ESIPAP European School of Instrumentation in Particle and Astroparticle Physics

2019 edition

Supervisers: Eric CHABERT, Eric CONTE

Computing session 2

Implementing a C++ class PsychrometricCalculator

Abstract:
The goal of this computing session is to write step-by-step a complete and operational C++ class. The class content is described by a UML (Unified Modeling Language) diagram. The class implementation must be tested in a main program.

The class to implement is called `PsychrometricCalculator` and must be able to derive all the properties of the humid air from the measures of the temperature, the relative humidity and the pressure obtained in Computing Session 1. For instance, can be computed the absolute humidity, the specific enthalpy, the dew temperature and more.

Pedagogical goals:

C++ language

- Writing new classes from UML diagrams.
- Instantiating objects from classes and initializing them.
- Reading and adapting an existing piece of code.
- Improving the robustness of the code in order to prevent abnormal termination or unexpected actions.

Collaboration work

- Respecting a given set of programming rules and conventions.
- Generating automatically the reference documentation related to the code with DOXYGEN.

Compiling/linking

- Creating an executable file from a simple source file.
- Compiling and linking via a script a project made up of several source files.

Requirements:

- Concept of class in C++, including constructors, destructor, mutators, accessors, ...
- Some particular C++ points: I/O access, arrays, pointers/references, algorithms.

Contents

Part I
Development environment

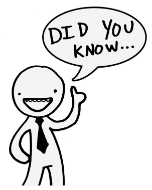
1 Foreword

Computing sessions belong to the educational program of the ESIPAP (European School in Instrumentation for Particle and Astroparticle Physics). Their goal is to teach the secrets of C++ programming through practical work in the context of high energy physics. The session is designed to be pedagogical. It is advised to read this document section-by-section. Indeed, except the *Physics context*, each section of the document is a milestone allowing to acquire computing skills and to validate them. The sections related to C++ programming are ranked in terms of complexity. In order to facilitate the reading of this document and to measure his progress, the student must **fill up the dedicated roadmap** which includes a check-list and empty fields for personal report.

In the document, some graphical tags are used for highlighting some particular points. The list of tags and their description are given below.

The student is invited to perform a pratical work by **writing a piece of code** following some instructions.

Analyzing or interpreting task is requested and the results must be reported in the roadmap.



Some **additional information** is provided for extending the main explanations. It is devoted to curious students.

A piece of **advice** is given to help the student in his task.

2 The ESIPAP framework

The practical works must be performed on devoted machines where all required software are properly installed. The user will find below all the instructions for setting the environment at each beginning of session.

2.1 Launching the `Windows` machine

You must choose a computer in the computing room, spot its name and check that no peripheral is missing (mouse, keyboard, ...). Then boot it and login to the `Windows` operator system (supervisors will provide the password access).

2.2 Accessing the `Linux` virtual machine

The practical sessions will be achieved on a `Linux` machine for pedagogical motivations. You must connect a virtual machine. First click on the "Start" button, i.e. the button with the `Windows` logo, located on the bottom left of the screen (see Figure **??**).

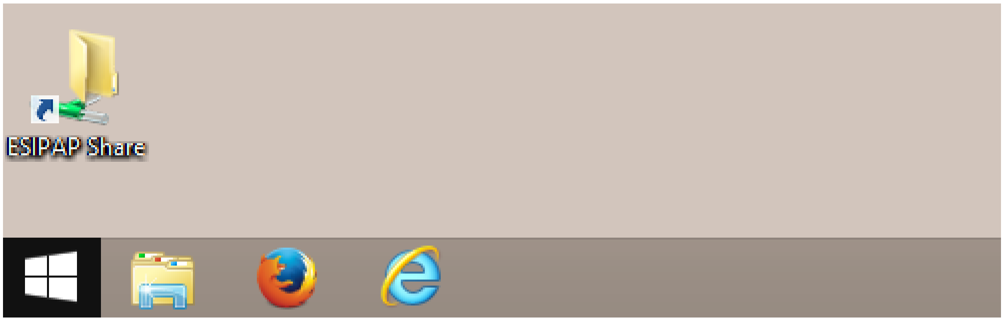


Figure 1: The `Windows` Start button

According to Figure **??**, click on the virtual machine called **"ESIPAP_slc6"**. A password could be necessary and should be supplied by the supervisors.

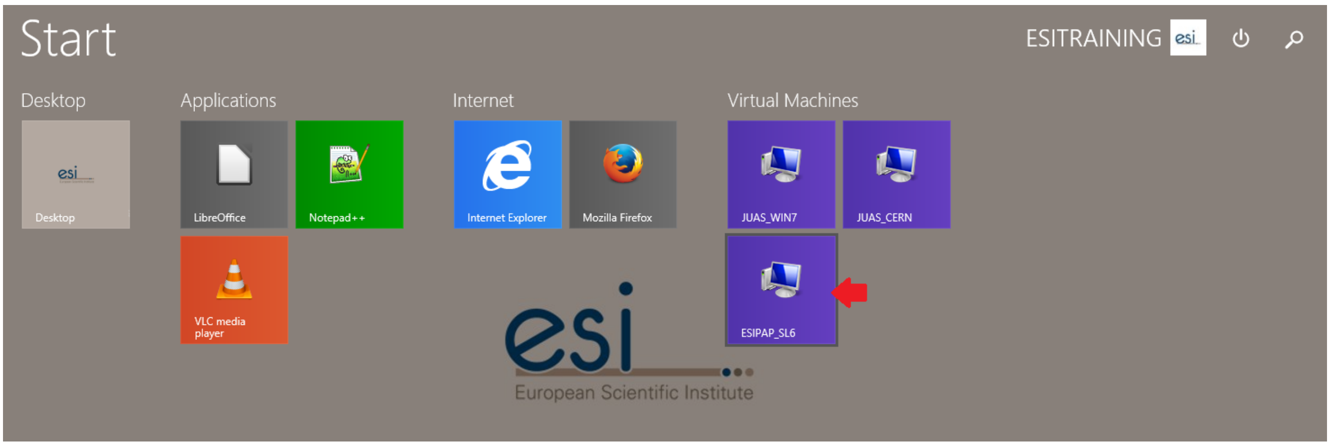


Figure 2: The screen showing the available virtual machines

2.3 Setting the environment

To load the work environment, you can issue the command below at the shell prompt.

```
bash$source␣/home/esipap/tools/setup.sh
```

If the system is properly installed, the version of each tool to study should be displayed at the screen like below. If you have an error, please call the supervisors.

```
-----------------------------------------------
             ESIPAP environment
-----------------------------------------------
 - GNU g++  version 4.9.1
 - ROOT     version 6.06/00
 - Geant4   version 10.2.0
-----------------------------------------------
```

Part II
The class PsychrometricCalculator

3 First implementation of the class

In this section, a class called `PsychrometricCalculator`, corresponding to the files called `PsychrometricCalculator.h` and `PsychrometricCalculator.cpp`, must be written. This class must be able to compute the properties of the humid air.

3.1 Specifications

Here are enumerated the functionalities of the class `PsychrometricCalculator`.

- The class must contain the dry temperature `t_` in [Kelvin], the absolute humidity `ah_` in [kg water / kg dry air] and the pressure of the humid air `p0_` in [Pa].
- Two constructors will be implemented for this class: one constructor with no argument where data members will be initialized to arbitrary values and a second constructor where the values of data members are given into arguments.
- The class must contain a function `Clear` and `Print` for respectively clearing and displaying at the screen the data members.
- The class must contain the mutator `SetTemperatureAbsoluteHumidityPressure` for setting the data members. The units of the inputs are Celsius degree for temperature, kg water / kg dry air for absolute humidity and bar for pressure.
- The class must contain the mutator `SetTemperatureRelativeHumidityPressure` for setting the data members. In this case, the absolute humidity `ah_` must be calculated from the relative humidity by using Equations 3 and 1. The units of the inputs are Celsius degree for temperature, % for relative humidity and bar for pressure.
- The class must contain some accessors in order to get the data member values. These accessors will be called:
  - `GetTemperature` (or equivalently) `GetDryTemperature` which returns the temperature in Celcius degree,
  - `GetAbsoluteHumidity` which returns the absolute humidity in kg water / kg dry air,
  - `GetPressure` which returns the pressure in bar.
- The class must contain several specific functions for computing the properties of the humid air:
  - `GetVapourPressure`: computing and returning the water vapour pressure [in bar] according to Equation 1.
  - `GetRelativeHumidity`: computing and returning the relative humidity [in %] according to Equation 2.
  - `GetSpecificVolume`: computing and returning the air specific volume [in m^3 / kg dry air] according to Equation 4.
  - `GetDensity`: computing and returning the air density [in kg dry air / m^3] according to Equation 5.

- `GetSpecificEnthalpy`: computing and returning the air specific enthalpy [in J / kg dry air] according to Equation 6.

The UML diagram corresponding to the class `PsychrometricCalculator` is supplied below.

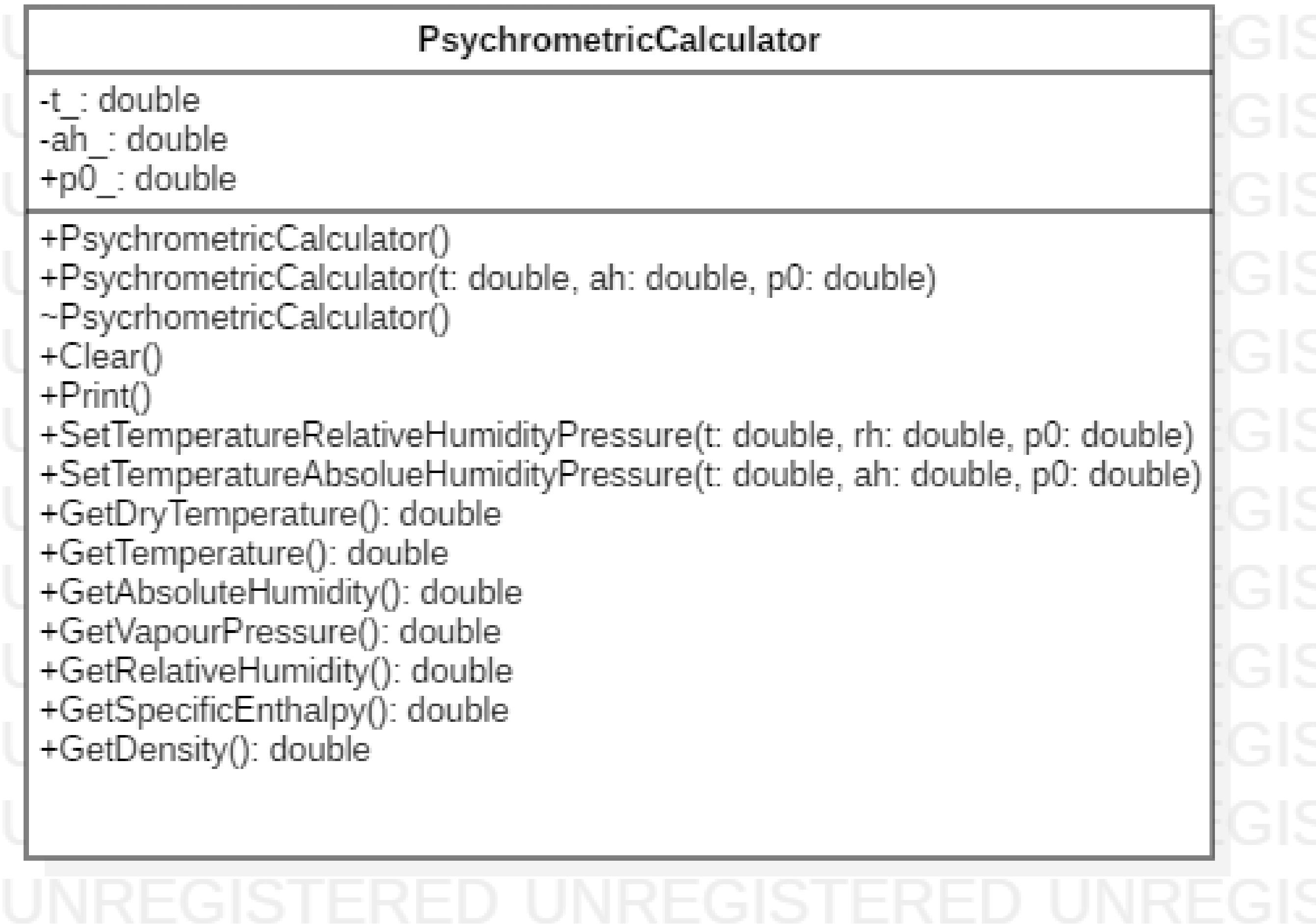


Figure 3: UML diagram of the class ***PsychrometricCalculator***

3.2 Equations of the humid air

Here the list of equations governing the physics properties of the humid air:

- vapour pressure parametrization for water : p_v (vapour pressure of the water) [Pa] as a function of the dry temperature T [K] [Buck (1981), J. Appl. Meteorel., 20: 1527-1532]

$$p_v(T) = \exp\left[a_{-1}/T + a_0 + a_1 T + a_2 T^2 + a_3 T^3 + a_4 \log(T)\right] \tag{1}$$

with the following coefficients:

a_{-1}	$-5.8002206 \times 10^{+3}$
a_0	1.3914993
a_1	$-4.8640239 \times 10^{-2}$
a_2	4.1764768×10^{-5}
a_3	$-1.4452093 \times 10^{-8}$
a_4	6.5459673

- Relative humidity rh [no unit] as a function of the dry temperature T [K], the pressure p_0 [Pa] and the absolute humidity ah [kg water / kg dry air]:

$$rh = \frac{\mu}{1 - (1 - \mu)\dfrac{p_v(T)}{p_0}} \text{ where } \mu = \frac{ah}{w} \cdot \left(\frac{p_0}{p_v(T)} - 1 \right) \tag{2}$$

- The previous relation can be inverted in order to express the absolute humidity ah [kg water / kg dry air] as a function of the dry temperature T [K], the pressure p_0 [Pa] and the relative humidity rh [no unit]:

$$ah = \frac{w \cdot \mu}{\dfrac{p_0}{p_v(T)} - 1} \text{ where } \mu = rh \cdot \frac{p_0 - p_v(T)}{p_0 - rh \cdot p_v(T)} \tag{3}$$

- Specific volume v as a function of the dry temperature T [K] and the absolute humidity r [kg water / kg dry air]:

$$v = \frac{RT\left(1 + \dfrac{ah}{w}\right)}{M_a \cdot p_0} \tag{4}$$

- Density ρ as a function of the specific volume v:

$$\rho = 1/v \tag{5}$$

- Specific enthalpy h as function of the dry temperature T [K] and the absolute humidity ah [kg water / kg dry air]:

$$h = 2501 \cdot ah + (1.805 \cdot ah + 1.006)(T - 273.15) \tag{6}$$

Constants:

R	ideal gas constant	8.3144621 J/mol/K
M_a	molar mass of dry air	28.9653×10^{-3} kg/mol
M_w	molar mass of water	18.0153×10^{-3} kg/mol
w	ratio of molar masses	M_w/M_a

Physics observables:

ah	absolute humidity [kg water / kg dry air]
rh	relative humidity [no unit]
h	specific enthalpy [J / kg dry air]
v	specific volume [m^3 / kg dry air]
ρ	density [kg dry air / m^3]
T	dry temperature [K]
p_v	vapour pressure of water [Pa]
p_0	total pressure [Pa]

3.3 Tasks to do

- **Implement this first version of class `PsychrometricCalculator` according to the UML diagram.**
- **Test the class definition by instantiating an object and by performing some operations.**
- **Adapt the script `mymake` for building this project.**

4 Enriching the functionalities of the class

The goal is to add an additional function to the UML diagram. The function `GetDewTemperature` which allows to compute and return the Dew Temperature. To reach this aim, the developer must:

- determine the vapour pressure of water p_v for the absolute humidity ah by using the equation:

$$p_v = \frac{ah \cdot p_0}{ah + w}$$

- extract the temperature T corresponding to p_v by inversing numerically Equation 1.

- **Implement the function `GetDewTemperature`**

5 Enriching the structure of the class

We would like to improve the structure of the class. These improvements are not crucial for the next developments. Their goal is totally pedagogical.

- **Add a copy constructor to the class.**
- **Associate the reserved word** `const` **to the appropriated functions.**
- **Overload the operator << to display the data member values when** `std::cout` **is applied directly to instance of this class.**
- **Overload the operator** `[]` **to get the dew temperature for** `[0]`**, the specific enthalpy for** `[1]` **and the specific volume for** `[2]`**.**

Part III
Generation of the class reference documentation

6 Generating documentation from C++ sources

Annotation and comments inside the code is very useful for the understanding. In order to increase the documentation level, it is also possible to generate automatically reference documentation by reading the syntax and the annotations of the code. Whereas some documentation generators such as JAVADOC are specific to one programming language, the DOXYGEN program has the advantage to be used for plenty languages.

6.1 First words about the Doxygen package

DOXYGEN can read not only C++ language but also JAVA PYTHON, FORTRAN, PHP and others. The formats of the generated documentation are mainly HTML and LATEX (PDF or PS after LATEX compilation). It can cross reference documentation and code, so that the reader of a document can easily refer to the documentation.

The package can be downloaded from the official website (`www.doxygen.org`). From the LXPLUS session, DOXYGEN program can be launched from any folder. A small test to check the presence of this package consists in issuing the command below at the shell prompt. If the program is found, the version release must appear at the screen.

```
bash$doxygen --version
```

6.2 Standard doxygen configuration file

The starting point consists in writing a DOXYGEN configuration file. A template of a such file can be generated by typing the following command:

```
bash$doxygen -g doxygen.cfg
```

A text file called `doxygen.cfg` is then created and can be modified with a text editor. It contains all the available `Doxygen` options set with the default values. The syntax is very similar to a shell script. To enter into details, comment line begins with a # character and options are specified by the scheme `tag = value`. The options values are usually the reserved words `YES` or `NO` for binary options, or string for other option kinds. Appearance order of the options is not relevant.

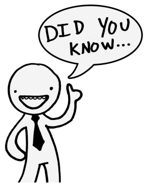
For generating HTML, the user must set the following settings:

```
GENERATE_HMTL = YES
HMTL_OUTPUT = html # name of the folder where HTML document
                   # will be generated
```

and for LATEX, the following lines

```
GENERATE_LATEX = YES
LATEX_OUTPUT = latex # name of the folder where LATEX document
                     # will be generated
```

By default, all source files (C++ and other programming languages) placed in the local folder are taken into account. These properties can be tuned by changing options such as `FILE_PATTERNS`, `RECURSIVE` and `EXCLUDE`.



A GUI (Graphical User Interface) wizard configuration tool, called `doxywizard`, exists also. It facilitates the DOXYGEN configuration and running. Nonethess this program is not installed on LXPLUS session.

6.3 Adding graphics in the reference documentation

DOXYGEN tool can use GRAPHVIZ package for generating graphs and diagrams. It can be downloaded from the official website (`http://www.graphviz.org/`). From the LXPLUS session, GRAPHVIZ is already installed and ready to used. A small test to check the presence of this package consists in issuing at the shell prompt the command below. The version of GRAPHVIZ must appear at the screen.

```
bash$dot␣-v
```

For enabling all the graphical options in the report, the user must apply the following settings:

```
HAVE_DOT␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣=␣YES
CLASS_GRAPH␣␣␣␣␣␣␣␣␣␣␣␣=␣YES
COLLABORATION_GRAPH␣␣␣␣=␣YES
GROUP_GRAPHS␣␣␣␣␣␣␣␣␣␣␣=␣YES
UML_LOOK␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣=␣NO
TEMPLATE_RELATIONS␣␣␣␣␣=␣YES
INCLUDE_GRAPH␣␣␣␣␣␣␣␣␣␣=␣YES
INCLUDED_BY_GRAPH␣␣␣␣␣␣=␣YES
CALL_GRAPH␣␣␣␣␣␣␣␣␣␣␣␣␣=␣YES
CALLER_GRAPH␣␣␣␣␣␣␣␣␣␣␣=␣YES
GRAPHICAL_HIERARCHY␣␣␣␣=␣YES
DIRECTORY_GRAPH␣␣␣␣␣␣␣␣=␣YES
DOT_MULTI_TARGETS␣␣␣␣␣␣=␣YES
```

6.4 Launching Doxygen

To generate automatically documentation, the user has just to type the `Doxygen` command following the name of the configuration file:

```
bash$doxygen␣doxygen.cfg
```

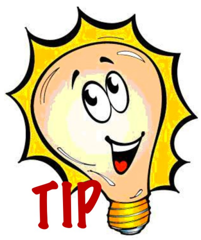
During the documentation generation, error or warning could be displayed. The user is invited to read these messages and to investigate the relevant ones. If the running is successful, folders `html` and `latex` are generated according to the configuration file.

- `html` folder contains all HTML files and can be browsed with a navigator internet from the file `index.html`.

- `latex` folder contains `latex` files and can be compiled with latex with a makefile. By issiung the command `make`, a PDF file is created and can be viewed with a PDF reader.

6.5 Work to do

- **Generate the documentation related to your code in Latex and HTML format**
- **Add/adjust annotations in your code in order to improve the generated documentation.**



Some suggestions about the documentation layout:

```
FULL_PATH_NAMES␣␣␣␣␣␣␣␣=␣NO
JAVADOC_AUTOBRIEF␣␣␣␣␣␣=␣YES
HIDE_UNDOC_CLASSES␣␣␣␣␣=␣NO
GENERATE_LATEX␣␣␣␣␣␣␣␣␣=␣NO
TAB_SIZE␣␣␣␣␣␣␣␣␣␣␣␣␣␣␣=␣4
OPTIMIZE_OUTPUT_FOR_C␣␣=␣YES
BUILTIN_STL_SUPPORT␣␣␣␣=␣YES
EXTRACT_ALL␣␣␣␣␣␣␣␣␣␣␣␣=␣YES
RECURSIVE␣␣␣␣␣␣␣␣␣␣␣␣␣␣=␣YES
SOURCE_BROWSER␣␣␣␣␣␣␣␣␣=␣YES
ALPHABETICAL_INDEX␣␣␣␣␣=␣YES
GENERATE_TREEVIEW␣␣␣␣␣␣=␣YES
TEMPLATE_RELATIONS␣␣␣␣␣=␣YES
SEARCHENGINE␣␣␣␣␣␣␣␣␣␣␣=␣YES
REFERENCED_BY_RELATION␣=␣YES
```