



C++

Разбор домашней работы



Модификаторы доступа. Геттеры и сеттеры

Теория



15 минут



Инкапсуляция

Инкапсуляция - размещение в одном компоненте данных и методов, которые с ними работают.

По сути, классы уже являются примером использования инкапсуляции

private

private - модификатор доступа, при котором поля и методы класса доступны только внутри этого класса.

Если не указывать модификатор доступа, то содержимое класса по умолчанию будет приватным

Зачем скрывать данные в классе?

Скрытие данных

Скрывать данные в классе можно по нескольким причинам. Например:

- Есть ограничения на хранимые данные, которые необходимо учитывать
 - Например - возраст человека. Возраст не может быть отрицательным или же очень большим
- Данные внутри класса хранятся в ином от отображаемого виде
 - Время хранится в секундах, для более удобного изменения времени, а уже выводится оно в часах, минутах и секундах

Как использовать приватные поля, если они не доступны вне класса?

Геттеры и сеттеры

Для получения доступа к приватным данным в классе нужно использовать публичные методы.

Для получения значения полей и их изменения используются методы, называемые **геттерами** и **сеттерами** соответственно.



Пример

```
class Hero {  
private:  
    string name;  
    int exp, hp;  
public:  
    void setName(string s) {  
        name = s;  
    }  
    void setHp(int h) {  
        if (h < 0) h = 0;  
        if (h > 100) h = 100;  
        hp = h;  
    }  
};
```

```
    void setExp(int e) {  
        if (e < 0) e = 0;  
        exp = e;  
    }  
    string getName() { return name; }  
    int getHp() { return hp; }  
    int getExp() { return exp; }
```

Пример

```
class Time {  
    int seconds;  
public:  
    int getSeconds() { return seconds / 3600 % 60; }  
    int getMinutes() { return seconds / 60 % 60; }  
    int getHours() { return seconds / 3600 % 24; }  
    void print() {  
        int h = getHours(), m = getMinutes(), s = getSeconds();  
        cout << h / 10 << h % 10 << ":" << m / 10 << m % 10;  
        cout << ":" << s / 10 << s % 10 << endl;  
    }  
    void changeTime(int s) {  
        seconds += s;  
        if (seconds < 0) seconds += 3600 * 24;  
        if (seconds > 3600 * 24) seconds -= 3600 * 24;  
    }  
};
```

Пример

```
class Time {  
    int seconds;  
public:  
    int getSeconds() { return seconds / 3600 % 60; }  
    int getMinutes() { return seconds / 60 % 60; }  
    int getHours() { return seconds / 3600 % 24; }  
    void print() {  
        int h = getHours(), m = getMinutes(), s = getSeconds();  
        cout << h / 10 << h % 10 << ":" << m / 10 << m % 10;  
        cout << ":" << s / 10 << s % 10 << endl;  
    }  
    void changeTime(int s) {  
        seconds += s;  
        if (seconds < 0) seconds += 3600 * 24;  
        if (seconds > 3600 * 24) seconds -= 3600 * 24;  
    }  
};
```

В данном случае
указывать private не
обязательно

Тест



5 минут



Разберем тест

Практика



30 минут



Итоги урока

- 1) Что такое инкапсуляция
- 2) Как скрывать данные
- 3) Зачем скрывать данные
- 4) Как работать со скрытыми данными: геттеры и сеттеры