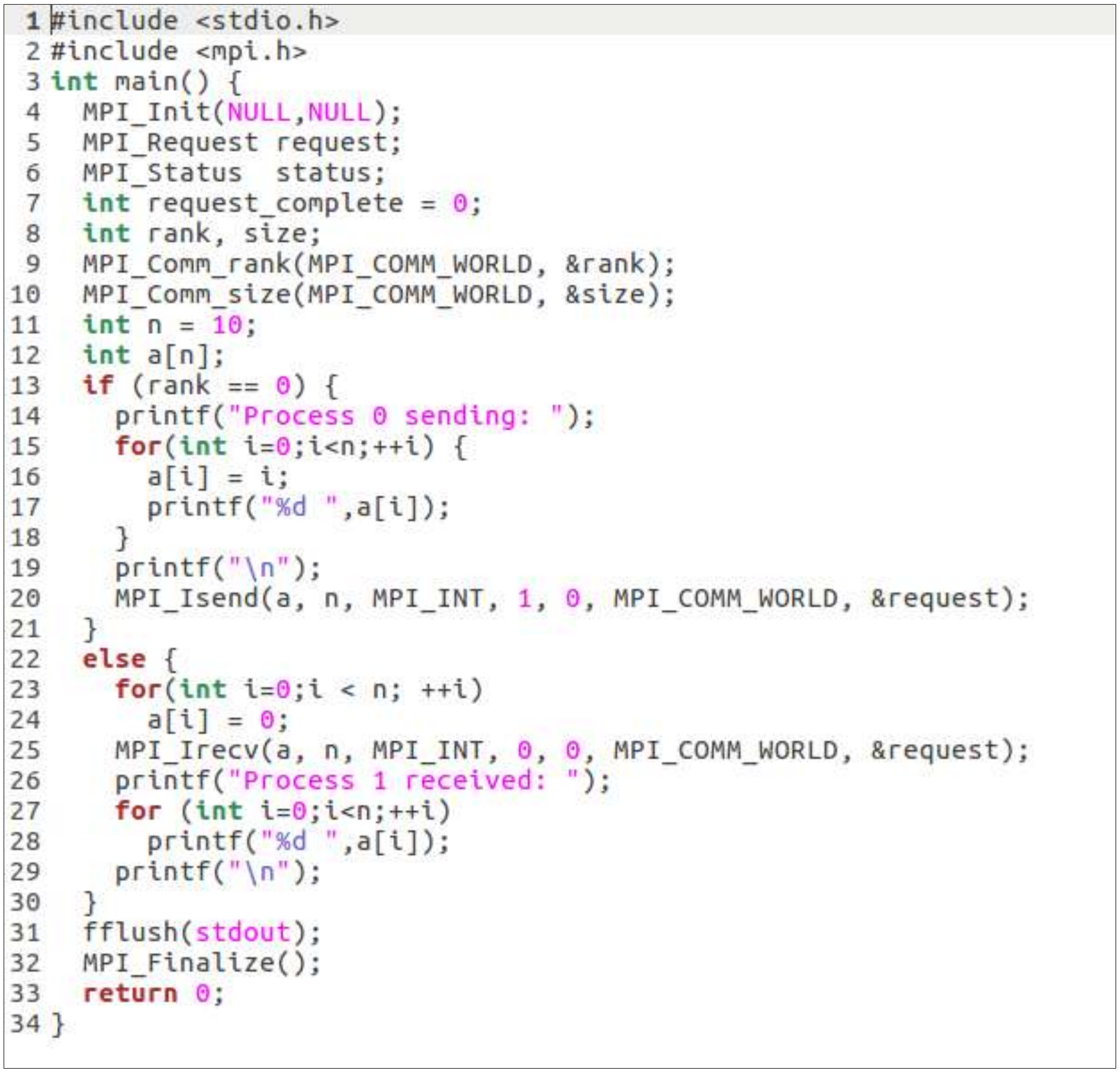
DHRUV GARG 16 BCE 1190

PDC LAB
CHALLENGING TASK

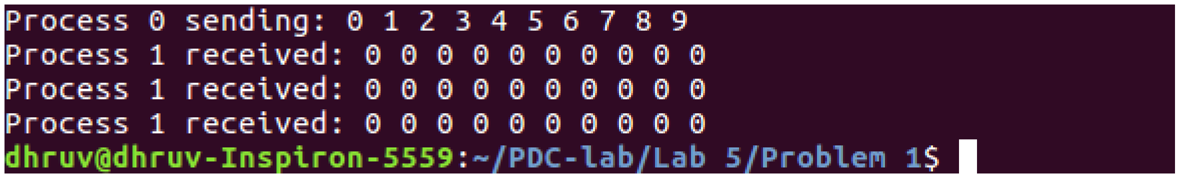
Race Condition in MPI_Isend and MPI_Irecv

Code



```c
#include <stdio.h>
#include <mpi.h>
int main() {
  MPI_Init(NULL,NULL);
  MPI_Request request;
  MPI_Status  status;
  int request_complete = 0;
  int rank, size;
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  int n = 10;
  int a[n];
  if (rank == 0) {
    printf("Process 0 sending: ");
    for(int i=0;i<n;++i) {
      a[i] = i;
      printf("%d ",a[i]);
    }
    printf("\n");
    MPI_Isend(a, n, MPI_INT, 1, 0, MPI_COMM_WORLD, &request);
  }
  else {
    for(int i=0;i < n; ++i)
      a[i] = 0;
    MPI_Irecv(a, n, MPI_INT, 0, 0, MPI_COMM_WORLD, &request);
    printf("Process 1 received: ");
    for (int i=0;i<n;++i)
      printf("%d ",a[i]);
    printf("\n");
  }
  fflush(stdout);
  MPI_Finalize();
  return 0;
}
```

Output

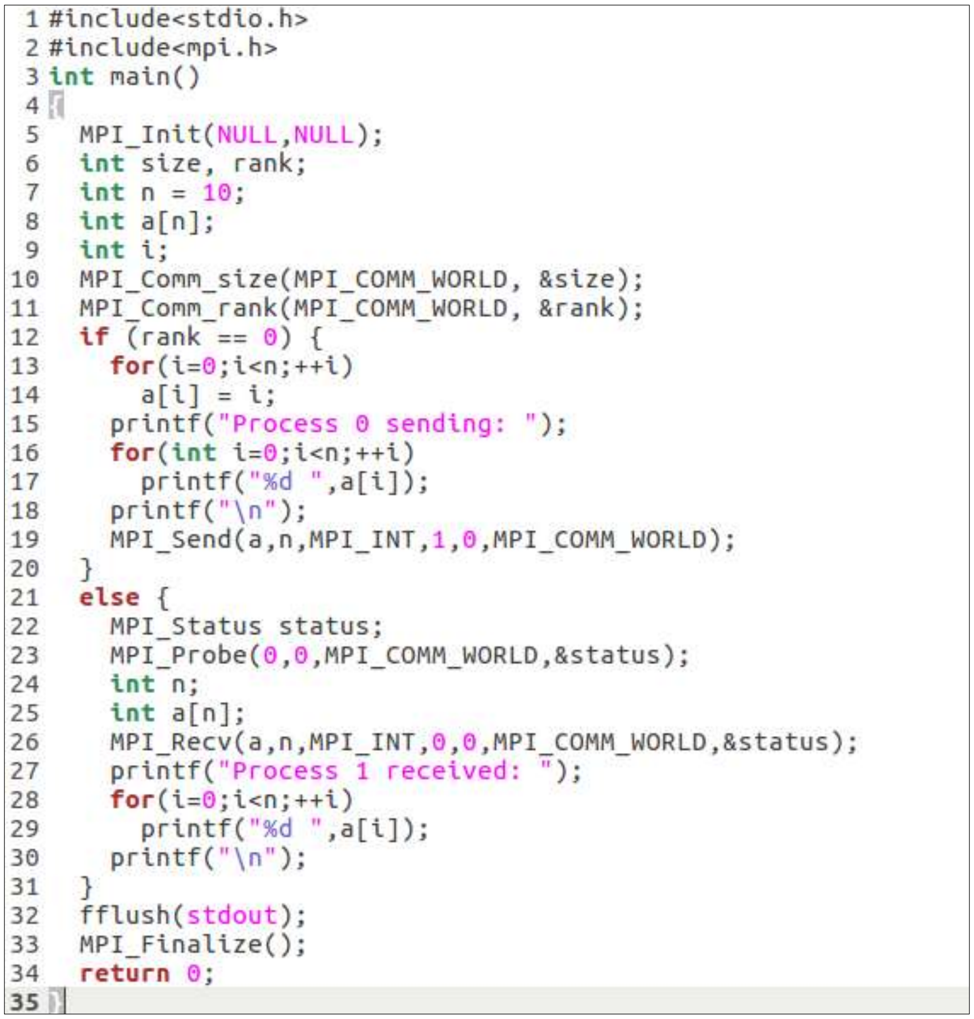


```
Process 0 sending: 0 1 2 3 4 5 6 7 8 9
Process 1 received: 0 0 0 0 0 0 0 0 0 0
Process 1 received: 0 0 0 0 0 0 0 0 0 0
Process 1 received: 0 0 0 0 0 0 0 0 0 0
dhruv@dhruv-Inspiron-5559:~/PDC-lab/Lab 5/Problem 1$
```

The following condition happens due to the race condition between process 1 and process 0. P0 sends a message to P1 and then goes to the same buffer from it sent the values. P1 waits for receiving the values but id P0 has re-written the buffer before sending all the values, P1 receives the wrong set of values.

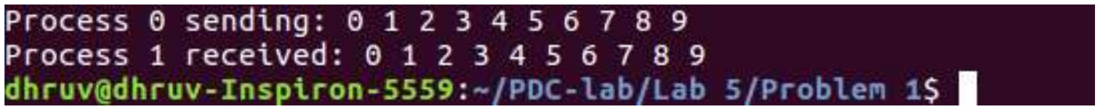
__Solution to Race Condition in MPI_send and MPI_recv__

__Code__



```
#include<stdio.h>
#include<mpi.h>
int main()
{
  MPI_Init(NULL,NULL);
  int size, rank;
  int n = 10;
  int a[n];
  int i;
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  if (rank == 0) {
    for(i=0;i<n;++i)
      a[i] = i;
    printf("Process 0 sending: ");
    for(int i=0;i<n;++i)
      printf("%d ",a[i]);
    printf("\n");
    MPI_Send(a,n,MPI_INT,1,0,MPI_COMM_WORLD);
  }
  else {
    MPI_Status status;
    MPI_Probe(0,0,MPI_COMM_WORLD,&status);
    int n;
    int a[n];
    MPI_Recv(a,n,MPI_INT,0,0,MPI_COMM_WORLD,&status);
    printf("Process 1 received: ");
    for(i=0;i<n;++i)
      printf("%d ",a[i]);
    printf("\n");
  }
  fflush(stdout);
  MPI_Finalize();
  return 0;
}
```

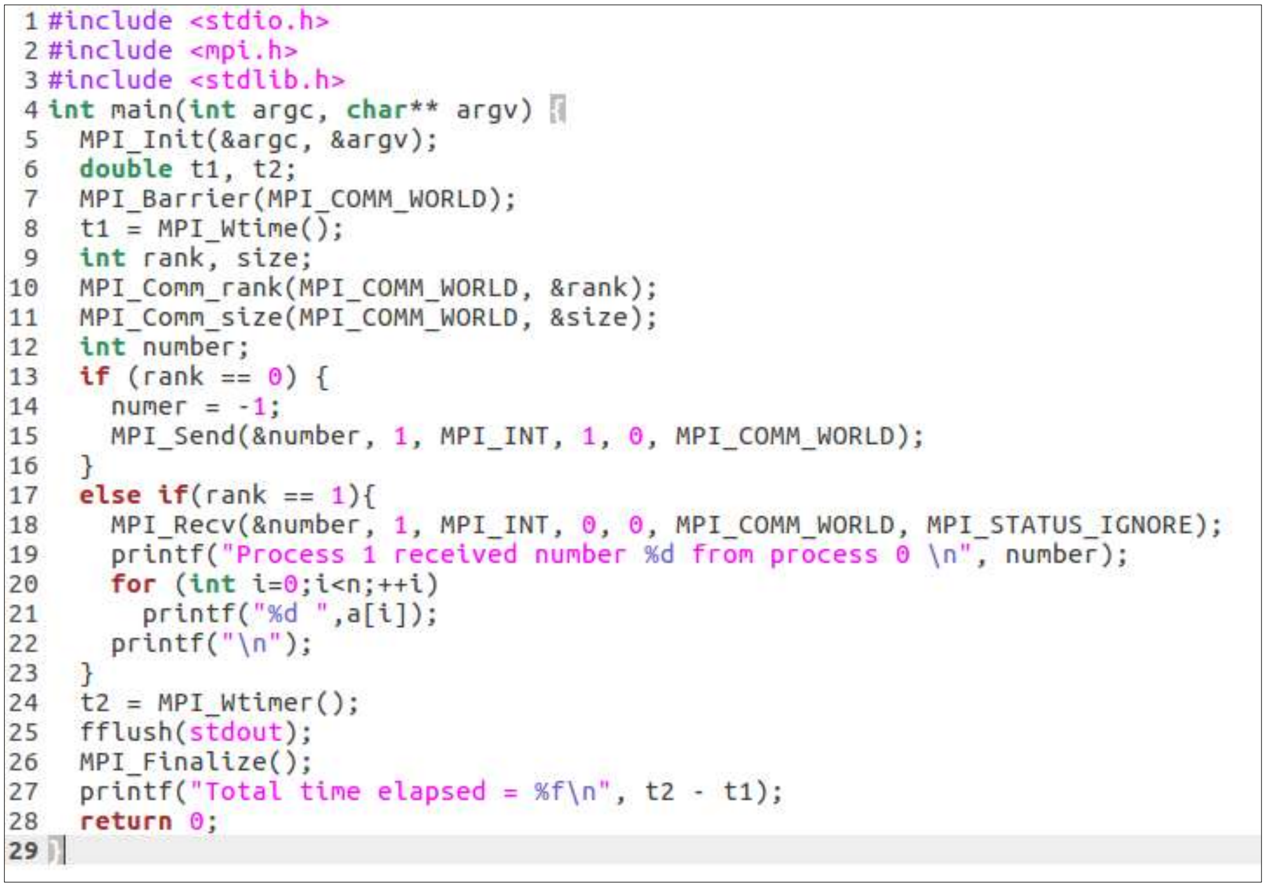
Output



```
Process 0 sending: 0 1 2 3 4 5 6 7 8 9
Process 1 received: 0 1 2 3 4 5 6 7 8 9
dhruv@dhruv-Inspiron-5559:~/PDC-lab/Lab 5/Problem 1$
```

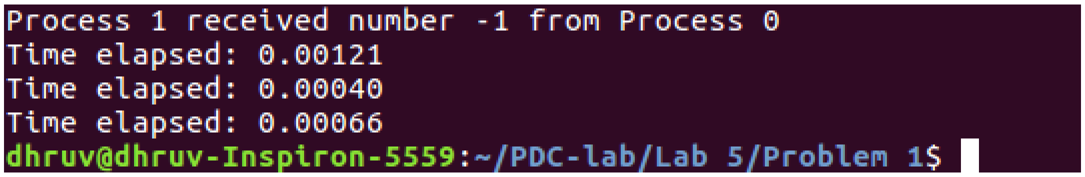
<u>Time taken for MPI_Send and MPI_Recieve</u>

<u>Code</u>



```c
#include <stdio.h>
#include <mpi.h>
#include <stdlib.h>
int main(int argc, char** argv) {
  MPI_Init(&argc, &argv);
  double t1, t2;
  MPI_Barrier(MPI_COMM_WORLD);
  t1 = MPI_Wtime();
  int rank, size;
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  int number;
  if (rank == 0) {
    numer = -1;
    MPI_Send(&number, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
  }
  else if(rank == 1){
    MPI_Recv(&number, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
    printf("Process 1 received number %d from process 0 \n", number);
    for (int i=0;i<n;++i)
      printf("%d ",a[i]);
    printf("\n");
  }
  t2 = MPI_Wtimer();
  fflush(stdout);
  MPI_Finalize();
  printf("Total time elapsed = %f\n", t2 - t1);
  return 0;
}
```

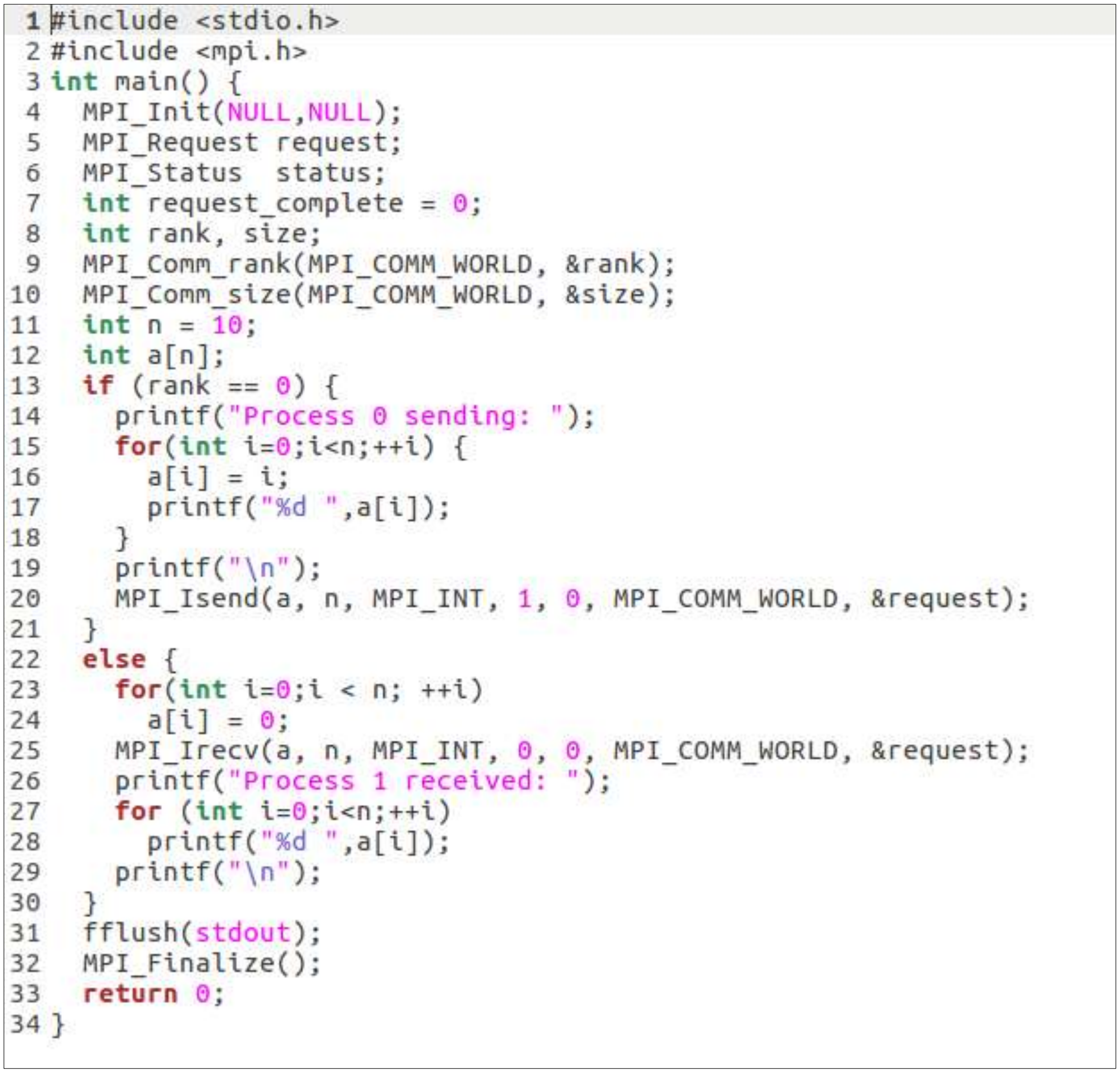
Output



```
Process 1 received number -1 from Process 0
Time elapsed: 0.00121
Time elapsed: 0.00040
Time elapsed: 0.00066
dhruv@dhruv-Inspiron-5559:~/PDC-lab/Lab 5/Problem 1$
```

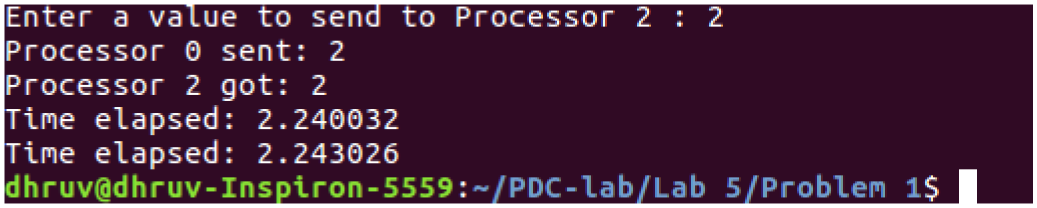
Time taken using MPI_ISend and MPI_IRecv

Code



```c
#include <stdio.h>
#include <mpi.h>
int main() {
  MPI_Init(NULL,NULL);
  MPI_Request request;
  MPI_Status  status;
  int request_complete = 0;
  int rank, size;
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
  MPI_Comm_size(MPI_COMM_WORLD, &size);
  int n = 10;
  int a[n];
  if (rank == 0) {
    printf("Process 0 sending: ");
    for(int i=0;i<n;++i) {
      a[i] = i;
      printf("%d ",a[i]);
    }
    printf("\n");
    MPI_Isend(a, n, MPI_INT, 1, 0, MPI_COMM_WORLD, &request);
  }
  else {
    for(int i=0;i < n; ++i)
      a[i] = 0;
    MPI_Irecv(a, n, MPI_INT, 0, 0, MPI_COMM_WORLD, &request);
    printf("Process 1 received: ");
    for (int i=0;i<n;++i)
      printf("%d ",a[i]);
    printf("\n");
  }
  fflush(stdout);
  MPI_Finalize();
  return 0;
}
```

Output



```
Enter a value to send to Processor 2 : 2
Processor 0 sent: 2
Processor 2 got: 2
Time elapsed: 2.240032
Time elapsed: 2.243026
dhruv@dhruv-Inspiron-5559:~/PDC-lab/Lab 5/Problem 1$
```