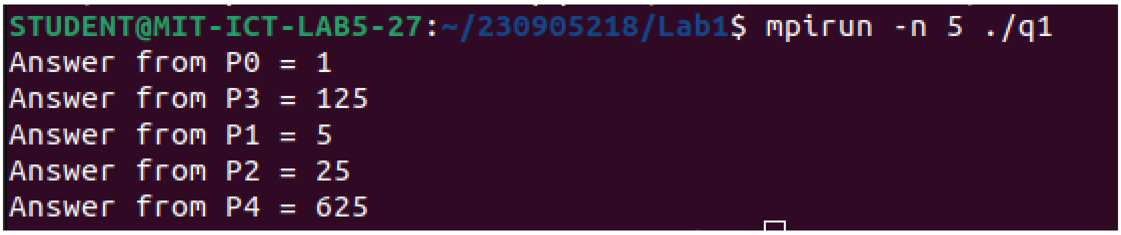
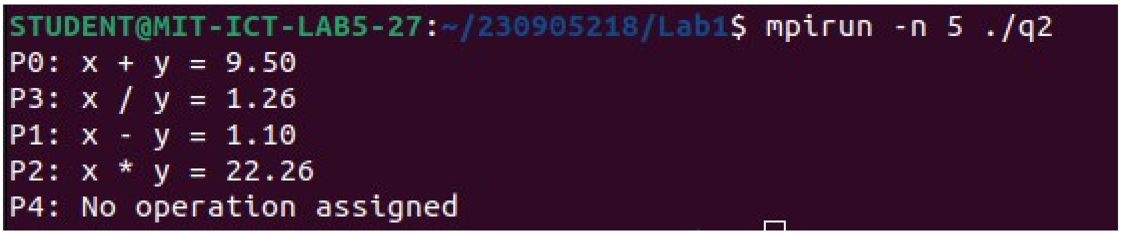
Lab 1: Introduction to execution environment of MPI

```
//Title : MPI program that calculate powers
//Author : Aditya Sinha
//Date : 9/1/2026

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

int mypow(int x, int y);

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

	int ans = mypow(5, rank);
	printf("Answer from P%d = %d\n", rank, ans);

	MPI_Finalize();
	return 0;
}

int mypow(int x, int y){
	if(y==0) return 1;
	if((y&1) == 1) return x*mypow(x, y-1);
	else return mypow(x*x, y/2);
}
```



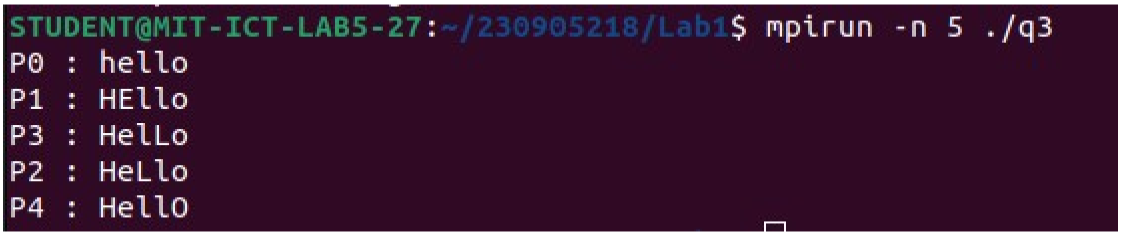
```
STUDENT@MIT-ICT-LAB5-27:~/230905218/Lab1$ mpirun -n 5 ./q1
Answer from P0 = 1
Answer from P3 = 125
Answer from P1 = 5
Answer from P2 = 25
Answer from P4 = 625
```

```
//Title : Calculator with MPI
//Author : Aditya Sinha
//Date : 9/1/2026

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

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

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

    double x = 5.3;
    double y = 4.2;

    switch (rank) {
        case 0:
            printf("P%d: x + y = %.2f\n", rank, x + y);
            break;

        case 1:
            printf("P%d: x - y = %.2f\n", rank, x - y);
            break;

        case 2:
            printf("P%d: x * y = %.2f\n", rank, x * y);
            break;

        case 3:
            printf("P%d: x / y = %.2f\n", rank, x / y);
            break;

        default:
            printf("P%d: No operation assigned\n", rank);
    }

    MPI_Finalize();
    return 0;
}
```



```
STUDENT@MIT-ICT-LAB5-27:~/230905218/Lab1$ mpirun -n 5 ./q2
P0: x + y = 9.50
P3: x / y = 1.26
P1: x - y = 1.10
P2: x * y = 22.26
P4: No operation assigned
```

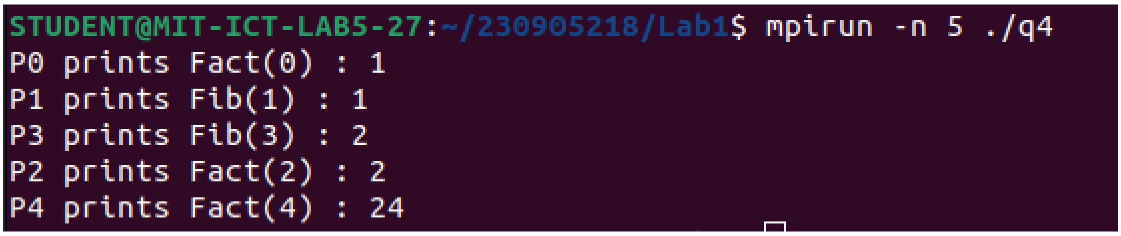
```
//Title : MPI program that toggle Characters
//Author : Aditya Sinha
//Date : 9/1/2026

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

char* toggle(char* s, int i);

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

    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    char s[] = "Hello";

    printf("P%d : %s\n", rank, toggle(s, rank));

    MPI_Finalize();
    return 0;
}

char* toggle(char* s, int i) {
    if (i >= strlen(s))
        return s;

    if (s[i] >= 'a' && s[i] <= 'z')
        s[i] = s[i] - 'a' + 'A';
    else if (s[i] >= 'A' && s[i] <= 'Z')
        s[i] = s[i] - 'A' + 'a';

    return s;
}
```



```
STUDENT@MIT-ICT-LAB5-27:~/230905218/Lab1$ mpirun -n 5 ./q3
P0 : hello
P1 : HEllo
P3 : HelLo
P2 : HeLlo
P4 : HellO
```

```
//Title : MPI program that calculate factorial and fibonacci of the rank if even or odd respectively
//Author : Aditya Sinha
//Date : 9/1/2026

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

int fact(int n);
int fib(int n);

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

        if((rank & 1) == 1)
                printf("P%d prints Fib(%d) : %d\n", rank, rank, fib(rank));
        else
                printf("P%d prints Fact(%d) : %d\n", rank, rank, fact(rank));

        MPI_Finalize();
```

```
	return 0;
}

int fact(int n){
	return n <= 1 ? 1 : n*fact(n-1);
}

int fib(int n){
	return n <= 1 ? n : fib(n-1)+fib(n-2);
}
```



```
STUDENT@MIT-ICT-LAB5-27:~/230905218/Lab1$ mpirun -n 5 ./q4
P0 prints Fact(0) : 1
P1 prints Fib(1) : 1
P3 prints Fib(3) : 2
P2 prints Fact(2) : 2
P4 prints Fact(4) : 24
```