//Samantha Yee , Asad Siddiqui
Email:
asadsidd16@csu.fullerton.edu
samanthayee@csu.fullerton.edu

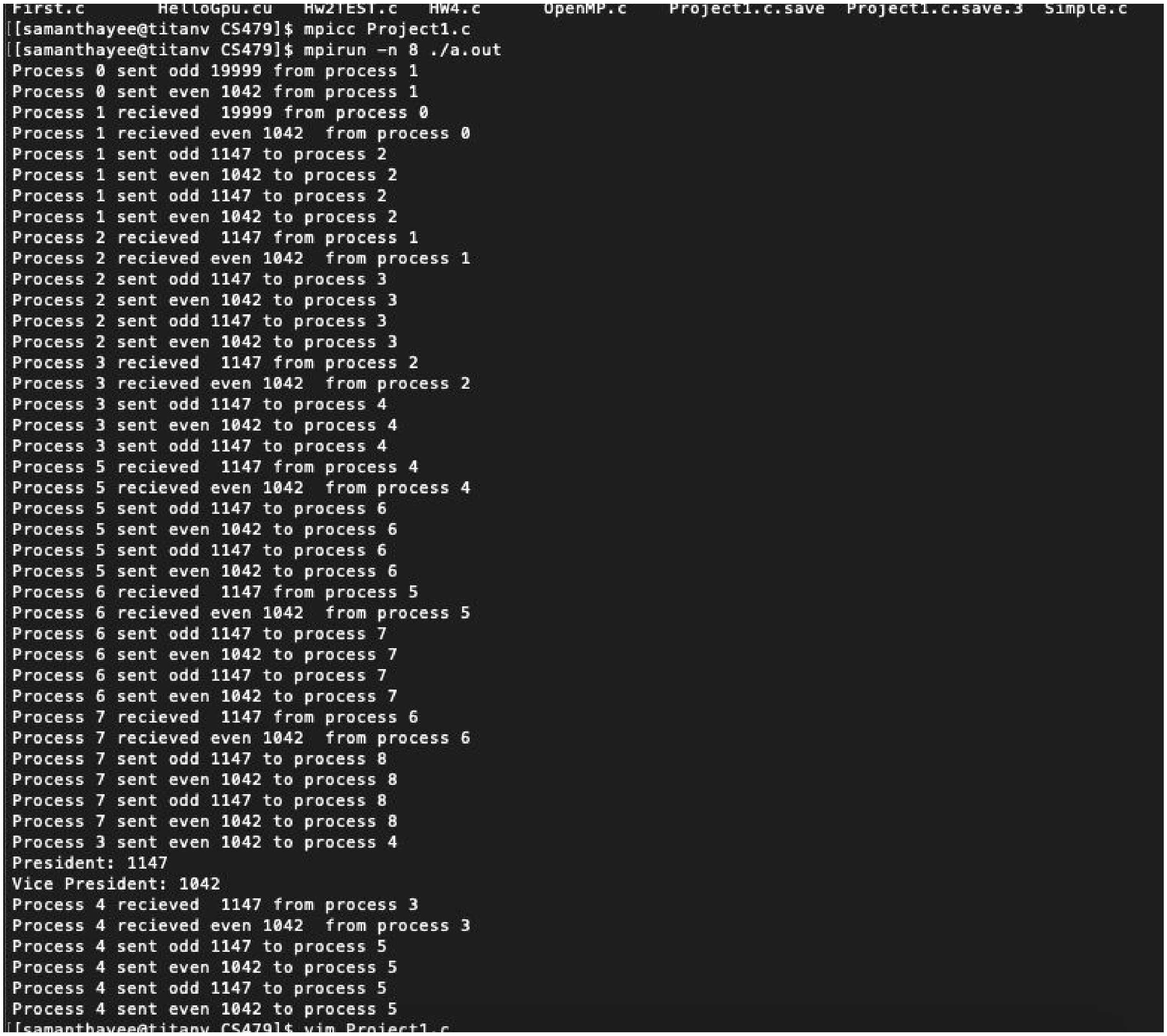
//Sequential two leader election algorithm
//sends 2 messages with 1 value each

To run:

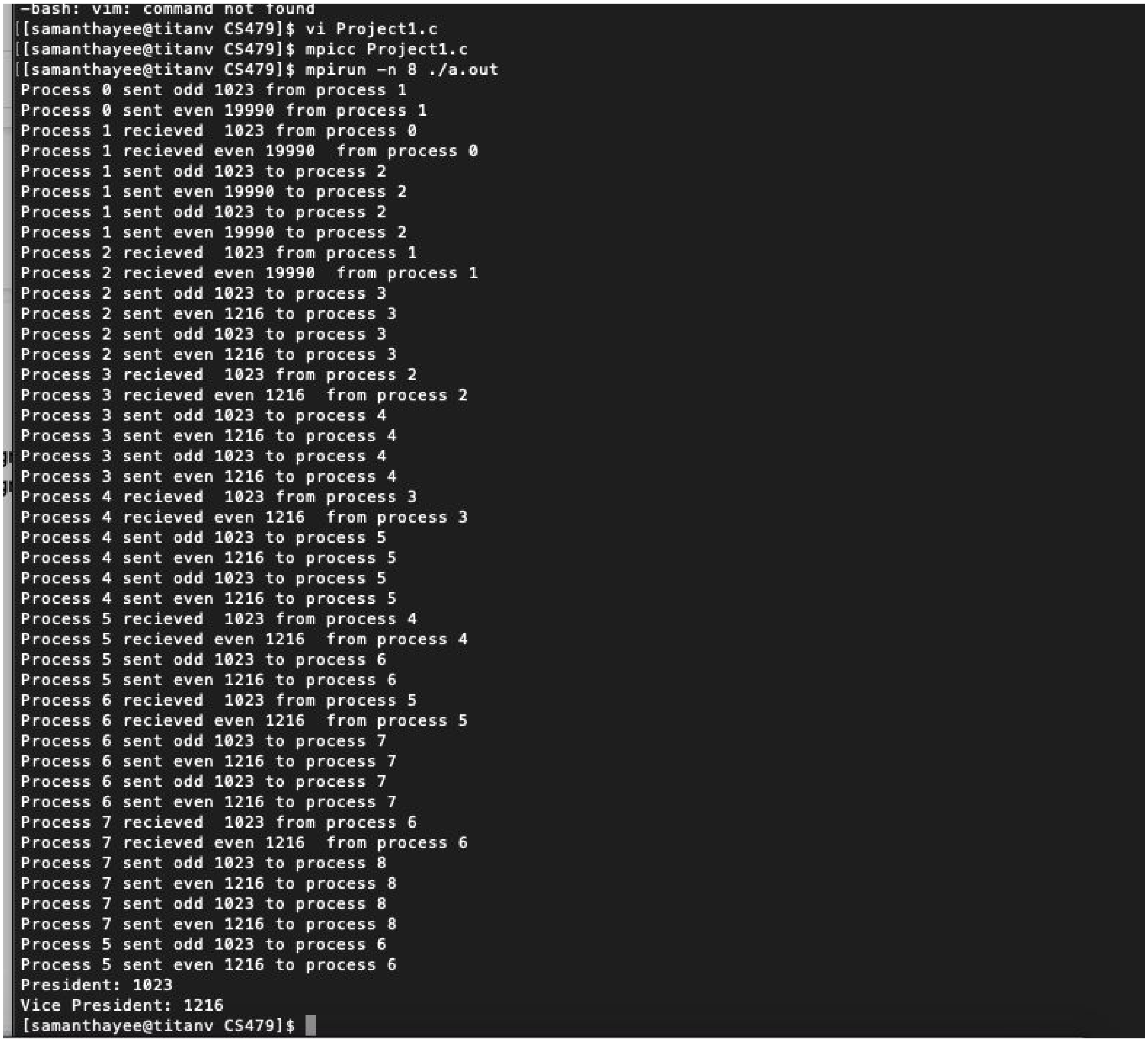
```
    mpicc [filename]
    mpirun -n 8 ./a.out
```

```
//Pseudocode
#include <stdio.h> <mpi.h><time.h>
#define N 8

int main(int argc, char*argv[]){
    //msg1 = odd number, msg2 = even number
    Int val, rank, size, msg 1, msg 2 (2 messages 1 value each), cont
    MPI_Init(&argc, &argv);
    MPI_COMM_rank(mpi_comm_world, &rank);
    MPI_COMM_size(mpi_comm_world, &size);

//IF
//If rank is equal to 0

If(rank==0){
//Process 0 creates a value
        val=rand();
        if val is < 0 : val = |val|;
        if val is greater than 100: val= val%100;
        if val is less than 10: temp=temp+10;
        cont= 1000+(100*rank)+val; //concatenate;

        //if value is odd
        if(val%2 !=0){
            msg1 = cont;
            msg2 = 19990;
            }
        //if value is even
        if(val%2 == 0){
            msg2 = cont;
            msg1 = 19999;
            }
    MPI_send(&msg1,1 , (rank+1)%2, 0,mpi_comm_world);
    printf(“Process %d sent odd %d to process%d\n”, rank, msg1,rank+1);
    MPI_send(&msg2,1, (rank+1)%2, 1,mpi_comm_world);
    printf(“Process %d sent even %d to process%d\n”, rank, msg1,rank+1);

else{
    MPI_recv(&msg1, 1, rank-1, 0, mpi_comm_world, mpi_status_ignore)
    printf("Process %d received odd %d  from process %d\n", rank, msg1, rank-1);
    MPI_recv(&msg2, 1, rank-1, 1 mpi_comm_world, mpi_status_ignore )
    printf("Process %d received even %d from process %d \n", rank, msg2, rank-1);

    val=rand();
    if val is < 0 : val = |val|;
    if val is greater than 100: val= val%100;
    if val is less than 10: temp=temp+10;
    cont= 1000+(100*rank)+val; //concatenate;

    //if value is odd
        if( msg1 > cont && cont%2!=0 ){
            msg1 = cont;
        }
    //if value is even
    if(cont%2 == 0 && msg2>cont){
            msg2 = cont;
        }
    //send msg1 &msg2 to the next process
    MPI_send(&msg1,1 , MPI_INT, (rank+1)%size, 0,mpi_comm_world);
    printf(“Process %d sent odd %d to process%d\n”, rank , msg1, rank+1);
    MPI_send(&msg2,1, MPI_INT, (rank+1)%size, 1,mpi_comm_world);
    printf(“Process %d sent even %d to process%d\n”, rank, msg2, rank+1);

    //send msg1 &msg2 to the next process
    MPI_send(&msg1,1 , (rank+1)%2, 0,mpi_comm_world);
    printf(“Process 0 sent odd %d to process%d\n”, rank, msg1, rank+1);
    MPI_send(&msg2,1, (rank+1)%2, 1,mpi_comm_world);
    printf(“Process 0 sent even %d to process%d\n”, rank, msg2, rank+1);
    }


//IF
//Process 0 receives msg1 from last process
//Process 0 receives msg2 from last process
```

```
If(rank==0){

MPI_recv(&msg1, 1, size-1, 0, mpi_comm_world, mpi_status_ignore);
MPI_recv(&msg2, 1, size-1, 1, mpi_comm_world, mpi_status_ignore );

printf(“President: %d , Vice President: %d \n”, msg1, msg2);

}

MPI_Finalize();
Return 0;
}
```



```
 First.c        HelloGpu.cu   Hw2TEST.c   HW4.c       OpenMP.c    Project1.c.save  Project1.c.save.3  Simple.c
[[samanthayee@titanv CS479]$ mpicc Project1.c
[[samanthayee@titanv CS479]$ mpirun -n 8 ./a.out
 Process 0 sent odd 19999 from process 1
 Process 0 sent even 1042 from process 1
 Process 1 recieved  19999 from process 0
 Process 1 recieved even 1042  from process 0
 Process 1 sent odd 1147 to process 2
 Process 1 sent even 1042 to process 2
 Process 1 sent odd 1147 to process 2
 Process 1 sent even 1042 to process 2
 Process 2 recieved  1147 from process 1
 Process 2 recieved even 1042  from process 1
 Process 2 sent odd 1147 to process 3
 Process 2 sent even 1042 to process 3
 Process 2 sent odd 1147 to process 3
 Process 2 sent even 1042 to process 3
 Process 3 recieved  1147 from process 2
 Process 3 recieved even 1042  from process 2
 Process 3 sent odd 1147 to process 4
 Process 3 sent even 1042 to process 4
 Process 3 sent odd 1147 to process 4
 Process 5 recieved  1147 from process 4
 Process 5 recieved even 1042  from process 4
 Process 5 sent odd 1147 to process 6
 Process 5 sent even 1042 to process 6
 Process 5 sent odd 1147 to process 6
 Process 5 sent even 1042 to process 6
 Process 6 recieved  1147 from process 5
 Process 6 recieved even 1042  from process 5
 Process 6 sent odd 1147 to process 7
 Process 6 sent even 1042 to process 7
 Process 6 sent odd 1147 to process 7
 Process 6 sent even 1042 to process 7
 Process 7 recieved  1147 from process 6
 Process 7 recieved even 1042  from process 6
 Process 7 sent odd 1147 to process 8
 Process 7 sent even 1042 to process 8
 Process 7 sent odd 1147 to process 8
 Process 7 sent even 1042 to process 8
 Process 3 sent even 1042 to process 4
 President: 1147
 Vice President: 1042
 Process 4 recieved  1147 from process 3
 Process 4 recieved even 1042  from process 3
 Process 4 sent odd 1147 to process 5
 Process 4 sent even 1042 to process 5
 Process 4 sent odd 1147 to process 5
 Process 4 sent even 1042 to process 5
[[samanthayee@titanv CS479]$ vim Project1.c
```

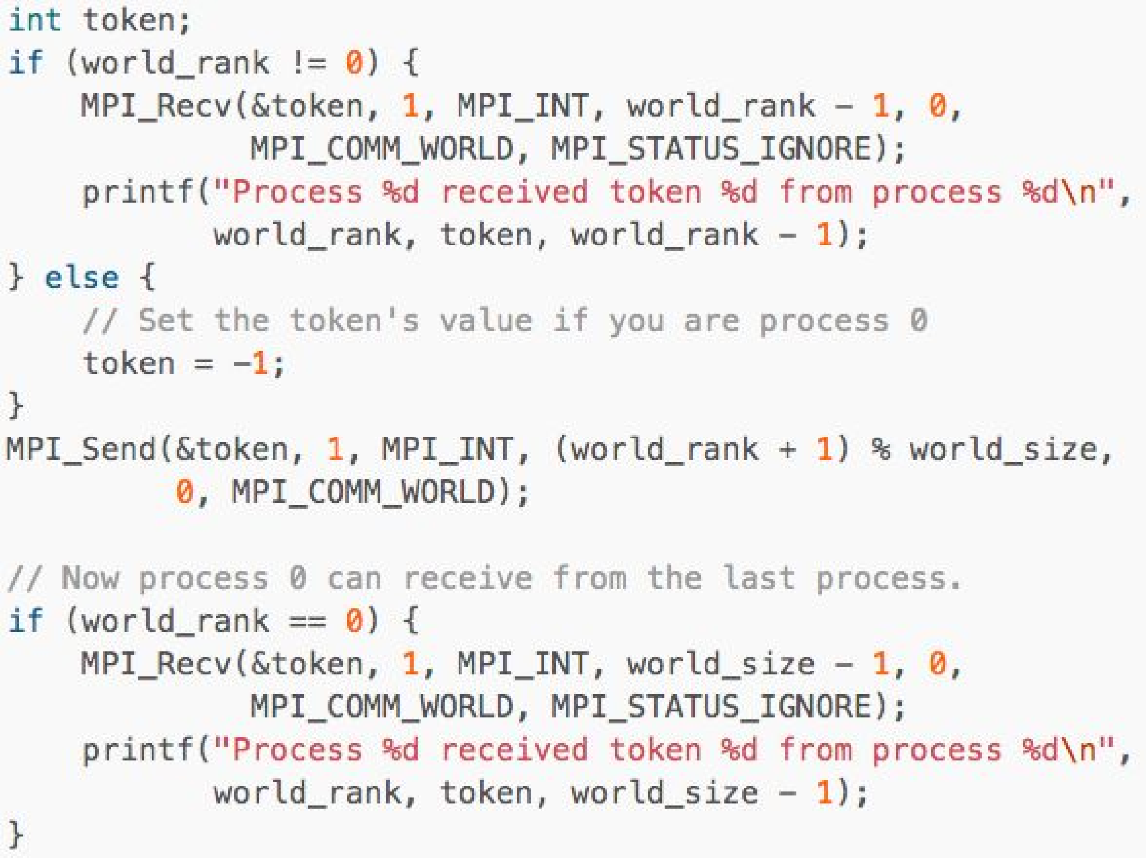


```
-bash: vim: command not found
[[samanthayee@titanv CS479]$ vi Project1.c
[[samanthayee@titanv CS479]$ mpicc Project1.c
[[samanthayee@titanv CS479]$ mpirun -n 8 ./a.out
Process 0 sent odd 1023 from process 1
Process 0 sent even 19990 from process 1
Process 1 recieved  1023 from process 0
Process 1 recieved even 19990  from process 0
Process 1 sent odd 1023 to process 2
Process 1 sent even 19990 to process 2
Process 1 sent odd 1023 to process 2
Process 1 sent even 19990 to process 2
Process 2 recieved  1023 from process 1
Process 2 recieved even 19990  from process 1
Process 2 sent odd 1023 to process 3
Process 2 sent even 1216 to process 3
Process 2 sent odd 1023 to process 3
Process 2 sent even 1216 to process 3
Process 3 recieved  1023 from process 2
Process 3 recieved even 1216  from process 2
Process 3 sent odd 1023 to process 4
Process 3 sent even 1216 to process 4
Process 3 sent odd 1023 to process 4
Process 3 sent even 1216 to process 4
Process 4 recieved  1023 from process 3
Process 4 recieved even 1216  from process 3
Process 4 sent odd 1023 to process 5
Process 4 sent even 1216 to process 5
Process 4 sent odd 1023 to process 5
Process 4 sent even 1216 to process 5
Process 5 recieved  1023 from process 4
Process 5 recieved even 1216  from process 4
Process 5 sent odd 1023 to process 6
Process 5 sent even 1216 to process 6
Process 6 recieved  1023 from process 5
Process 6 recieved even 1216  from process 5
Process 6 sent odd 1023 to process 7
Process 6 sent even 1216 to process 7
Process 6 sent odd 1023 to process 7
Process 6 sent even 1216 to process 7
Process 7 recieved  1023 from process 6
Process 7 recieved even 1216  from process 6
Process 7 sent odd 1023 to process 8
Process 7 sent even 1216 to process 8
Process 7 sent odd 1023 to process 8
Process 7 sent even 1216 to process 8
Process 5 sent odd 1023 to process 6
Process 5 sent even 1216 to process 6
President: 1023
Vice President: 1216
[samanthayee@titanv CS479]$
```

1. Process 0 creates a value
2. Determine if value is odd or even
    a. If value is odd → msg1 = value, msg2 = 19990
    b. Else value is even → msg2 = value, msg1 = 19999
    c. Send msg1 to next process
    d. Send msg2 to next process

3. Next process receives a msg1
4. Next process receives a msg2
5. Next process creates a value
6. Determine if value is odd or even
    a. If odd →
        i. Compare msg1 and created value
            1. If msg1 > value → msg1 = value
        ii. Send msg1 to next process
        iii. Send msg2 to next process
    b. Else even →
        i. Compare msg2 and created value
            1. If msg2 > value → msg2 = value
        ii. Send msg1 to next process
        iii. Send msg2 to next process

7. Process 0 receives msg1 from last process
8. Process 0 receives msg2 from last process



```c
int token;
if (world_rank != 0) {
    MPI_Recv(&token, 1, MPI_INT, world_rank - 1, 0,
             MPI_COMM_WORLD, MPI_STATUS_IGNORE);
    printf("Process %d received token %d from process %d\n",
           world_rank, token, world_rank - 1);
} else {
    // Set the token's value if you are process 0
    token = -1;
}
MPI_Send(&token, 1, MPI_INT, (world_rank + 1) % world_size,
         0, MPI_COMM_WORLD);

// Now process 0 can receive from the last process.
if (world_rank == 0) {
    MPI_Recv(&token, 1, MPI_INT, world_size - 1, 0,
             MPI_COMM_WORLD, MPI_STATUS_IGNORE);
    printf("Process %d received token %d from process %d\n",
           world_rank, token, world_size - 1);
}
```