

```
1 //-----
2 // new type - structure
3 //-----
4 #if (defined __linux__) || (defined _AIX)
5     #include <sys/types.h>
6     #include <sys/stat.h>
7     #include <unistd.h>
8 #elif (defined _WIN32) || (defined _WIN64)
9     #include <conio.h>
10    #include <direct.h>
11 #endif
12
13 #include<mpi.h>
14 #include<stdlib.h>
15 #include<stdio.h>
16 #include <iostream>
17 using namespace std;
18
19 struct Tstruct {
20     int ival1;
21     int ival2;
22     double dval1;
23     double dval2;
24     double dval3;
25 };
26
27 int main(int argc, char** argv)
28 {
29     Tstruct x_struct, y_struct;
30     int rank, size;
31
32     MPI_Status status;
33     MPI_Datatype mytype, array_of_types[3];
34     int array_of_blocklengths[3];
35     MPI_Aint adress[3], array_of_displacements[3];
36
37     MPI_Init(&argc, &argv);
38
39     MPI_Comm_size(MPI_COMM_WORLD, &size);
40     MPI_Comm_rank(MPI_COMM_WORLD, &rank);
41     if (rank == 0) {
42         printf("\nThere are %d processes.\n", size);
43         fflush(stdout); // Skuste zakomentovat tento riadok
44     }
45
46     array_of_types[0] = MPI_INT;
47     array_of_types[1] = MPI_DOUBLE;
48     array_of_types[2] = MPI_UB; // FINTA !!!
49
50     array_of_blocklengths[0] = 2;
51     array_of_blocklengths[1] = 3;
52     array_of_blocklengths[2] = 1;
53
```

```
54 MPI_Get_address(&x_struct.ival1, &adress[0]);
55 MPI_Get_address(&x_struct.dval1, &adress[1]);
56 array_of_displacements[0] = 0;
57 array_of_displacements[1] = adress[1] - adress[0];
58 array_of_displacements[2] = sizeof(x_struct);
59
60 MPI_Type_create_struct(3, array_of_blocklengths, array_of_displacements,
61     array_of_types, &mytype);
62
63 MPI_Type_commit(&mytype);
64
65 x_struct.ival1 = 1;
66 x_struct.ival2 = 2;
67 x_struct.dval1 = (double)1.14;
68 x_struct.dval2 = (double)2.14;
69 x_struct.dval3 = (double)3.14;
70 y_struct.ival1 = 0;
71 y_struct.ival2 = 0;
72 y_struct.dval1 = (double)0;
73 y_struct.dval2 = (double)0;
74 y_struct.dval3 = (double)0;
75
76 if (rank == 1) {
77     printf("\nValues of y_struct before sending:\n");
78     printf("y_struct.ival1 = %d\n", y_struct.ival1);
79     printf("y_struct.ival2 = %d\n", y_struct.ival2);
80     printf("y_struct.dval1 = %lf\n", y_struct.dval1);
81     printf("y_struct.dval2 = %lf\n", y_struct.dval2);
82     printf("y_struct.dval3 = %lf\n", y_struct.dval3);
83 }
84
85 if (rank == 0)
86     MPI_Ssend(&x_struct, 1, mytype, 1, 0, MPI_COMM_WORLD);
87
88 if (rank == 1) {
89     MPI_Recv(&y_struct, 1, mytype, 0, 0, MPI_COMM_WORLD, &status);
90     printf("\nValues of y_struct after sending:\n");
91     printf("y_struct.ival1 = %d\n", y_struct.ival1);
92     printf("y_struct.ival2 = %d\n", y_struct.ival2);
93     printf("y_struct.dval1 = %lf\n", y_struct.dval1);
94     printf("y_struct.dval2 = %lf\n", y_struct.dval2);
95     printf("y_struct.dval3 = %lf\n", y_struct.dval3);
96 }
97
98 MPI_Finalize();
99
100 return 0;
101 }
102
```