

SEARCH

RESOURCES

CONCEPTS

1. Classes and OOP

2. Bjarne On Classes In C++

3. Jupyter Notebooks

4. Structures

5. Member Initialization

6. Access Specifiers

7. Classes

8. Encapsulation and Abstraction

9. Bjarne on Encapsulation

10. Constructors

11. Scope Resolution

12. Initializer Lists

13. Initializing Constant Members

14. Encapsulation

15. Accessor Functions

16. Mutator Functions

17. Quiz: Classes In C++

18. Exercise: Pyramid Class

19. Exercise: Student Class

20. Encapsulation in C++

21. Bjarne On Abstraction

22. Abstraction

23. Exercise: Sphere Class

24. Exercise: Private Method

25. Exercise: Static Members

26. Exercise: Static Methods

27. Bjarne On Solving Problems

## Exercise: Encapsulation

Add a private member function that calculates the number of days in a month, and use it to update the class invariants. Be sure to account for [leap years!](https://en.wikipedia.org/wiki/Leap_year#Algorithm) ([https://en.wikipedia.org/wiki/Leap\\_year#Algorithm](https://en.wikipedia.org/wiki/Leap_year#Algorithm))

```
In [ ]: #include <cassert>

class Date {
public:
    Date(int day, int month, int year):
        int Day() const { return day_; }
    void Day(int day):
        int Month() const { return month_; }
    void Month(int month):
        int Year() const { return year_; }
    void Year(int year):

private:
    bool LeapYear(int year) const:
    int DaysInMonth(int month, int year) const:
    int day_{1};
    int month_{1};
    int year_{0};
};

Date::Date(int day, int month, int year) {
    Year(year);
    Month(month);
    Day(day);
}

bool Date::LeapYear(int year) const {
    if(year % 4 != 0)
        return false;
    else if(year % 100 != 0)
        return true;
    else if(year % 400 != 0)
        return false;
    else
        return true;
}

int Date::DaysInMonth(int month, int year) const {
    if(month == 2)
        return LeapYear(year) ? 29 : 28;
    else if(month == 4 || month == 6 || month == 9 || month == 11)
        return 30;
    else
        return 31;
}

void Date::Day(int day) {
    if (day >= 1 && day <= DaysInMonth(Month(), Year()))
        day_ = day;
}

void Date::Month(int month) {
    if (month >= 1 && month <= 12)
        month_ = month;
}

void Date::Year(int year) { year_ = year; }

// Test
int main() {
    Date date(29, 2, 2016);
    assert(date.Day() == 29);
    assert(date.Month() == 2);
    assert(date.Year() == 2016);

    Date date2(29, 2, 2019);
    assert(date2.Day() != 29);
    assert(date2.Month() == 2);
    assert(date2.Year() == 2019);
}
```

Compile & Run

Explain

Loading terminal (id\_gfhfd7), please wait...