```c
//--------------------------------------------------------------------------
// new type - vector_2
//--------------------------------------------------------------------------
#if (defined __linux__) || (defined _AIX) || (defined __APPLE__)
    #include <sys/types.h>
    #include <sys/stat.h>
    #include <unistd.h>
#elif (defined _WIN32) || (defined _WIN64)
    #include <conio.h>
    #include <direct.h>
#endif

#include<mpi.h>
#include<stdlib.h>
#include<stdio.h>
//--------------------------------------------------------------------------
int main(int argc, char* argv[])
{
    int i, j, rank, size;
    double vector_0[5][5], vector_1[5][5];

    MPI_Status status;
    MPI_Datatype my_type;

    MPI_Init(&argc, &argv);

    MPI_Comm_size(MPI_COMM_WORLD, &size);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    if (rank == 0)
        printf("\nThere are %d processes.\n", size);

    MPI_Type_vector(5, 1, 5, MPI_DOUBLE, &my_type);

    MPI_Type_commit(&my_type);

    if (rank == 0)
        for (i = 0; i < 5; i++)
            for (j = 0; j < 5; j++)
                vector_0[i][j] = (double)j + 1;

    if (rank == 1) {
        for (i = 0; i < 5; i++)
            for (j = 0; j < 5; j++)
                vector_1[i][j] = 0;

        printf("\nValues of a matrix on a process 1 before sending:\n");
        for (i = 0; i < 5; i++) {
            for (j = 0; j < 5; j++)
                printf("%lf ", vector_1[i][j]);
            printf("\n");
        }
    }

    if (rank == 0)
        MPI_Ssend(&(vector_0[0][3]), 1, my_type, 1, 0, MPI_COMM_WORLD);

    if (rank == 1) {
        MPI_Recv(&(vector_1[0][3]), 1, my_type, 0, 0, MPI_COMM_WORLD, &status);
        printf("\nValues of the matrix on a process 1 after sending:\n");
        for (i = 0; i < 5; i++) {
            for (j = 0; j < 5; j++)
                printf("%lf ", vector_1[i][j]);
            printf("\n");
        }
    }

    MPI_Finalize();

    return 0;
}

```