Tema 2

1. **Modificați codul de mai jos pentru a defini o structură punct de coordonate (x,y,z).**

```
#include <stdio.h>
#include "mpi.h"

struct _point {
        int x, y;
};

int main( int argc, char ** argv ) {
        int rank, size;
        int newRank, newSize;

        MPI_Datatype simpleStruct;
        MPI_Status stat;
        struct _point test;

        MPI_Init( &argc, &argv );
        MPI_Comm_rank( MPI_COMM_WORLD, &rank );
        MPI_Comm_size( MPI_COMM_WORLD, &size );

        MPI_Type_contiguous( 2, MPI_INT, &simpleStruct );
        MPI_Type_commit( &simpleStruct );

        if( 0 == rank ) {
                test.x = 10;
                test.y = 12;
                MPI_Send( &test, 1, simpleStruct, 1, 99, MPI_COMM_WORLD );
        } else {
                if( 1 == rank ) {
                        MPI_Recv( &test, 1, simpleStruct, 0, 99, MPI_COMM_WORLD, &stat );
                        printf( "%d %d\n", test.x, test.y );
                }
        }

        MPI_Type_free( &simpleStruct );
        MPI_Finalize( );
        return 0;
}
```

2. **Modificati codul de mai jos pentru a defini o structură de tip vector în care elementele sunt puncte de coordonate (x,y,z) separate cu o zonă liberă de lunginea a două puncte.**

```
#include <stdio.h>
#include "mpi.h"

struct _point {
    int x, y;
};

int main( int argc, char ** argv ) {
    int rank, size;
    int newRank, newSize;
    int i;

    MPI_Datatype simpleStruct, vectorType;
    MPI_Status stat;
    struct _point test[ 10 ];

    MPI_Init( &argc, &argv );
    MPI_Comm_rank( MPI_COMM_WORLD, &rank );
    MPI_Comm_size( MPI_COMM_WORLD, &size );

    MPI_Type_contiguous( 2, MPI_INT, &simpleStruct );
    MPI_Type_commit( &simpleStruct );
    MPI_Type_vector( 10, 1, 1, simpleStruct, &vectorType );
    MPI_Type_commit( &vectorType );

    if( 0 == rank ) {
        for( i=0; i<10; ++i ) {
            test[i].x = i+1;
            test[i].y = i+2;
        }
        MPI_Send( test, 1, vectorType, 1, 99, MPI_COMM_WORLD );
    } else {
        if( 1 == rank ) {
            MPI_Recv( test, 1, vectorType, 0, 99, MPI_COMM_WORLD, &stat );
            for( i=0; i<10; ++i ) {
                printf( "%d %d\n", test[i].x, test[i].y );
            }
        }
    }

    MPI_Type_free( &vectorType );
    MPI_Type_free( &simpleStruct );
    MPI_Finalize( );
    return 0;
}
```

3. **Aplicați ideea din exercițiul 2 pentru a extrage coloana a doua dintr-o matrice 3x3 formată din numere întregi.**