1 Exercise 1: Download and install the tutorial

```
$ cd ~
$ mkdir alpstut
$ cd alpstut
$ git clone https://github.com/ALPSCore/Tutorial2
$ cd Tutorial2
$ tutorial=$PWD
$ ls -l
```

You should see a list of files and no error messages.

2 Exercise 2: Download/install prerequisites

2.1 Ubuntu Linux

```
$ sudo apt-get install cmake
$ sudo apt-get install libhdf5-dev
$ sudo apt-get install libboost-all-dev
$ sudo apt-get install mpi-default-dev
```

2.2 Mac OS X

```
$ port install alpscore
```

This will install the latest ALPSCore release (we don't need it!) and prerequisites.

2.3 Check

```
$ cmake --version
$ g++ --version
$ h5cc --version
$ mpicxx --version
```

3 Exercise 3: Download and install ALPSCore.

```
$ git https://github.com/ALPSCore/ALPSCore
$ cd ALPSCore
$ mkdir build
$ cd build
$ export ALPSCore_DIR=$PWD/install
$ cmake -DCMAKE_INSTALL_PREFIX=$ALPSCore_DIR ..
$ make
```

```
$ make test
$ make install
```

4 Exercise 4: Build and run a dummy program that uses ALPSCore and does nothing.

The code is at $tutorial/step1_trivial.

CMake file online: `https://git.io/alpstut2_s1_cmake`

Source file online: `https://git.io/alpstut2_s1_main`

```
$ cd $tutorial/step1_trivial
$ mkdir 000build
$ cd 000build
$ cmake ..
$ make
$ ./alpsdemo
```

5 Exercise 5: Build and run a program that uses parameters.

The code is at $tutorial/step2_params.

Online: `https://git.io/alpstut2_s2`

1. Play with the different values of parameters.
2. Try to override them from the command line.
3. Change the program to make `--loud` parameter an integer, with value 0 meaning "be quiet".

```
$ cd $tutorial/step2_params
$ mkdir 000build
$ cd 000build
$ cmake ..
$ make
$ ./alpsdemo
$ ./alpsdemo --help
$ ./alpsdemo ../params.ini
$ ./alpsdemo ../params.ini --count=3
....
```

6 Exercise 6: Build and run a trivial MC program

The code is at $tutorial/step3_trivial_mc.

Online: `https://git.io/alpstut2_s3`

1. Build and run.
   `$ ./alpsdemo --help`
2. Run with small counts:

```
$ ./alpsdemo --count=2
$ ./alpsdemo --count=2 --verbose
```

3. Run with large count and small timelimit; time the execution:
   `$ time -p ''your_command''`

```
$ time -p ./alpsdemo --count=10000000 --timelimit=1
```

4. Change `fraction_completed()` so that `--count=0` would mean "till timeout".
5. Change the name of the `update()` method and see it does not compile any more.

7 Git cheatsheet for today

Clone ("download") repository:
`git␣clone␣`*repo*`␣`*local_directory*
or
`git␣clone␣`*repo*
Example:
`git␣clone␣https://github.com/ALPSCore/Tutorial2.git`

See the current changes:
`git␣diff`

Revert changes (CAUTION — no way back!):
`git␣reset␣--hard`

8 Important links

ALPSCore site: `http://alpscore.org`

ALPSCore repo: `https://github.com/ALPSCore/ALPSCore`

Doxygen docs: `https://alpscore.ci.cloudbees.com`
`/job/alpscore_fedora17_doc-only/ALPSCore_reference/`

Tutorial repo: `https://github.com/ALPSCore/Tutorial2`

9 ALPSCore CMake and environment variables

CMake variables: set as `cmake -D`*`variable=value`*.

Variable	Default value	Comment
`CMAKE_CXX_COMPILER`	(system default)	Path to C++ compiler executable. Can be set only once.
`CMAKE_INSTALL_PREFIX`	`/usr/local`	ALPSCore target install directory.
`CMAKE_BUILD_TYPE`		Specifies build type; set to `Release` to maximize performance.
`BOOST_ROOT`		Boost install directory. Set if CMake fails to find Boost.
`Boost_NO_SYSTEM_PATHS`	false	Disable search in default system directories. Set if the wrong version of Boost is found.
`Boost_NO_BOOST_CMAKE`	false	Disable search for Boost CMake file. Set if the wrong version of Boost is found.
`Documentation`	ON	Build ALPSCore developer's documentation.
`ENABLE_MPI`	ON	Enable MPI build.
`Testing`	ON	Build unit tests (recommended).
`ALPS_BUILD_SHARED`	ON	Build ALPSCore as shared libraries. Mutually exclusive with `ALPS_BUILD_STATIC=ON`.
`ALPS_BUILD_STATIC`	OFF	Build ALPSCore as static libraries. Mutually exclusive with `ALPS_BUILD_SHARED=ON`.

The environment variables are set via `export` *`variable=value`* before running CMake. The relevant variables are:

Variable	**Comment**
`CXX`	Path to C++ compiler executable. Can be set only once.
`BOOST_ROOT`	Boost install directory. Set if CMake fails to find Boost.
`HDF5_ROOT`	HDF5 install directory. Set if CMake fails to find HDF5.