C++

Урок 15

Разбор ДЗ

Разбор дз

1 задача:

Задача: Создайте класс фигура, у которой есть два поля (две стороны) a и b. Требуется посчитать периметр и площадь.

Разбор дз

1 задача:

```
int main()
{
    figure kv1;
    read(kv1);
    show(per(kv1));
    show(sq(kv1));
}
```

main()

Разбор дз

1 задача:

```
class figure
{
public:
    int a;
    int b;
};
```

Класс figure

Разбор дз

1 задача:

```
void read(figure &test)
{
    cin >> test.a >> test.b;
}

void show(double number)
{
    cout << number << endl;
}

double per(figure &test)
{
    return 2 * (test.a + test.b);
}

double sq(figure &test)
{
    return (test.a * test.b);
}
```

Функции

Разбор дз

2/3 задача:

Задача: Создайте класс фигура, у которой есть два поля (две стороны) a и b. Требуется посчитать периметр и площадь+ валидация значений.

Разбор дз

2/3 задача:

```
int main()
{
    figure kv1;
    read(kv1);
    show(per(kv1));
    show(sq(kv1));
}
```

main()

Разбор дз

2/3 задача:

```
class figure
{

    double a;
    double b;

public:
    void set(double a_, double b_)
    {
        a = a_;
        b = b_;
        if (a < 0)
            a = 0;
        if (b < 0)
            b = 0;
    }

    int get_a()
    {
        return a;
    }
    int get_b()
    {
        return b;
    }
};
```

Класс figure

Разбор дз

2/3 задача:

```
void read(figure &test)
{
    double a, b;
    cin >> a >> b;
    test.set(a, b);
}

void show(double number)
{
    cout << number << endl;
}

double per(figure &test)
{
    return 2 * (test.get_a() + test.get_b());
}

double sq(figure &test)
{
    return (test.get_a() * test.get_b());
}
```

Функции

Разбор дз

4 задача:

Задача: Добавьте функцию сравнения двух фигур. Функция comparisson принимает на вход два аргумента (фигура1, фигура2). Требуется посчитать во сколько одна фигура больше другой в соотношении площадь/периметр.

Разбор дз

4 задача:

```
int main()
{
    figure kv1, kv2;
    read(kv1);
    read(kv2);
    comparisson(kv1, kv2);
}
```

main()

Разбор дз

4 задача:

```
class figure
{

    double a;
    double b;

public:
    void set(double a_, double b_)
    {
        a = a_;
        b = b_;
        if (a < 0)
            a = 0;
        if (b < 0)
            b = 0;
    }

    int get_a()
    {
        return a;
    }
    int get_b()
    {
        return b;
    }
};
```

Класс figure

Разбор дз

4 задача:

```
void read(figure &test)
{
    double a, b;
    cin >> a >> b;
    test.set(a, b);
}

void show(double number)
{
    cout << number << endl;
}

double per(figure &test)
{
    return 2 * (test.get_a() + test.get_b());
}

double sq(figure &test)
{
    return (test.get_a() * test.get_b());
}

void comparisson(figure &a, figure &b)
{
    double compare = 0;
    double val1 = sq(a) / per(a);
    double val2 = sq(b) / per(a);
    cout << "Отношение двух фигур равно: " << val1 / val2 << endl;
}
```

Функции

Разбор дз

5 задача:

Задача: Добавьте функцию set, которая заносит в поля объекта result значения медианы стороны a (объектов kv1, kv2, kv3) и медиану сторон b от объектов kv1, kv2, kv3.

Разбор дз

5 задача:

```
int main()
{
    figure kv1, kv2, kv3;
    read(kv1);
    read(kv2);
    read(kv3);
    figure result;
    set(result,med_a(kv1,kv2,kv3),med_b(kv1,kv2,kv3));
    show(result);
}
```

main() функция

Разбор дз

5 задача:

```
class figure
{
    double a;
    double b;

    void check()
    {
        if (a < 0)
            a = 0;
        if (b < 0)
            b = 0;
    }

public:
    void set(double a_, double b_)
    {
        a = a_;
        b = b_;
        check();
    }

    void set(double *a_, double *b_)
    {
        a = *a_;
        b = *b_;
        check();
    }
    int get_a()
    {
        return a;
    }
    int get_b()
    {
        return b;
    }
};
```

класс figure

Разбор дз

5 задача:

```
void read(figure &test)
{
    double a, b;
    cin >> a >> b;
    test.set(a, b);
}

void show(double number)
{
    cout << number << endl;
}

void show(figure &test)
{
    cout << test.get_a() << " " << test.get_b() << endl;
}
double per(figure &test)
{
    return 2 * (test.get_a() + test.get_b());
}

double sq(figure &test)
{
    return (test.get_a() * test.get_b());
}
```

Функции 1 часть

Разбор дз

5 задача:

```
void sort(double *ptr, int size)
{
    for (int i = 0; i < size; i++)
    {
        for (int j = 0; j < size - i - 1; j++)
        {
            if (ptr[j] > ptr[j + 1])
            {
                double temp = ptr[j];
                ptr[j] = ptr[j + 1];
                ptr[j + 1] = temp;
            }
        }
    }
}
double *med_a(figure &a, figure &b, figure &c)
{
    double *res = new double;
    double *tmp = new double[3];
    tmp[0] = a.get_a();
    tmp[1] = b.get_a();
    tmp[3] = c.get_a();
    sort(tmp, 3);
    *res = tmp[1];
    delete[] tmp;
    return res;
}

double *med_b(figure &a, figure &b, figure &c)
{
    double *res = new double;
    double *tmp = new double[3];
    tmp[0] = a.get_b();
    tmp[1] = b.get_b();
    tmp[3] = c.get_b();
    sort(tmp, 3);
    *res = tmp[1];
    delete[] tmp;
    return res;
}
void set(figure &a, double *val1, double *val2)
{
    a.set(val1, val2);
}
```

Функции 2 часть

Повтороение

Что такое инкапсуляция?

Для чего нужна?

Что такое геттер и для чего нужен?

Что такое сеттер и для чего нужен?

Какие модификаторы доступа вы знаете?

Конструкторы

конструктор

Конструктор – функция, которая задает значения объектам (само вызывающийся сеттер).

Конструкторы имеют **имена**, **совпадающие с именами классов**, и **не** имеют возвращаемых значений.

Конструктор можно **перегружать**.

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    void setAll(string test, int age)
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    void setAll(string test, int age)
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

Нашли конструктор?

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    void setAll(string test, int age)
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

Нашли конструктор? Его тут **нет**

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string test, int age) //конструктор класса Animal
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

Конструктор класса Animal

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string test, int age) //конструктор класса Animal
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

```
int main()
{
    Animal cot("Том", 12); //создали объект и в момент его создания инициализировали его поля
}
```

Вызов конструктора из main()

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string test, int age) //конструктор класса Animal
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

```
int main()
{
    Animal cot("Том", 12); //создали объект и в момент его создания инициализировали его поля
    Animal mouse; //только объявили о наличии объекта mouse без инициализации
}
```

Объявим еще один объект

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string test, int age) //конструктор класса Animal
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

```
int main()
{
    Animal cot("Том", 12); //создали объект и в момент его создания инициализировали его поля
    Animal mouse; //только объявили о наличии объекта mouse без инициализации
}
```

Скомпилируется ли программа?

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string test, int age) //конструктор класса Animal
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

```
int main()
{
    Animal cot("Том", 12); //создали объект и в момент его создания инициализировали его поля
    Animal mouse; //только объявили о наличии объекта mouse без инициализации
}
```

Скомпилируется ли программа? **НЕТ**

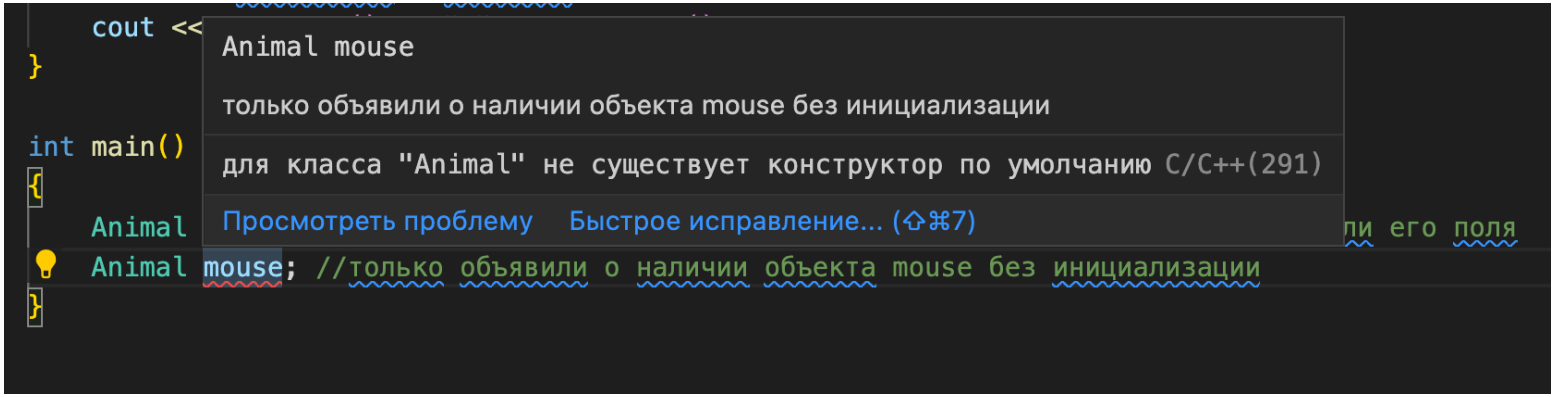
конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string test, int age) //конструктор класса Animal
    {
        name = test;
        vozr = age;
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

```
int main()
{
    Animal cot("Том", 12); //создали объект и в момент его создания инициализировали его поля
    Animal mouse; //только объявили о наличии объекта mouse без инициализации
}
```

Скомпилируется ли программа? Почему?

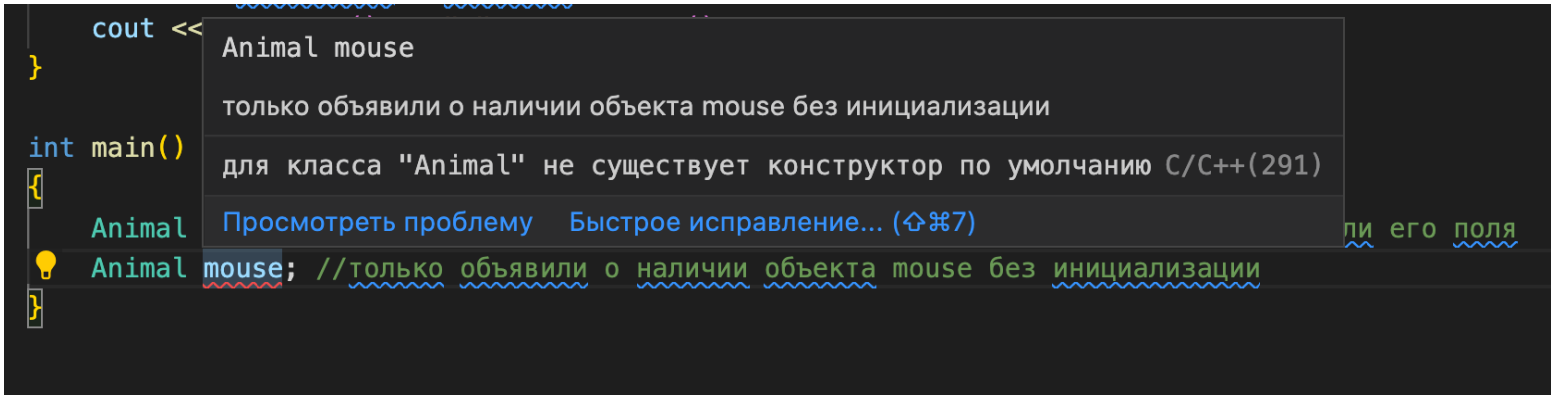
конструктор



Посмотрим на то, что говорит компилятор

конструктор

Причина: по умолчанию у каждого объекта есть неявный конструктор (конструктор по умолчанию), когда мы переопределили функцию (написали другой конструктор), то неявный конструктор больше не будет работать.



Посмотрим на то, что говорит компилятор

конструктор

Причина: по умолчанию у каждого объекта есть неявный конструктор (конструктор по умолчанию), когда мы переопределили функцию (написали другой конструктор), то неявный конструктор больше не будет работать.

Решение: переопределить вручную конструктор без аргументов.

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string test, int age) //конструктор класса Animal
    {
        name = test;
        vozr = age;
        valid();
    }
    Animal(){} //конструктор по умолчанию, который ничего не делает
```

Конструктор + конструктор по умолчанию

конструктор

```
int main()
{
    Animal cot("Том", 12); //создали объект и в момент его создания инициализировали его поля
    Animal mouse; //только объявили о наличии объекта mouse без инициализации
}
```

Посмотрим на то, что говорит компилятор

Задача

На осмотр в ветклинику хозяева приводят своих животных. У каждого животного есть кличка, вес, размер и пол.

Задача:. С помощью конструктора задайте значение объектам. Не все так просто: если пользователь не написал кличку, то должно быть написано ”спросить кличку”, вес и размер не могут быть отрицательными. (проверку реализуем в конструкторе)

Задача

```
int main()
{
    Animal cot("Том", 12);
    Animal mouse, dog("Лайка"), zebra(23);

    show(cot);
    show(dog);
    show(zebra);
    show(mouse);
}
```

Список инициализации

конструктор

Список инициализации – вариант записи конструктора, у которого значения в поля объекта заносятся напрямую, ускоряя работу.

конструктор

```
Animal(string test, int age) : name(test), vozr(age) //список иниуиализации класса Animal
{
    valid(); //тело конструктора
}
```

Конструктор со списком инициализации

конструктор

```
Animal(string name, int vozr) : name(name), vozr(vozr) //список иниуиализации класса Animal
{
    valid(); //тело конструктора
}
```

Конструктор со списком инициализации с одинаковыми полями

Задача

На осмотр в ветклинику хозяева приводят своих животных. У каждого животного есть кличка, вес, размер и пол.

Задача:. С помощью списка инициализации задайте значение объектам. Не все так просто: если пользователь не написал кличку, то должно быть написано ”спросить кличку”, вес и размер не могут быть отрицательными. (проверку реализуем в конструкторе)

конструктор

```
class Animal
{

    string name;
    int vozr;

    void valid()
    {
        if (vozr < 0)
            vozr = 0;
    }

public:
    Animal(string name = "спросить кличку", int vozr = 0) : name(name), vozr(vozr)
 //список иниуиализации класса Animal
    {
        valid();
    }

    string getN()
    {
        return name;
    }

    int getA()
    {
        return vozr;
    }
};
```

Конструктор со списком инициализации + конструктор с одинаковыми полями