Sahil Saini Salaria

Reg no 180905048

Roll No. 11 C

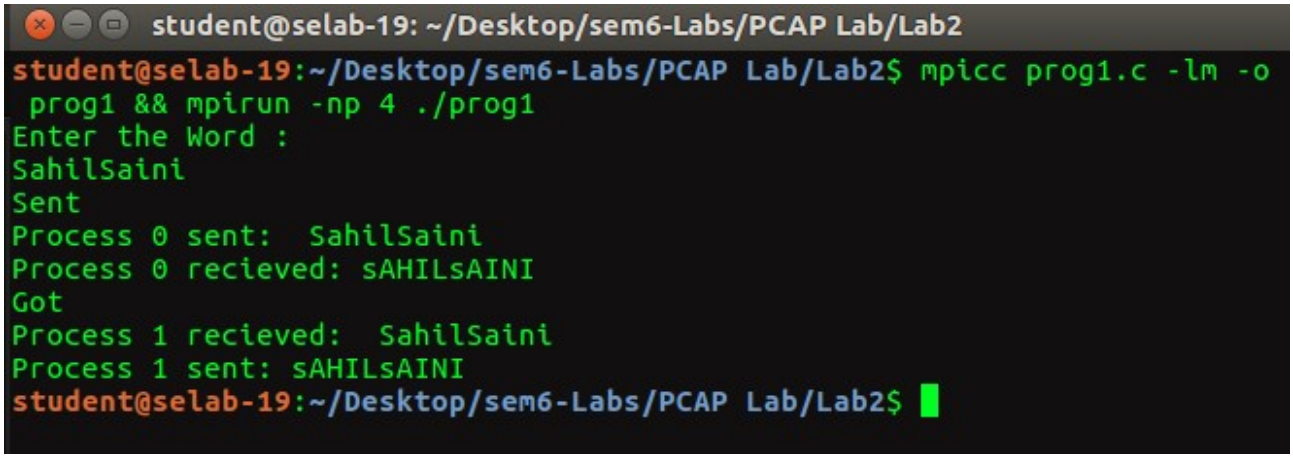
Q1

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

int main(int argc, char *argv[])
{
   int rank, size, n;
   MPI_Status status;
   MPI_Init(&argc, &argv);
   MPI_Comm_size(MPI_COMM_WORLD, &size);
   MPI_Comm_rank(MPI_COMM_WORLD, &rank);

   if (rank == 0)
   {
      char a[25];

      fprintf(stdout, "Enter the Word : \n");
      fflush(stdout);
      scanf("%s", a);

      n = strlen(a);
```

```
    MPI_Ssend(&n, 1, MPI_INT, 1, 0, MPI_COMM_WORLD);
    printf("Sent\n");
    MPI_Ssend(a, n+1, MPI_CHAR, 1, 1, MPI_COMM_WORLD);
    fprintf(stdout, "Process %d sent:  %s\n", rank, a);
    fflush(stdout);
    MPI_Recv(a, n+1, MPI_CHAR, 1, 2, MPI_COMM_WORLD, &status);
    fprintf(stdout, "Process %d recieved: %s\n", rank, a);
    fflush(stdout);
}
else if (rank == 1)
{
    char b[25];

    MPI_Recv(&n, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, &status);
    printf("Got\n");
    MPI_Recv(b, n+1, MPI_CHAR, 0, 1, MPI_COMM_WORLD, &status);
    fprintf(stdout, "Process %d recieved:  %s\n", rank, b);
    fflush(stdout);

    for (int i = 0; i < n; i++)
    {
        if (b[i] >= 'a' && b[i] <= 'z')
        {
            b[i] = b[i] - 32;
        }
        else
        {
            b[i] = b[i] + 32;
        }
    }
```

```
        MPI_Ssend(b, n+1, MPI_CHAR, 0, 2, MPI_COMM_WORLD);
        fprintf(stdout, "Process %d sent: %s\n", rank, b);
        fflush(stdout);
    }
    MPI_Finalize();
    return 0;
}
```

mpicc prog1.c -lm -o prog1 && mpirun -np 4 ./prog1

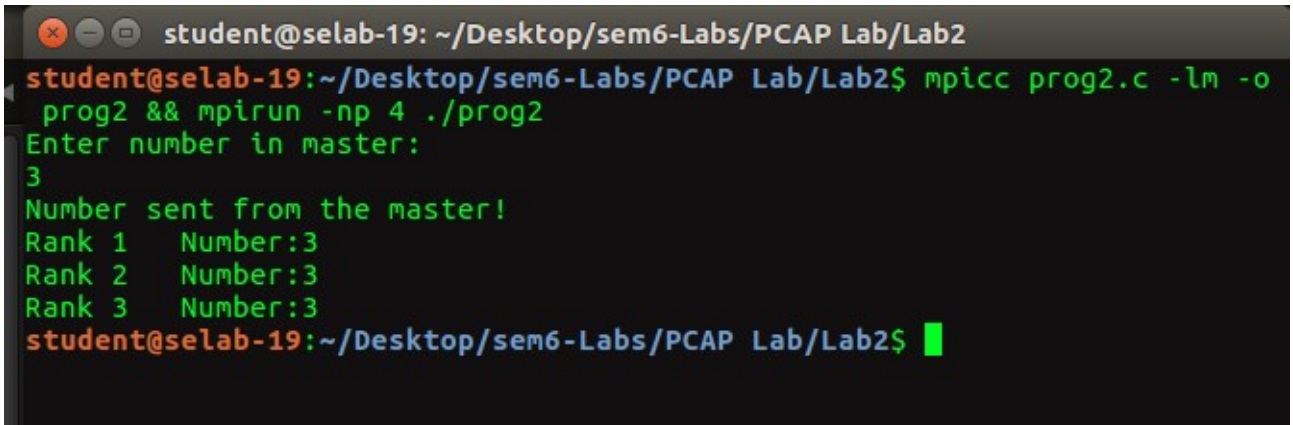


Q2

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

int main(int argc, char *argv[])
{
    int rank, size;

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);
```

```
    MPI_Status status;
    int ele;

    if (rank == 0)
    {
        printf("Enter number in master:\n");
        scanf("%d",&ele);

        printf("Number sent from the master!\n");
        for (int i = 1; i < size; i++)
        {
            MPI_Send(&ele, 1, MPI_INT, i, i, MPI_COMM_WORLD);
        }
    }
    else
    {
        int ele2;
        MPI_Recv(&ele2, 1, MPI_INT, 0, rank, MPI_COMM_WORLD, &status);
        printf("Rank %d\t Number:%d \n", rank, ele2);
    }

    MPI_Finalize();

    return 0;
}
```

// mpicc prog2.c -lm -o prog2 && mpirun -np 4 ./prog2



Q3

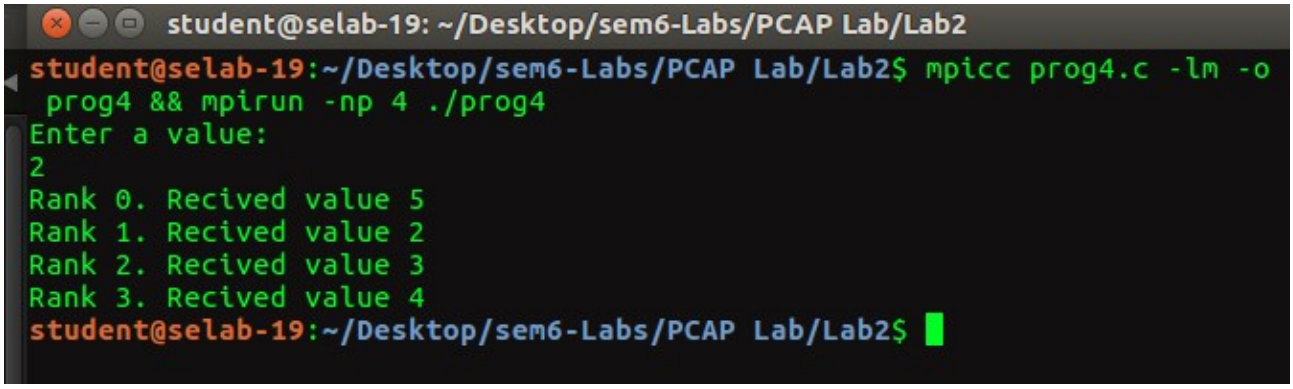
```
#include <stdio.h>
#include <mpi.h>

int main(int argc, char *argv[])
{
   int rank, size;

   MPI_Init(&argc, &argv);
   MPI_Comm_rank(MPI_COMM_WORLD, &rank);
   MPI_Comm_size(MPI_COMM_WORLD, &size);

   MPI_Status status;

   char buff[100];
   int n = 100l;
   MPI_Buffer_attach(buff, n);
   if (rank == 0)
   {
      int arr[100];

      printf("Enter %d numbers:\n", size);
      scanf("%d", &arr[0]);
      printf("Rank 0: Square of %d is :%d\n", arr[0], arr[0] * arr[0]);

      for (int i = 1; i < size; i++)
      {
         scanf("%d", &arr[i]);
         MPI_Bsend(&arr[i], 1, MPI_INT, i, i, MPI_COMM_WORLD);
      }
   }
   else
   {
      int ele;
      MPI_Recv(&ele, 1, MPI_INT, 0, rank, MPI_COMM_WORLD, &status);

      if (rank % 2 == 0)
         printf("Rank %d: Square of %d is :%d\n", rank, ele, ele*ele);
      else
         printf("Rank %d: Cube of %d is :%d\n", rank, ele,ele* ele* ele);
   }

   MPI_Buffer_detach(buff, &n);
   MPI_Finalize();

   return 0;
}

//  mpicc prog3.c -lm -o prog3 && mpirun -np 4 ./prog3
```

```
student@selab-19: ~/Desktop/sem6-Labs/PCAP Lab/Lab2
student@selab-19:~/Desktop/sem6-Labs/PCAP Lab/Lab2$ mpicc prog3.c -lm -o
 prog3 && mpirun -np 4 ./prog3
Enter 4 numbers:
1 2 3 4
Rank 0: Square of 1 is :1
Rank 2: Square of 3 is :9
Rank 3: Cube of 4 is :64
Rank 1: Cube of 2 is :8
student@selab-19:~/Desktop/sem6-Labs/PCAP Lab/Lab2$
```

Q4

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

int main(int argc, char *argv[])
{
   int rank, size;

   MPI_Init(&argc, &argv);
   MPI_Comm_rank(MPI_COMM_WORLD, &rank);
   MPI_Comm_size(MPI_COMM_WORLD, &size);

   MPI_Status status;

   if (rank == 0)
   {
      int ele;
      printf("Enter a value:\n");
      scanf("%d", &ele);

      MPI_Send(&ele, 1, MPI_INT, 1, 1, MPI_COMM_WORLD);

      MPI_Recv(&ele, 1, MPI_INT, size-1, 1, MPI_COMM_WORLD, &status);
      printf("Rank 0. Recived value %d \n", ele);

   }
   else
   {
      int ele;
      MPI_Recv(&ele, 1, MPI_INT, rank - 1, 1, MPI_COMM_WORLD, &status);
      printf("Rank %d. Recived value %d \n", rank, ele);
      ele += 1;
      if (rank < size - 1)
      {
         MPI_Send(&ele, 1, MPI_INT, rank + 1, 1, MPI_COMM_WORLD);
      }
      else
```

```
        MPI_Send(&ele, 1, MPI_INT, 0, 1, MPI_COMM_WORLD);
    }

    MPI_Finalize();

    return 0;
}

//   mpicc prog4.c -lm -o prog4 && mpirun -np 4 ./prog4
```



```
student@selab-19:~/Desktop/sem6-Labs/PCAP Lab/Lab2$ mpicc prog4.c -lm -o
 prog4 && mpirun -np 4 ./prog4
Enter a value:
2
Rank 0. Recived value 5
Rank 1. Recived value 2
Rank 2. Recived value 3
Rank 3. Recived value 4
student@selab-19:~/Desktop/sem6-Labs/PCAP Lab/Lab2$ 
```

//

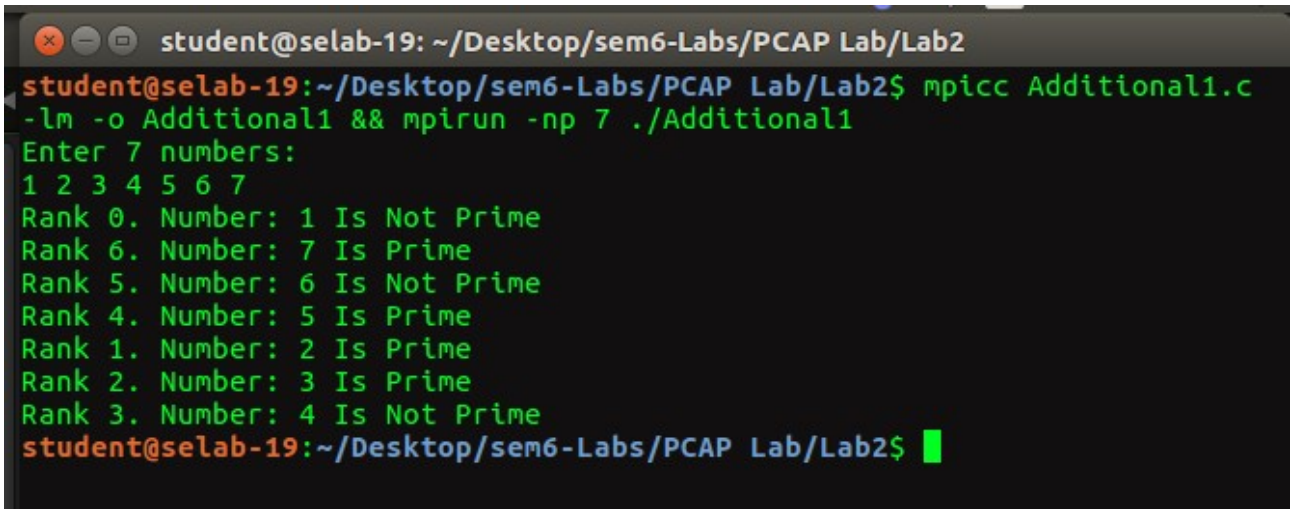
Additional:

Additional 1.

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

int isPrime(int num)
{
    if(num==1)
        return 0;
```

```
    for (int i = 2; i*i <= num; i++)
    {
        if(num%i==0)
        return 0;
    }
    return 1;
}

int main(int argc, char *argv[])
{
    int rank, size;

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    MPI_Comm_size(MPI_COMM_WORLD, &size);

    MPI_Status status;

    char buff[100];
    int n = 100l;
    if (rank == 0)
    {
        int arr[100];

        printf("Enter %d numbers:\n", size);
        scanf("%d", &arr[0]);

        if (isPrime(arr[0]))
        {
            printf("Rank %d. Number: %d Is Prime\n",rank,arr[0]);
        }
        else
            printf("Rank %d. Number: %d Is Not Prime\n",rank,arr[0]);

        for (int i = 1; i < size; i++)
        {
            scanf("%d", &arr[i]);
            MPI_Send(&arr[i], 1, MPI_INT, i, i, MPI_COMM_WORLD);
        }
    }
    else
    {
        int ele;
        MPI_Recv(&ele, 1, MPI_INT, 0, rank, MPI_COMM_WORLD, &status);
        if (isPrime(ele))
        {
            printf("Rank %d. Number: %d Is Prime\n",rank,ele);
        }
        else
            printf("Rank %d. Number: %d Is Not Prime\n",rank,ele);

    }
```

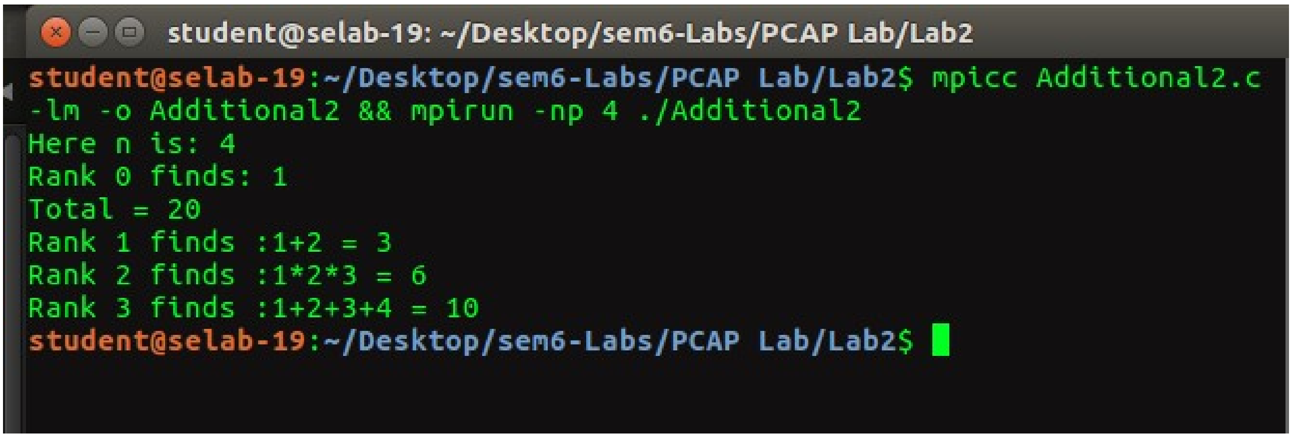
```
    MPI_Finalize();

    return 0;
}

//  mpicc Additional1.c -lm -o Additional1 && mpirun -np 4 ./Additional1
```



Additional 2.

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


int main(int argc, char *argv[])
{
        int rank,size;
        MPI_Init(&argc,&argv);
        MPI_Comm_rank(MPI_COMM_WORLD,&rank);
        MPI_Comm_size(MPI_COMM_WORLD,&size);

        MPI_Status status;

        if(rank==0)
        {
                int n;
                long long int sum=0;
                printf("Here n is: %d\n",size);

                printf("Rank 0 finds: 1\n");
                sum=sum+1;

                for (int i = 1; i < size; i++)
```

```
		{
			int ele;

	MPI_Recv(&ele,1,MPI_INT,MPI_ANY_SOURCE,1,MPI_COMM_WORLD,&status);
			sum+=ele;
		}
		printf("Total = %lld\n",sum );

	}
	else
	{
		long long int sumTemp=1;
		printf("Rank %d finds :1",rank);
		if(rank%2==1)
		{

			for (int i = 2; i <=rank+1; i++)
			{
				sumTemp+=i;
				printf("+%d",i);
			}
		}
		else{

			for (int i = 2; i <=rank+1; i++)
			{
				sumTemp*=i;
				printf("*%d",i);

			}
		}

		printf(" = %lld\n",sumTemp );
		MPI_Send(&sumTemp,1,MPI_INT,0,1,MPI_COMM_WORLD);

	}

	MPI_Finalize();
	return 0;

	// mpicc Additional2.c -lm -o Additional2 && mpirun -np 4 ./Additional2
}
```



```
student@selab-19:~/Desktop/sem6-Labs/PCAP Lab/Lab2$ mpicc Additional2.c
-lm -o Additional2 && mpirun -np 4 ./Additional2
Here n is: 4
Rank 0 finds: 1
Total = 20
Rank 1 finds :1+2 = 3
Rank 2 finds :1*2*3 = 6
Rank 3 finds :1+2+3+4 = 10
student@selab-19:~/Desktop/sem6-Labs/PCAP Lab/Lab2$ 
```

//