

LAB 3  
HRITIK AJAY BANSAL  
CSE A 15  
180905105

Q1)#include<mpi.h>

#include<stdio.h>

#include<string.h>

```
int facto(int c){
    int t=1;int flag=1;
    for (int i = 1; i <=c; ++i)
    {
        t*=i;
    }
    return t;
}
int main(int argc, char const *argv[])
{
    int fact;
    int x;
    int rank;int size;
    int root=0;int flag=1;

    MPI_Init(NULL,NULL);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    int a[size],b[size];
    if(rank==root){
        printf("enter %d no of elemnts\n",size );
        for (int i = 0; i < size; ++i)
        {
            scanf("%d",&a[i]);
        }
    }
    MPI_Scatter(a,1,MPI_INT,&x,1,MPI_INT,0,MPI_COMM_WORLD);

    fact=facto(x);
    printf("process %d: factorial of %d: %d\n",rank,x,fact );

    MPI_Gather(&fact,1,MPI_INT,b,1,MPI_INT,0,MPI_COMM_WORLD);

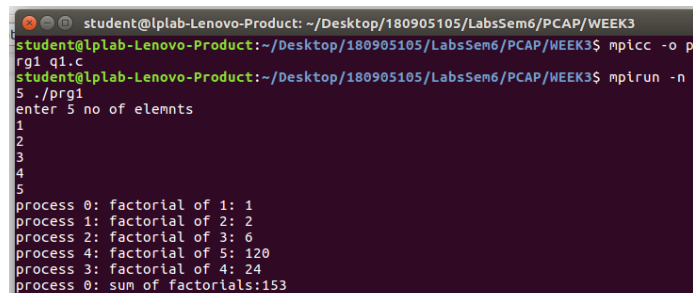
    if (rank==0)
    {
        int recv=0;
        for (int i = 0; i < size; ++i)
        {
            recv+=b[i];
        }
        printf("process 0: sum of factorials:%d\n",recv );
    }
}
```

```

    MPI_Finalize();
    return 0;
}

```

OUTPUT:



```

student@lplab-Lenovo-Product: ~/Desktop/180905105/LabsSem6/PCAP/WEEK3
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpicc -o p
rg1 q1.c
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpirun -n
5 ./prg1
enter 5 no of elemts
1
2
3
4
5
process 0: factorial of 1: 1
process 1: factorial of 2: 2
process 2: factorial of 3: 6
process 4: factorial of 5: 120
process 3: factorial of 4: 24
process 0: sum of factorials:153

```

```

Q2)#include<mpi.h>
#include<stdio.h>
#include<string.h>

```

```

double avrg(int* b,int m){
    double sum=0,avv;
    for (int i = 0; i < m; ++i)
    {
        sum+=b[i];
    }
    avv=sum/m;
    return avv;
}
int main(int argc, char const *argv[])
{
    int rank;int size;
    int root=0;int flag=1;

    MPI_Init(NULL,NULL);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    MPI_Comm_size(MPI_COMM_WORLD,&size);\
    int n=size;
    int m;
    int *a;
    double avr[n];
    if (rank==root)
    {
        printf("enter the value of m\n");
        scanf("%d",&m);
        a=malloc(m*n*(sizeof(int)));
        printf("enter %d elements\n",n*m);
        for (int i = 0; i < n*m; ++i)
        {
            scanf("%d",&a[i]);
        }
    }
}

```

```

MPI_Bcast(&m,1,MPI_INT,0,MPI_COMM_WORLD);
int *b;
b=malloc(m*sizeof(int));

MPI_Scatter(a,m,MPI_INT,b,m,MPI_INT,0,MPI_COMM_WORLD);
double avg=avrg(b,m);
printf("process %d: average of my m nos is: %f\n",rank,avg);

MPI_Gather(&avg,1,MPI_DOUBLE,avr,1,MPI_DOUBLE,0,MPI_COMM_WORLD);
if (rank==0)
{double sum=0;
    for (int i = 0; i < n; ++i)
    {
        sum+=avr[i];
    }
    double avg1=((sum)/n);
    printf("process 0: average of all nos is: %f\n",avg1);
}

MPI_Finalize();
return 0;
}

```

output:

```

student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpicc -o prg2 q2.c
q2.c: In function 'main':
q2.c:30:5: warning: implicit declaration of function 'malloc' [-Wimplicit-function-declaration]
    a=malloc(m*n*(sizeof(int)));
    ^
q2.c:30:5: warning: incompatible implicit declaration of built-in function 'malloc'
q2.c:30:5: note: include '<stdlib.h>' or provide a declaration of 'malloc'
q2.c:41:4: warning: incompatible implicit declaration of built-in function 'malloc'
    b=malloc(m*sizeof(int));
    ^
q2.c:41:4: note: include '<stdlib.h>' or provide a declaration of 'malloc'
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpirun -n 3 ./prg2
enter the value of m
3
enter 9 elements
1
2
3
4
5
6
7
8
9
process 0: average of my m nos is: 2.000000
process 0: average of all nos is: 5.000000
process 1: average of my m nos is: 5.000000
process 2: average of my m nos is: 8.000000
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$

```

Q3)#include<mpi.h>

#include<stdio.h>

#include<string.h>

```

int non_count(char* b){
    int non=0;
    for (int i = 0; i < strlen(b); ++i)
    {
        if(isalpha(b[i]))
        if (b[i]!='a'&&b[i]!='e'&&b[i]!='i'&&b[i]!='o'&&b[i]!='u'&&b[i])
        {
            non++;
        }
    }
}

```

```

        return non;
    }
int main(int argc, char const *argv[])
{
    int fact;
    int x;
    int rank;int size;
    int root=0;int flag=1;

    MPI_Init(NULL,NULL);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    char a[100],b[100];
    int c[size];
    int m;
    if(rank==root){
        printf("enter a string having multiple of %d characters\n",size );
        gets(a);
        m=(strlen(a)/size);

    }
    MPI_Bcast(&m,1,MPI_INT,0,MPI_COMM_WORLD);

    MPI_Scatter(a,m,MPI_CHAR,&b,m,MPI_CHAR,0,MPI_COMM_WORLD);

    int nonc=non_count(b);
    printf("process %d: non-vowel alphabet count is: %d\n",rank,nonc );

    MPI_Gather(&nonc,1,MPI_INT,c,1,MPI_INT,0,MPI_COMM_WORLD);

    if (rank==0)
    {
        int recv=0;
        for (int i = 0; i < size; ++i)
        {
            recv+=c[i];
        }
        printf("process %d: final non-vowel alphabet count is: %d\n",rank,recv );
    }
    MPI_Finalize();
    return 0;
}
output:

```

```

student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpicc -o prg3 q3.c
q3.c: In function 'non_count':
q3.c:9:16: warning: implicit declaration of function 'isalpha' [-Wimplicit-function-declaration]
    if(isalpha(b[i]))
       ^
q3.c: In function 'main':
q3.c:32:3: warning: implicit declaration of function 'gets' [-Wimplicit-function-declaration]
    gets(a);
    ^
/tmp/ccYz0tr0.o: In function 'main':
q3.c:(.text+0x205): warning: the 'gets' function is dangerous and should not be used.
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpirun -n 3 ./prg3
enter a string having multiple of 3 characters
hritik is my
process 0: non-vowel alphabet count is: 3
process 0: final non-vowel alphabet count is: 7
process 1: non-vowel alphabet count is: 1
process 2: non-vowel alphabet count is: 3
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$

```

```

Q4)#include<mpi.h>
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
char* mixed(char* d, char*f){
    char* xx=(char *)calloc(100,sizeof(char));
    int j=0,k=0;
    for (int i = 0; i < strlen(d)*2,j<strlen(d),k<strlen(d); ++i)
    {
        if (i%2==0)
        {
            xx[i]=d[j++];
        }
        else
            xx[i]=f[k++];
    }
    xx[2*strlen(d)]='\0';

    return xx;
}
int main(int argc, char const *argv[])
{
    int fact;
    int x;
    int rank;int size;
    int root=0;int flag=1;

    MPI_Init(NULL,NULL);
    MPI_Comm_rank(MPI_COMM_WORLD,&rank);
    MPI_Comm_size(MPI_COMM_WORLD,&size);
    char a[100],b[100];
    char a1[100],b1[100];

    char c[100];
    int m;int n=0;
    if(rank==root){
        printf("enter 2 strings having multiple of %d characters\n",size );
        gets(a);
        gets(a1);
        m=(strlen(a)/size);
        n=2*strlen(a);
    }
}

```

```

}
MPI_Bcast(&m,1,MPI_INT,0,MPI_COMM_WORLD);

MPI_Scatter(a,m,MPI_CHAR,b,m,MPI_CHAR,0,MPI_COMM_WORLD);
MPI_Scatter(a1,m,MPI_CHAR,b1,m,MPI_CHAR,0,MPI_COMM_WORLD);

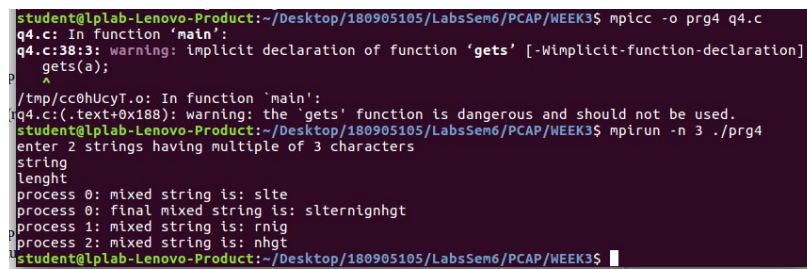
char* tt=(char *)calloc(100,sizeof(char));
strcpy(tt,mixed(b,b1));
printf("process %d: mixed string is: %s\n",rank,tt);

MPI_Gather(tt,2*m,MPI_CHAR,c,2*m,MPI_CHAR,0,MPI_COMM_WORLD);

if (rank==0)
{
    c[n]='\0';
    printf("process %d: final mixed string is: %s\n",rank,c);
}

MPI_Finalize();
return 0;
}
output:

```



```

student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpicc -o prg4 q4.c
q4.c: In function 'main':
q4.c:38:3: warning: implicit declaration of function 'gets' [-Wimplicit-function-declaration]
    gets(a);
    ^
/tmp/cc0hUcyT.o: In function 'main':
q4.c:(.text+0x188): warning: the 'gets' function is dangerous and should not be used.
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$ mpirun -n 3 ./prg4
enter 2 strings having multiple of 3 characters
string
length
process 0: mixed string is: slte
process 0: final mixed string is: slternighgt
process 1: mixed string is: rnig
process 2: mixed string is: nhgt
student@lplab-Lenovo-Product:~/Desktop/180905105/LabsSem6/PCAP/WEEK3$

```