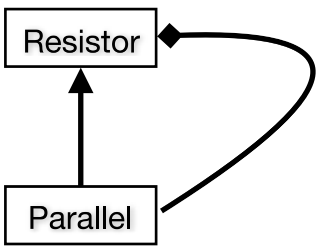
COMPUTING METHODS FOR PHYSICS
11 FEBRUARY 2019

You must submit your exam by **Wednesday Feb 13 at 12:00** following the instruction at http://www.roma1.infn.it/people/rahatlou/cmp/

Resistors in parallel with Composite Pattern (15 points)

Use the composite pattern to implement a polymorphic hierarchy of two classes to represent a simple `Resistor` and a composite `Parallel` object made of one or more `Resistor` or `Parallel` objects.



- `Resistor` must have 2 data members `_R` and `_name`
- Implement virtual methods `value()`, `name()`, `print()` for `Resistor`
- Implement `value()`, `name()`, `print()` for `Parallel`. `print()` must also produce info about all components in the composite object
- Implement `add()` method (with proper signature and argument) for `Resistor`
- Test your classes with a `exam.cc` application to obtain this output

```
$ /tmp/app.exe
Create R1 and R2
R1: 10 ohm
R2: 10 ohm
make a parallel R6 of R1 and R2 and print it
R1||R2: 5 ohm
  R1: 10 ohm
  R2: 10 ohm
Create R7 and R8
make a parallel R9 of R7 and R8 and print it
R7||R8: 2.5 ohm
  R7: 5 ohm
  R8: 5 ohm
add R6 to R9 and print it
R7||R8||R1||R2: 1.66667 ohm
  R7: 5 ohm
  R8: 5 ohm
  R1||R2: 5 ohm
    R1: 10 ohm
    R2: 10 ohm
+++++  bonus +++++
R1: 10 ohm
R2: 10 ohm
R3 = R1 + R2
(R1+R2): 20 ohm
R4 = R1 || R2
(R1||R2): 5 ohm
```

- No additional functions, classes or applications are needed. If additional functions are present, they will be evaluated and possible mistakes will decrease the total score.

Provide instructions for compiling your code in the comments at the beginning of `exam.cc` .

Evaluation will be based on: successful compilation, correct use of C++ syntax, polymorphism, return type and arguments of functions, data members and interface of classes, unnecessary void functions, use of unnecessary C features, correct rules for calculation of resistance.

Extra (2 points): Overload operator **+** to add two `Resistor` in series. Overload operator `||` to add two `Resistor` in parallel.

NB: You cannot replace a mandatory part with the extra question. Extra will be considered only if all mandatory parts completed.

COMPUTING METHODS FOR PHYSICS
11 FEBRUARY 2019

Motion with gravity in 3D with Euler method (15 points)

We want to study the motion of a body of mass `m` in 3 dimensions with initial position `r₀=(x₀,y₀,z₀)` and initial velocity `v₀=(vx₀, vy₀, vz₀)` moving under the effect of gravity until it hits the ground. In particular we want to use the Euler method (see the example we used in the simulation of the solar system)

$$x_{i+1} = x_i + \frac{dx}{dt}\Delta t$$
$$v_{i+1} = v_i + \frac{dv}{dt}\Delta t$$

to integrate the equation of motion and compare it to the exact analytical solution.

1. Use a NumPy array of 3 elements to represent a 3D vector for position, velocity and acceleration
2. Ask user for initial conditions r_0 and v_0 and provide default values that can be used if no value entered by user
3. Define a function called **`move`** to implement the Euler method
4. Evolve the position of the particle until it hits the ground
5. During the trajectory compute the difference Δz between the altitude z_{comp} computed with the Euler method and z_{true} given by the analytical equation of motion
6. Plot the distribution of Δz as a function of time during the motion until the body hits the ground (no animation needed)
7. Create a dictionary with time as the key and the 3D distance between the estimated position and the analytical position as the value
8. Visualise the 3D trajectory of the motion   (no animation)

Extra (1 points): Animate the motion of the particle and **show simultaneously** the estimated position with Euler method in blue and the analytical position with dashed red line.

Evaluation will be based on: Use of python features and data structures, comprehensions (instead of C-style for loops), NumPy objects, labels , units and clarity of plots.

NB: You cannot replace a mandatory part with the extra question. Extra will be considered only if all mandatory parts completed.