Use the following code to test your implementation. Interpret the test result in your report.

```
int main(int argc, char *argv[], char *environ[])
{

    int npes,myrank,number,i;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(&npes);
    MPI_Comm_rank(&myrank);

    if(myrank == 0)
    {
        for(i = 1; i < npes ; i++)
        {
            MPI_Recv(&number, 1, sizeof(int), i, 0);
            printf("From process %d  data= %d, RECEIVED!\n",i,number);
        }
    }
    else
    {
        MPI_Send(&myrank, 1, sizeof(int), 0, 0);
    }


    if(myrank == 0)
    {
        for(i = 1; i < npes ; i++)
        {
            MPI_Send(&i, 1, sizeof(int), i, 0);
        }
    }
    else
    {
        MPI_Recv(&number, 1, sizeof(int), 0, 0);
        printf("RECEIVED from %d data= %d, pid=%d \n",myrank,number,getpid());
    }



    for(i = 0; i < 100000 ; i++)
    {
        if(myrank%2 == 0)
        {
            MPI_Recv(&number, 1, sizeof(int), (myrank+1)%npes, 0);
            MPI_Send(&number, 1, sizeof(int), (myrank+1)%npes, 0);
        } else {
            MPI_Send(&number, 1, sizeof(int), (myrank-1+npes)%npes, 0);
            MPI_Recv(&number, 1, sizeof(int), (myrank-1+npes)%npes, 0);
        }
    }

    printf("FINISHED  %d\n",myrank);

    MPI_Finalize();
}
```

Use the following code to test your implementation. Interpret the test result in your report. This code is intended just for two processes.

```
int main(int argc, char *argv[], char *environ[])
{
    int npes,myrank,number,i;

    MPI_Init(&argc, &argv);
    MPI_Comm_size(&npes);
    MPI_Comm_rank(&myrank);

    if(myrank == 0)
    {

        MPI_Recv(&number, 1, sizeof(int), 1, 0);
        printf("RECEIVED!\n");
    }
    else
    {
        MPI_Send(&myrank, 1, sizeof(int), 0, 0);
        printf("SENT!\n");
    }

    printf("STAGE 2!\n");

    if(myrank == 0)
    {

        MPI_Recv(&number, 1, sizeof(int), 1, 1);
        printf("RECEIVED  >>> 2\n");
    }
    else
    {
        MPI_Send(&myrank, 1, sizeof(int), 0, 0);
        printf("SENT  >>> 2\n");
    }

    printf("FINISHED  %d\n",myrank);

    MPI_Finalize();
}
```