Abstraction_Lab3

May 9, 2020

In [ ]:
```cpp
#include <cassert>
#include <cmath>
#include <stdexcept>

class Sphere {
 public:
  Sphere(int radius) : radius_(radius), volume_(M_PI * 4/3 * pow(radius_, 3)){
    if (radius <= 0) throw std::invalid_argument("radius must be positive");
  }

  int Radius() const { return radius_; }
  int Volume() const { return volume_; }

  // TODO: mutators
  void Radius(int radius){
      if (radius <= 0){
          throw std::invalid_argument("radius must be positive");
      }
      else{
          radius_ = radius;
          volume_ = M_PI * 4 / 3 * pow(radius_, 3);
      }
  }

 private:
  int radius_;
  float volume_;
};

// Test
int main(void) {
  Sphere sphere(5);
  assert(sphere.Radius() == 5);
  assert(abs(sphere.Volume() - 523.6) < 1);

  sphere.Radius(3);
  assert(sphere.Radius() == 3);
  assert(abs(sphere.Volume() - 113.1) < 1);
```

```
    bool caught{false};
    try {
      sphere.Radius(-1);
    } catch (...) {
      caught = true;
    }
    assert(caught);
  }
```

Compile & Run

Explain
Loading terminal (id_2q0fnq4), please wait...