C++ - Assignment 6

Polymorphism and abstract classes

Charanjit Kaur & Caterina Doglioni, PHYS30762 - OOP in C++ 2023

MANCHESTER 1824
The University of Manchester

A set of classes for shapes

- Write a class hierarchy to describe the properties of 2D and 3D shapes.
- **At a minimum, your class hierarchy should contain:**
  - An abstract base class used as an interface for all shapes;
  - An abstract derived class for a) 2D shapes and b) 3D shapes;
  - Concrete derived classes for at least 4 2D shapes (rectangle,square,ellipse,circle);
  - Concrete derived classes for at least 4 3D shapes (cuboid,cube,ellipsoid,sphere);
  - Additional inheritance for specialisation (“is a”);
  - Constructors and (virtual) destructors;
  - Pure virtual functions to return area and volume (can be zero for 2D shapes).
- **Your main program must:**
  - Implement an array or vector of at least 8 base class pointers, each pointing to a different shape ;
  - Output the area (for 2D case) and volume (for 3D) of each shape;
  - Clean up by deleting objects and array when finished.
- Assignment deadline: **Friday April 21st, 19:00**

Marking / marking rubric

- The following marks apply if your program contains:
  - An abstract base class for shapes with appropriate pure virtual functions and virtual destructor. **[0.5 marks]**
  - Eight derived classes (4 2D, 4 3D), each appropriately overriding (pure virtual) functions. **[0.5 marks]**
  - Correct use of inheritance for specialization. **[0.5 marks]**
  - Code demonstrates polymorphism using an array or vector of base class pointers, each pointing to different shape. **[0.5 marks]**
  - Code correctly prints out the area (2D) and area and volume (3D) of each shape. **[0.5 marks]**
  - Code correctly cleans up, deleting objects and array. **[0.5 marks]**
- Challenge mark: write an extra prism class (again, a derived class!) whose constructor takes a double (the depth of the prism) and a 2D shape through a base class pointer. Class must contain a function to return volume of the prism (prism depth × area from 2D shape). **[1 mark]**
- Total: 4 marks

Marking / marking rubric - I

	Not at all	Partially	Completely
Code contains abstract base class for shapes with appropriate pure virtual functions and virtual destructor	**0** (0.00%)	**0.25** (6.25%)	**0.5** (12.50%)
Code contains 8 derived classes (square, rectangle, ellipse, circle, cube, cuboid, ellipsoid, sphere), each appropriately overriding pure virtual functions	**0** (0.00%)	**0.25** (6.25%)	**0.5** (12.50%)
Correct use of inheritance for specialization	**0** (0.00%)	**0.25** (6.25%)	**0.5** (12.50%)
Code demonstrates polymorphism using an array (or vector) of base class pointers, each pointing to different shape	**0** (0.00%)	**0.25** (6.25%)	**0.5** (12.50%)
Code correctly prints out area (2D) and area/volume (3D) of each shape	**0** (0.00%)	**0.25** (6.25%)	**0.5** (12.50%)
Code cleans up, deleting objects and array/clear vector	**0** (0.00%)	**0.25** (6.25%)	**0.5** (12.50%)
Challenge mark: write an extra prism class (a derived class!) whose constructor takes a double (depth of prism) and a base class pointer to any of the 2D shapes. Class must contain a function to return volume of the prism	**0** (0.00%)	**0.5** (12.50%)	**1** (25.00%)

The University of Manchester

Marking / marking rubric - II

- [total 4 marks] - you lose marks if:
- The code generates **error** messages on compilation. [up to -1 mark]
  - If you want to clear all doubts from differences between your compiler and the demonstrator's compiler, you can also submit a screenshot of the compilation output from one of the lab computers
  - Try to get rid of compilation warnings as well
- Code layout, presentation, comments, variable names not clear (think also of house style here). [up to -1 mark]
- The code does not adequately demonstrate the use and implementation of the class. [up to -1 mark]

Program does compile and run	**-1** (-25.00%)	**-0.5** (-12.50%)	**0** (0.00%)
The code layout, presentation, comments, variable names are of sufficient quality.	**-1** (-25.00%)	**-0.5** (-12.50%)	**0** (0.00%)
The code does adequately demonstrate the use and implementation of the class	**-1** (-25.00%)	**-0.5** (-12.50%)	**0** (0.00%)

- You lose all marks if you do not submit the .cpp file(s) (and .h files if used). You can use a zip file to pack everything