

Introduction to execution environment of MPI

Name : Satyam Singh

Reg No: 230905256

Roll no: 37

CSE B2

Lab Exercises:

Q1) Write a simple MPI program to find out $\text{pow}(x, \text{rank})$ for all processes where 'x' is the integer constant and 'rank' is the rank of the process. Write a program in MPI where even ranked process prints "Hello" and odd ranked process prints "World".

Ans)

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

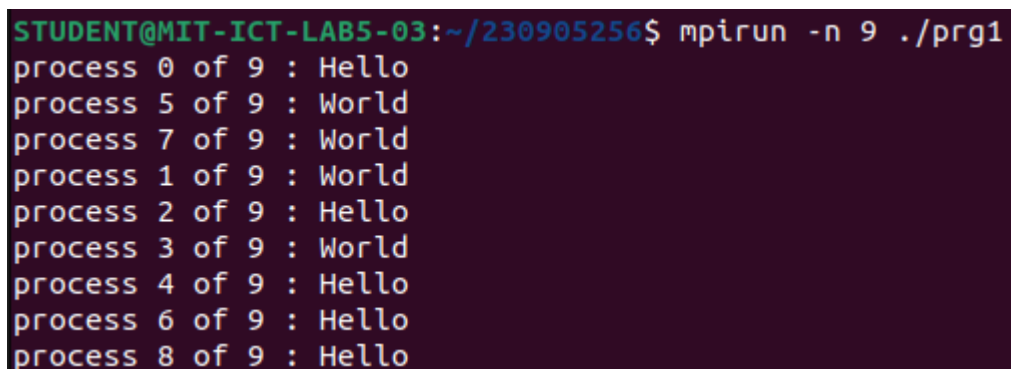
```
#include<mpi.h>
```

```
#include<ctype.h>
```

```
#include<math.h>
```

```
int main(int argc, char *argv[]){
    int rank, size, n=4;
    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    int res = pow(n,rank);
    if(rank%2==0){
        printf("process %d of %d : Hello \n",rank,size);
    }else{
        printf("process %d of %d : World \n",rank,size);
    }
    MPI_Finalize();
    return 0;
}
```

Output:

A terminal window with a dark purple background. The prompt is 'STUDENT@MIT-ICT-LABS-03:~/230905256\$'. The command 'mpirun -n 9 ./prg1' has been executed. The output shows 9 processes, each printing a message. Processes with even ranks (0, 2, 4, 6, 8) print 'Hello', and processes with odd ranks (1, 3, 5, 7) print 'World'.

```
STUDENT@MIT-ICT-LABS-03:~/230905256$ mpirun -n 9 ./prg1
process 0 of 9 : Hello
process 5 of 9 : World
process 7 of 9 : World
process 1 of 9 : World
process 2 of 9 : Hello
process 3 of 9 : World
process 4 of 9 : Hello
process 6 of 9 : Hello
process 8 of 9 : Hello
```

Q2) Write a program in MPI to simulate simple calculator. Perform each operation using different process in parallel.

Ans)

```

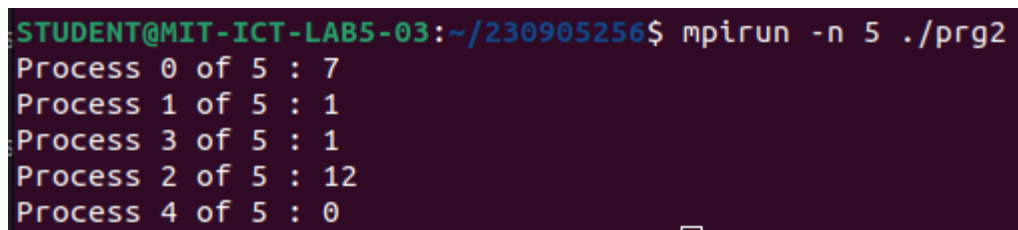
#include<stdio.h>
#include<mpi.h>
#include<ctype.h>
#include<math.h>

int main(int argc, char *argv[]){
    int rank, size , n=4;
    int ch, a =4,b = 3, res;
    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    switch(rank){
    case 0 : res = a + b;
        printf("Process %d of %d : %d \n",rank,size,res);
        break;
    case 1 : res = a - b;
        printf("Process %d of %d : %d \n",rank,size,res);
        break;
    case 2 : res = a * b;
        printf("Process %d of %d : %d \n",rank,size,res);
        break;
    case 3 : res = a / b;
        printf("Process %d of %d : %d \n",rank,size,res);
        break;
    default : res = 0;
        printf("Process %d of %d : %d \n",rank,size,res);
    }

    MPI_Finalize();
    return 0;
}

```

Output:



```

STUDENT@MIT-ICT-LAB5-03:~/230905256$ mpirun -n 5 ./prg2
Process 0 of 5 : 7
Process 1 of 5 : 1
Process 3 of 5 : 1
Process 2 of 5 : 12
Process 4 of 5 : 0

```

Q3) Write a program in MPI to toggle the character of a given string indexed by the rank of the process . Hint : suppose the string is HELLO and there are 5 processes, then process 0 toggle 'H' to 'h', process 1 toggle 'E' to 'e' and so on .

Ans)

```

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

```

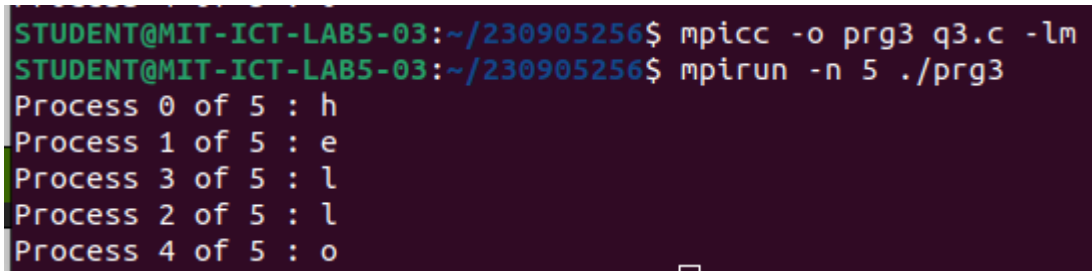
```

#include<math.h>
#include<string.h>

int main(int argc, char *argv[]){
    int rank, size, n=4;
    char s[] = "HELLO";
    MPI_Init(&argc,&argv);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    if(rank<strlen(s)){
        s[rank] = tolower(s[rank]);
        printf("Process %d of %d : %c \n",rank, size, s[rank]);
    }
    MPI_Finalize();
    return 0;
}

```

Output:



```

STUDENT@MIT-ICT-LAB5-03:~/230905256$ mpicc -o prg3 q3.c -lm
STUDENT@MIT-ICT-LAB5-03:~/230905256$ mpirun -n 5 ./prg3
Process 0 of 5 : h
Process 1 of 5 : e
Process 3 of 5 : l
Process 2 of 5 : l
Process 4 of 5 : o

```

Q4) Write a program in MPI where even ranked process prints factorial of the rank and odd ranked process prints ranks Fibonacci number .

Ans)

```

#include<stdio.h>
#include<mpi.h>
#include<ctype.h>
#include<math.h>
int fact(int a){
    int res = 1;
    while(a!=0){
        res = res * a;
        a--;
    }
    return res;
}
int fib(int a){
    if(a==2 || a == 1) return 1;
    return fib(a-1) + fib(a-2);
}
int main(int argc, char *argv[]){

```

```

int rank, size, n=4;
MPI_Init(&argc,&argv);
MPI_Comm_size(MPI_COMM_WORLD,&size);
MPI_Comm_rank(MPI_COMM_WORLD,&rank);
if(rank%2 == 0){
    printf("Process %d of %d : %d \n",rank, size, fact(rank));
}else{
    printf("Process %d of %d : %d \n",rank, size, fib(rank));
}
MPI_Finalize();
return 0;
}

```

Output:

```

STUDENT@MIT-ICT-LAB5-03:~/230905256$ mpicc -o prg4 q4.c -lm
STUDENT@MIT-ICT-LAB5-03:~/230905256$ mpirun -n 9 ./prg4
Process 3 of 9 : 2
Process 7 of 9 : 13
Process 0 of 9 : 1
Process 1 of 9 : 1
Process 4 of 9 : 24
Process 5 of 9 : 5
Process 8 of 9 : 40320
Process 2 of 9 : 2
Process 6 of 9 : 720

```