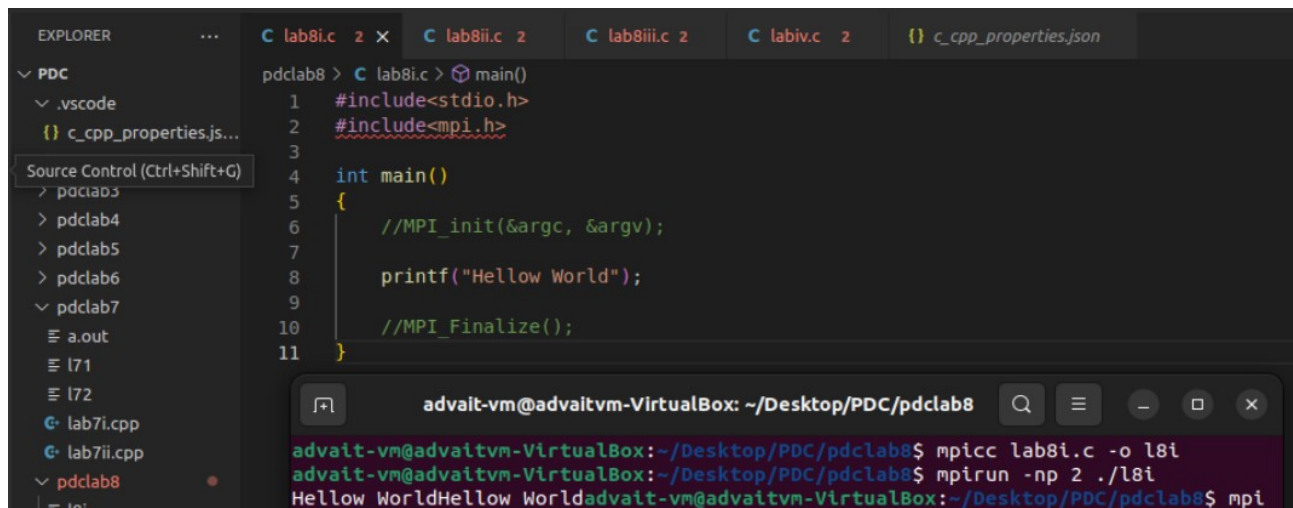


PDC Lab 8

Advait Deochakke

20BCE1143

Sample Hello World



The screenshot shows a Visual Studio Code editor with a project named 'PDC'. The Explorer sidebar on the left shows a file tree with folders 'pdclab3' through 'pdclab8' and files 'a.out', 'l71', 'l72', 'lab7i.cpp', 'lab7ii.cpp', and 'lab8i.c'. The main editor window displays the code for 'lab8i.c', which is a C program that includes `<stdio.h>` and `<mpi.h>`, and contains a `main()` function that initializes MPI, prints 'Hellow World', and finalizes MPI. Below the editor, a terminal window is open, showing the commands `mpicc lab8i.c -o l8i` and `mpirun -np 2 ./l8i`, followed by the output 'Hellow WorldHellow World'.

```
EXPLORER
PDC
  .vscode
  {} c_cpp_properties.js...
Source Control (Ctrl+Shift+G)
  > pdclab3
  > pdclab4
  > pdclab5
  > pdclab6
  > pdclab7
  > pdclab8
    a.out
    l71
    l72
    lab7i.cpp
    lab7ii.cpp
    pdclab8
      l8i

lab8i.c
1  #include<stdio.h>
2  #include<mpi.h>
3
4  int main()
5  {
6      //MPI_init(&argc, &argv);
7
8      printf("Hellow World");
9
10     //MPI_Finalize();
11 }
```

```
advait-vm@advaitvm-VirtualBox: ~/Desktop/PDC/pdclab8
advait-vm@advaitvm-VirtualBox:~/Desktop/PDC/pdclab8$ mpicc lab8i.c -o l8i
advait-vm@advaitvm-VirtualBox:~/Desktop/PDC/pdclab8$ mpirun -np 2 ./l8i
Hellow WorldHellow Worldadvait-vm@advaitvm-VirtualBox:~/Desktop/PDC/pdclab8$ mpi
```

Code:

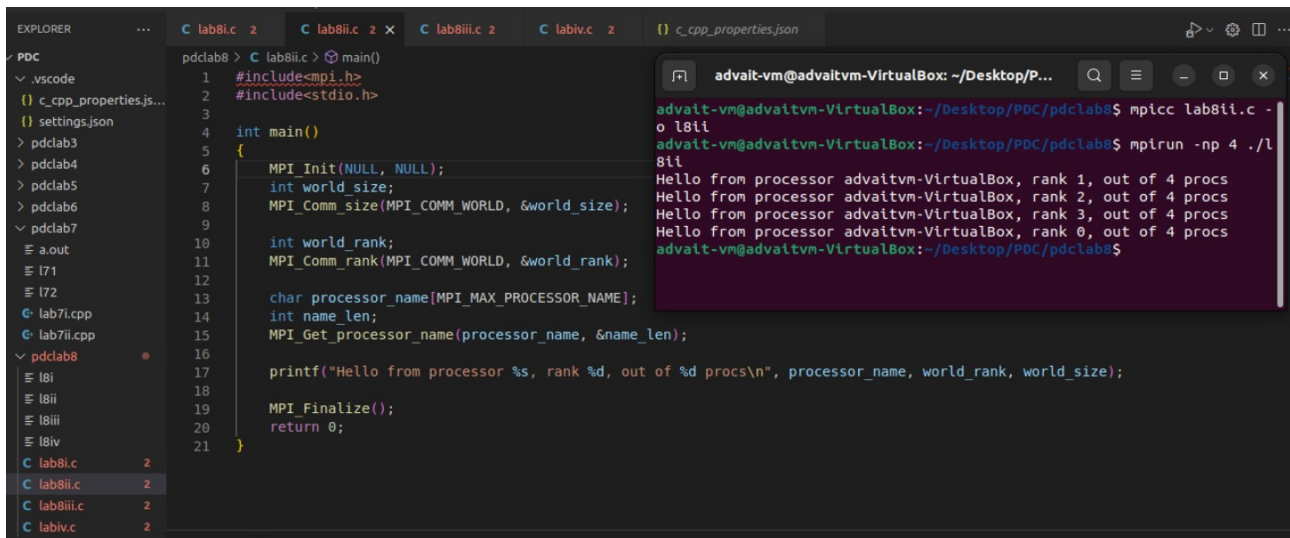
```
#include<stdio.h>
#include<mpi.h>

int main()
{
    //MPI_init(&argc, &argv);

    printf("Hellow World");

    //MPI_Finalize();
}
```

Print rank, world size and processor name



```
EXPLORER  ...  C lab8i.c 2  C lab8ii.c 2 x  C lab8iii.c 2  C labiv.c 2  {} c_cpp_properties.json

PDC
└─ .vscode
  {} c_cpp_properties.js...
  {} settings.json
  > pdclab3
  > pdclab4
  > pdclab5
  > pdclab6
  > pdclab7
  └─ a.out
  └─ l71
  └─ l72
  └─ lab7i.cpp
  └─ lab7ii.cpp
  > pdclab8
  └─ l8i
  └─ l8ii
  └─ l8iii
  └─ l8iv
  C lab8i.c 2
  C lab8ii.c 2
  C lab8iii.c 2
  C labiv.c 2

pdclab8 > C lab8ii.c > main()
1 #include<mpi.h>
2 #include<stdio.h>
3
4 int main()
5 {
6     MPI_Init(NULL, NULL);
7     int world_size;
8     MPI_Comm_size(MPI_COMM_WORLD, &world_size);
9
10    int world_rank;
11    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
12
13    char processor_name[MPI_MAX_PROCESSOR_NAME];
14    int name_len;
15    MPI_Get_processor_name(processor_name, &name_len);
16
17    printf("Hello from processor %s, rank %d, out of %d procs\n", processor_name, world_rank, world_size);
18
19    MPI_Finalize();
20    return 0;
21 }
```

```
advait-vm@advaitvm-VirtualBox: ~/Desktop/P...
advait-vm@advaitvm-VirtualBox:~/Desktop/PDC/pdclab8$ mpicc lab8ii.c -o l8ii
advait-vm@advaitvm-VirtualBox:~/Desktop/PDC/pdclab8$ mpirun -np 4 ./l8ii
Hello from processor advaitvm-VirtualBox, rank 1, out of 4 procs
Hello from processor advaitvm-VirtualBox, rank 2, out of 4 procs
Hello from processor advaitvm-VirtualBox, rank 3, out of 4 procs
Hello from processor advaitvm-VirtualBox, rank 0, out of 4 procs
advait-vm@advaitvm-VirtualBox:~/Desktop/PDC/pdclab8$
```

Code:

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

```
int main()
{
    MPI_Init(NULL, NULL);
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);
```

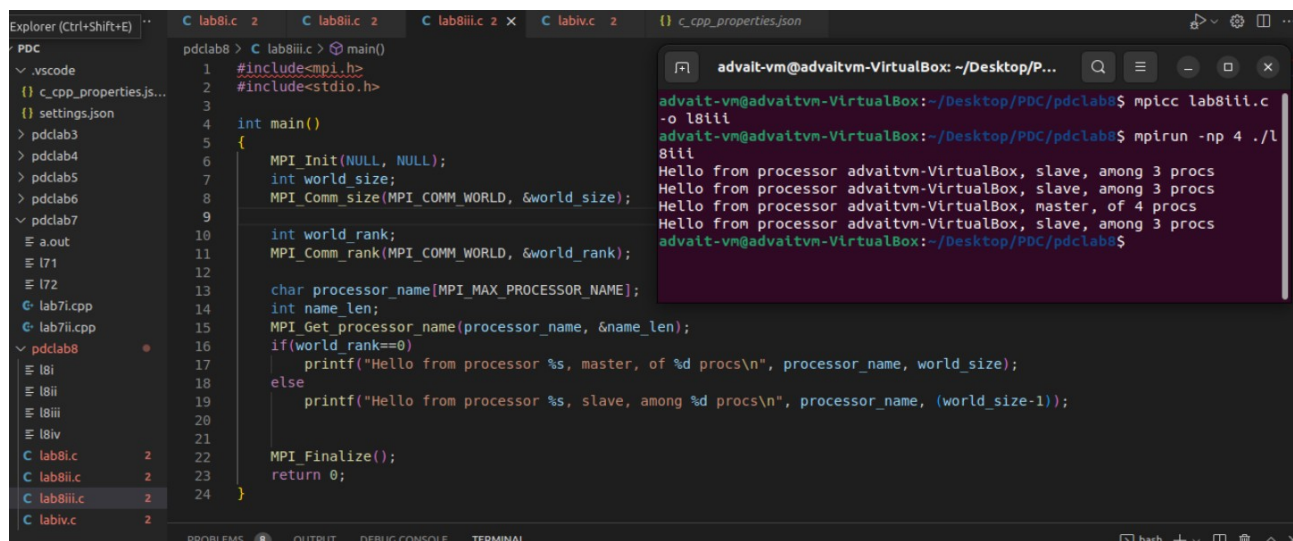
```
    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
```

```
    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);
```

```
    printf("Hello from processor %s, rank %d, out of %d procs\n", processor_name, world_rank, world_size);

    MPI_Finalize();
    return 0;
}
```

Master prints "I am Master", Worker prints "I am worker"



The screenshot shows a VS Code editor with a C file named `lab8iii.c` open. The code is an MPI program that prints the rank of each processor. The terminal window shows the command `mpicc lab8iii.c -o lab8iii` and `mpirun -np 4 ./lab8iii`, which produces the following output:

```
Hello from processor advaitvm-VirtualBox, slave, among 3 procs
Hello from processor advaitvm-VirtualBox, slave, among 3 procs
Hello from processor advaitvm-VirtualBox, master, of 4 procs
Hello from processor advaitvm-VirtualBox, slave, among 3 procs
```

Code:

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

int main()
{
    MPI_Init(NULL, NULL);
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);

    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);
    if(world_rank==0)
        printf("Hello from processor %s, master, of %d procs\n",
processor_name, world_size);
    else
        printf("Hello from processor %s, slave, among %d procs\n",
processor_name, (world_size-1));

    MPI_Finalize();
}
```

```
return 0;
}
```

Master generates 1/2,1/4,1/8,1/16...1/n; Worker generates 2,4,8,16...n

The screenshot shows a VS Code editor with a C file named `lab8iv.c` open. The code is as follows:

```
1 #include<mpi.h>
2 #include<omp.h>
3 #include<stdio.h>
4 #include<math.h>
5
6 int main()
7 {
8     MPI_Init(NULL, NULL);
9     int world_size;
10    MPI_Comm_size(MPI_COMM_WORLD, &world_size);
11
12    int world_rank;
13    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);
14
15    char processor_name[MPI_MAX_PROCESSOR_NAME];
16    int name_len;
17    MPI_Get_processor_name(processor_name, &name_len);
18    for(int i=0; i<10; i++)
19    {
20        if(world_rank==0)
21            printf("Master Output : 1/%d\n", (int)(pow(2, i)));
22        else
23            printf("Worker Output : %d\n", (int)(pow(2, i)));
24    }
25}
```

The terminal window shows the command `mpi run -np 2 ./l8iv` being executed, resulting in the following output:

```
Worker Output : 1
Worker Output : 2
Worker Output : 4
Worker Output : 8
Worker Output : 16
Worker Output : 32
Worker Output : 64
Worker Output : 128
Worker Output : 256
Worker Output : 512
Master Output : 1/1
Master Output : 1/2
Master Output : 1/4
Master Output : 1/8
Master Output : 1/16
Master Output : 1/32
Master Output : 1/64
Master Output : 1/128
Master Output : 1/256
Master Output : 1/512
```

Code:

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

int main()
{
    MPI_Init(NULL, NULL);
    int world_size;
    MPI_Comm_size(MPI_COMM_WORLD, &world_size);

    int world_rank;
    MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

    char processor_name[MPI_MAX_PROCESSOR_NAME];
    int name_len;
    MPI_Get_processor_name(processor_name, &name_len);
    for(int i=0; i<10; i++)
    {
        if(world_rank==0)
            printf("Master Output : 1/%d\n", (int)(pow(2, i)));
        else
            printf("Worker Output : %d\n", (int)(pow(2, i)));
    }
}
```

```
}
```

```
    MPI_Finalize();
```

```
    return 0;
```

```
}
```
