Programowanie Równoległe, Lista zadań MPI-1
Maciej Matyka, maq@ift.uni.wroc.pl

Uwaga: do kompilacji przykładów używamy mpic++ (wrapper dla kompilatora g++ z Open MPI).

1. Przepisz poniższy przykład, skompiluj i uruchom na kilku rdzeniach procesora

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

int main (int argc, char *argv[])
{
      MPI_Init(&argc, &argv);
      printf("Hello world!\n");
      MPI_Finalize();
      return 0;
}
```

2. Uzupełnij program z punktu 1. tak, aby proces o identyfikatorze 0 wypisał całkowitą ilość procesów w komunikatorze MPI_COMM_WORLD.

3. Uzupełnij program o podstawową komunikację między procesami i przetestuj go. Np. proces 0 wysyła procesowi 1 liczbę typu *double*. Proces 1 po odebraniu wiadomości niech wypisze ją na ekranie.

4. Poniższy program ma za zadanie wymienić 10 elementów między procesami. Skompiluj go, uruchom, spróbuj przeanalizować co się dzieje.

```
#include <stdio.h>
#include <mpi.h>
void main (int argc, char **argv)
{
   int myrank;
   MPI_Status status;
   double a[10], b[10];
   MPI_Init(&argc, &argv);
   MPI_Comm_rank(MPI_COMM_WORLD, &myrank);
   if( myrank == 0 )
   {
      // odbierz i wy lij komunikat
      MPI_Recv( b, 10, MPI_DOUBLE, 1, 19, MPI_COMM_WORLD, status );
      MPI_Send( a, 10, MPI_DOUBLE, 1, 17, MPI_COMM_WORLD );
   }
   else if( myrank == 1 )
   {
      // odbierz i wy lij komunikat
      MPI_Recv( b, 10, MPI_DOUBLE, 0, 17, MPI_COMM_WORLD, status );
      MPI_Send( a, 10, MPI_DOUBLE, 0, 19, MPI_COMM_WORLD );
   }
   MPI_Finalize();
}
```

5. Napisz program, który wykona równolegle K razy operację sqrt(x) na N-elementowej tablicy, gdzie każdy proces odpowiada za swój fragment danych. Sprawdź czasy wykonania dla 1,2,3,4 i 8 procesów MPI. Czy uzyskałeś przyspieszenie? Dlaczego?