: 2 3
Enter elements
1 2 3 4 5 6
The given Matrix along with Individual Row and Column Sum is:
1	2	3	|6
4	5	6	|15
_	_	_	.
5	7	9
```

Set 3:

```
Enter size: 3 3
Enter elements
1 2 3 4 5 6 7 8 9
The given Matrix along with Individual Row and Column Sum is:
1	2	3	|6
4	5	6	|15
7	8	9	|24
_	_	_	.
12	15	18
```

DISCUSSION:

This Program works for all cases (row-size=column-size, row-size>column-size and row-size<column-size). The Matrix is allocated dynamically in contiguous fashion.