```cpp
//--------------------------------------------------------------------------
// new type - vector_1
//--------------------------------------------------------------------------
#if (defined __linux__) || (defined _AIX)
   #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, rank, size;
   double vector_0[5], vector_1[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, 1, MPI_DOUBLE, &my_type);

   MPI_Type_commit(&my_type);

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

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

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

   if (rank == 0) // pre každé pole platí array = &array = &(array[0])
      MPI_Ssend(&(vector_0[0]), 1, my_type, 1, 0, MPI_COMM_WORLD);

   if (rank == 1) {
```

```
54        MPI_Recv(&(vector_1[0]), 1, my_type, 0, 0, MPI_COMM_WORLD, &status);
55        printf("\nValues of the vector_1 a process 1 after sending:\n");
56        for (i = 0; i < 5; i++)
57           printf("%lf ", vector_1[i]);
58        printf("\n");
59
60     }
61
62     MPI_Finalize();
63
64     return 0;
65  }
66
```