Lab 5.1.3 Obtaining derived data from an object

Objectives

Familiarize the student with:

- creating getter access methods;
- different strategies for obtaining derived data.

Scenario

One aspect of encapsulation is setting the data in the object, but there is another aspect to it.

We can also take different approaches to getting the data out of the object.

- In the previous example, we asked for the area to be recalculated every time the side was changed. When calculating the square of a number, this is not too costly, but in the case of a more complicated operation, this may be wasteful.
- A more ad hoc approach would be to calculate the area only when needed, i.e. not store it in the object at all. This way we don't recalculate the value on each side change; but again, more costly operations may become a burden.
- Yet another, lazy, approach would be to recalculate the value only when needed. We can add a small bit of information to the object that will tell us that the side has changed. When asked for the area, we can check if the side has changed since the last recalculation; if not, we simply return the stored value, and otherwise we recalculate the area and clear the information about side changes.

See the class definitions below and implement the two new approaches to the above problem.

```
class AdHocSquare
{
public:
  AdHocSquare(double side);
  void set_side(double side);
  double get_area();
private:
  double    side;
};


class LazySquare
{
public:
  AdHocSquare(double side);
  void set_side(double side);
  double get_area();
private:
  double    side;
  double    area;
  bool      side_changed;
};
```