
Introduction to High-Performance Computing **Exercise**

Giorgio Amati

Corso di dottorato in Ingegneria Aeronautica e Spaziale 2024

g.amati@cineca.it / g.amaticode@gmail.com

Agenda

- ✓ Simple “warm-up” exercise: Matrix-Matrix Multiplication
- ✓ Complete the code
 - Fortran/C
- ✓ Check the results
- ✓ Extract some Performance figure (in MFLOPs)

You can use:

- ✓ Any available HW
 - ✓ Any available Compiler
 - ✓ Any compiler option
-

How to do

✓ Clone the repository

- `git clone https://github.com/gamati01/HPCLessons.git`

```
.
├── ESER1
│   ├── clean.sh
│   ├── compile.c.sh
│   ├── compile.fortran.sh
│   ├── EXERCISE1.pdf
│   ├── inc_precision.h
│   ├── mm.c
│   ├── mm.F90
│   ├── mod_tools.F90
│   └── README
├── LESSON1
│   ├── HPC-1.pdf
│   └── HPC-1-spoiler.pdf
└── README.md
```

✓ Complete the code

```
70 ! main loop
71 call system("date      > time.log")
72 call timing(time1)
73 !
74 ! write bottom here 3 nested loops....
75 !
76           c(i,j) = c(i,j) + a(i,k)*b(k,j)
77
78 call timing(time2)
79 call system("date      >> time.log")
```

compile.fortran.sh

- ✓ simple script to compile the code
- ✓ choose the available compiler & compiler options

```
rm -rf *.o mm.x *.mod
#
...
#
# gfortran (GNU)
COMP=gfortran
OPT=
#
echo "compiling with " $COMP $OPT
#
$COMP $OPT mod_tools.F90 -c
$COMP $OPT mm.F90 -c
$COMP $OPT mod_tools.o mm.o -o mm.x
#
echo "That's all folks!!!"
```

- ✓ If correctly code it should give an output like that

```
-----  
Matrix-Matrix Multiplication  
precision used          15  
rel. 0, naive multiplication  
Which matrix size?  
1024  
Matrix size      =      1024  
Memory size (MB) =      24  
-----  
initialization  1.1718750000000000E-002  
0.4293334354359644      0.9410485065499756      0.0000000000000000  
-----  
CPU:time for multiplication  3.1406250000000000  
CPU:MFLOPS      683.7758861940298  
CPU:check      257.1789318419338
```

- ✓ size (e.g. 1024) given by standard input
- ✓ check ~ size/4
-

✓ Complete the code

```
45  time1 = clock();
46  /*                                     */
47  /* write here 3 nested loop */
48  /*                                     */
49          c[i][j] = c[i][j] + a[i][k]*b[k][j];
50
51
52  time2 = clock();
```

compile.c.sh

- ✓ simple script to compile the code
- ✓ choose the available compiler & compiler options

```
rm -rf *.o mm.x *.mod
#
...
#
# gcc (GNU)
COMP=gcc
OPT=
#
echo "compiling with " $COMP $OPT
#
$COMP $OPT mm.c -c
$COMP $OPT mm.o -o mm.x
#
echo "That's all folks!!!"
```


- ✓ If correctly code it should give an output like that

```
=====
double precision
size 1024
=====
Initialization
Elapsed time for initialization
Total time -----> 0.020759
=====
Tme -----> 2.280420
Mflops -----> 941.705321
Check -----> 252.884160
```

- ✓ size (e.g. 1024) hard-coded in the code
- ✓ check \sim size/4
-

Homework: Fill the table

Size	Fortran	C
1024*1024		
2048*2048		
4096*4096		
8192*8192		

- ✓ Compiler used:
 - ✓ Compiler option used:
 - ✓ HW used:
-