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

int main(int argc, char** argv) {
  // Initialize the MPI environment
  MPI_Init(NULL, NULL);
  // Find out rank, size
  int world_rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
  int world_size;
  MPI_Comm_size(MPI_COMM_WORLD, &world_size);

  // We are assuming at least 2 processes for this task
  if (world_size < 2) {
    fprintf(stderr, "World size must be greater than 1 for %s\n",
argv[0]);
    MPI_Abort(MPI_COMM_WORLD, 1);
  }
  char message[100];
  int len;
  if (world_rank == 0) {
    strcpy(message, "hello world!");
    len = strlen(message);
    MPI_Send(
      /* data         = */ &message,
      /* count        = */ len+1,
      /* datatype     = */ MPI_CHAR,
      /* destination  = */ 1,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD);
  } else if (world_rank == 1) {
    MPI_Recv(
      /* data         = */ &message,
      /* count        = */ 100,
      /* datatype     = */ MPI_CHAR,
      /* source       = */ 0,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD,
      /* status       = */ MPI_STATUS_IGNORE);
    printf("Process 1 received string %s from process 0\n", message);
  }
  MPI_Finalize();
}

> mpicc messaging.c
> mpirun -n 2 ./a.out
Process 1 received string hello world! from process 0
```

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

int main(int argc, char** argv) {
  // Initialize the MPI environment
  MPI_Init(NULL, NULL);
  // Find out rank, size
  int world_rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
  int world_size;
  MPI_Comm_size(MPI_COMM_WORLD, &world_size);

  // We are assuming at least 2 processes for this task
  if (world_size < 2) {
    fprintf(stderr, "World size must be greater than 1 for %s\n",
argv[0]);
    MPI_Abort(MPI_COMM_WORLD, 1);
  }
  char message[100];
  int len;
  if (world_rank == 0) {
    strcpy(message, "hello world!");
    len = strlen(message);
    MPI_Send(
      /* data         = */ &message,
      /* count        = */ len+1,
      /* datatype     = */ MPI_CHAR,
      /* destination  = */ 1,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD);
  } else if (world_rank == 1) {
    MPI_Recv(
      /* data         = */ &message,
      /* count        = */ 100,
      /* datatype     = */ MPI_CHAR,
      /* source       = */ 0,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD,
      /* status       = */ MPI_STATUS_IGNORE);
    printf("Process 1 received string %s from process 0\n", message);
  }
  MPI_Finalize();
}
if (world_rank == 0) {
    strcpy(message, "hello world!");
    len = strlen(message);
    MPI_Send(
      /* data         = */ &message,
```

```
      /* count        = */ len+1,
      /* datatype     = */ MPI_CHAR,
      /* destination  = */ 1,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD);
  } else if (world_rank == 1) {
    MPI_Recv(
      /* data         = */ &message,
      /* count        = */ 100,
      /* datatype     = */ MPI_CHAR,
      /* source       = */ 0,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD,
      /* status       = */ MPI_STATUS_IGNORE);
    printf("Process 0 received string %s from process 1\n", message);
  }
  MPI_Finalize();
}

> mpicc messaging.c
> mpirun -n 2 ./a.out
Process 1 received string hello world! from process 0
Process 0 received string hello world! from process 1

#include <mpi.h>
#include <stdio.h>
#include <stdlib.h>
#include<string.h>

int main(int argc, char** argv) {
  // Initialize the MPI environment
  MPI_Init(NULL, NULL);
  // Find out rank, size
  int world_rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
  int world_size;
  MPI_Comm_size(MPI_COMM_WORLD, &world_size);

  // We are assuming at least 2 processes for this task
  if (world_size < 2) {
    fprintf(stderr, "World size must be greater than 1 for %s\n",
argv[0]);
    MPI_Abort(MPI_COMM_WORLD, 1);
  }
  char message[100];
  int len;
  if (world_rank == 0) {
    strcpy(message, "hello world!");
    len = strlen(message);
    MPI_Send(
```

```
      /* data         = */ &message,
      /* count        = */ len+1,
      /* datatype     = */ MPI_CHAR,
      /* destination  = */ 1,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD);
  } else if (world_rank == 1) {
    strcpy(message, "hello world!");
    len = strlen(message);
    MPI_Send(
      /* data         = */ &message,
      /* count        = */ len+1,
      /* datatype     = */ MPI_CHAR,
      /* destination  = */ 0,
      /* tag          = */ 1,
      /* communicator = */ MPI_COMM_WORLD);
  }
  if (world_rank == 1) {
    MPI_Recv(
      /* data         = */ &message,
      /* count        = */ 100,
      /* datatype     = */ MPI_CHAR,
      /* source       = */ 0,
      /* tag          = */ 0,
      /* communicator = */ MPI_COMM_WORLD,
      /* status       = */ MPI_STATUS_IGNORE);
    printf("Process 1 received string %s from process 0\n", message);
  } else if (world_rank == 0) {
    MPI_Recv(
      /* data         = */ &message,
      /* count        = */ 100,
      /* datatype     = */ MPI_CHAR,
      /* source       = */ 1,
      /* tag          = */ 1,
      /* communicator = */ MPI_COMM_WORLD,
      /* status       = */ MPI_STATUS_IGNORE);
    printf("Process 0 received string %s from process 1\n", message);
  }
  MPI_Finalize();
}

> mpicc messaging.c
> mpirun -n 2 ./a.out
Process 1 received string hello world! from process 0
Process 0 received string hello world! from process 1
```