

Технологии программирования

Доц. каф. «Медиаменеджмента и медиапроизводства» Евич
Л.Н.

Лекция 13. ООП в C++.

Конструктор.

Недостатком рассмотренных ранее классов является отсутствие автоматической инициализации создаваемых объектов.

Для каждого вновь создаваемого объекта необходимо было вызвать функцию типа **set** (как для класса `complex`) либо явным образом присваивать значения данным объекта.

Однако для инициализации объектов класса в его определение можно явно включить специальную компонентную функцию, называемую **конструктором**.

```
имя_класса(список_форм_параметров)  
{операторы_тела_конструктора}
```



Лекция 13. Классы в C++.

Имя этой конструктора должно совпадать с именем класса.

Конструктор автоматически вызывается при определении или размещении в памяти с помощью оператора **new** каждого объекта класса.

Пример.

```
complex(double re1 = 0.0,double im1 = 0.0)
{
    re = re1; im = im1;
}
```



Лекция 13. Классы в C++.

Конструктор выделяет память для объекта и инициализирует данные – члены класса.

Конструктор имеет ряд особенностей:

- Для конструктора не определяется тип возвращаемого значения.
- Даже тип `void` не допустим.
- Указатель на конструктор не может быть определен, и соответственно нельзя получить адрес конструктора.
- Конструкторы не наследуются.
- Конструкторы не могут быть описаны с ключевыми словами `virtual`, `static`, `const`, `mutable`, `volatile`.



Лекция 13. Классы в C++.

Конструктор всегда существует для любого класса, причем, если он не определен явно, он создается автоматически.

По умолчанию создается конструктор без параметров и конструктор копирования.

Если конструктор описан явно, то конструктор по умолчанию не создается.

По умолчанию конструкторы создаются общедоступными (public).



Лекция 13. Классы в C++.

```
//MyClass1.cpp
```

```
#include <iostream.h>
```

```
class MyClass {
```

```
int a;
```

```
public:
```

```
MyClass(); //конструктор
```

```
~MyClass(); //деструктор
```

```
void Show() const;
```

```
};
```

```
MyClass::MyClass()
```

```
{
```

```
cout << "Работает конструктор" << endl; //отладочный вывод
```

```
a = 10; // инициализация поля 'a'
```

```
}
```



Лекция 13. Классы в C++.

```
//MyClass1.cpp
```

```
#include <iostream.h>
```

```
class MyClass {  
int a;  
public:  
MyClass(); //конструктор  
~MyClass(); //деструктор  
void Show() const;  
};
```

```
MyClass::~~MyClass()
```

```
{  
    cout << "Работает деструктор" << endl; //отладочный вывод  
}
```

```
void MyClass::Show() const
```

```
{  
    cout << a << endl;
```



```
}
```

Лекция 13. Классы в C++.

```
MyClass::MyClass() {  
    cout << "Работает конструктор" << endl; //отладочный вывод  
    a = 10;      // инициализация поля 'a'  
}
```

```
MyClass::~~MyClass() {  
    cout << "Работает деструктор" << endl; //отладочный вывод  
}
```

```
void MyClass::Show() const {  
    cout << a << endl;  
}
```

```
int main() {  
    MyClass ob;  
    ob.Show();  
    return 0;  
}
```



Лекция 13. Классы в C++.

Конструкторы с параметрами


Пример

```
//MyClass2.cpp  
#include <iostream.h>
```

```
class MyClass {  
int a;  
public:  
MyClass(int x); //конструктор  
~MyClass();    //деструктор  
void Show() const;  
};
```

```
MyClass::MyClass(int x)
```

```
{  
    cout << "Работает конструктор" << endl; //отладочный вывод  
    a = x;  
}
```



Лекция 13. Классы в C++.

Конструкторы с параметрами

```
MyClass::MyClass(int x) {  
    cout << "Работает конструктор" << endl; //отладочный вывод  
    a = x;  
}
```

```
void MyClass::Show() const {  
    cout << a << endl;  
}
```

```
int main() {  
    MyClass ob(4);  
    ob.Show();  
    return 0;  
}
```



Лекция 13. Классы в C++.

Конструкторы с параметром по умолчанию

Пример

```
//MyClass2.cpp
#include <iostream.h>

class MyClass {
int a;
public:
MyClass(int x=0); //конструктор с параметром по умолчанию
~MyClass(); //деструктор
void Show() const;
};

MyClass::MyClass(int x)
{
    cout << "Работает конструктор" << endl; //отладочный вывод
    a = x;
}
```

Лекция 13. Классы в C++.

Конструкторы с параметром по умолчанию

```
MyClass::MyClass(int x) {  
    cout << "Работает конструктор" << endl; //отладочный вывод  
    a = x;  
}
```

```
void MyClass::Show() const {  
    cout << a << endl;  
}
```

```
int main() {  
    MyClass ob;  
    ob.Show();  
    return 0;  
}
```



Лекция 13. Классы в C++.

Конструкторы

Пример

```
#include <iostream.h>
```

```
class MyClass {  
int a,c;  
public:  
    MyClass(int n, int k) //конструктор с параметрами  
    { a=n; c=k; }  
  
    MyClass(int n) //конструктор с параметрами  
    { a=n; c=1;}  
  
    MyClass( ) //конструктор с параметрами  
    { a=0; c=2; }  
};
```

```
int main() {  
    MyClass ob(6);  
    MyClass ob1(3,4);  
    MyClass ob2;  
    return 0;  
}
```



Лекция 13. Классы в C++.

Методы класса

Пример

```
class MyClass {
int a;
public:
    MyClass(); //конструктор с параметром по умолчанию
    ~MyClass(); //деструктор
    void Show();
};
void MyClass::Show() {
    cout << "Привет !" << endl;
}

void Show() {
    cout << "Пока !" << endl;
}

int main() {
    MyClass ob;
    Show();
    return 0;
}
```



Лекция 13. Классы в C++.

Методы класса

Пример

```
class MyClass {
int a;
public:
    MyClass(); //конструктор с параметром по умолчанию
    ~MyClass(); //деструктор
    void Show();
};
void MyClass::Show() {
    cout << "Привет !" << endl;
}

void Show() {
    cout << "Пока !" << endl;
}
```

```
int main() {
    MyClass ob;
    Show();
    return 0;
}
```



Лекция 13. Классы в C++.

Методы класса

Пример

```
class MyClass {
int a;
public:
    MyClass(); //конструктор с параметром по умолчанию
    ~MyClass(); //деструктор
    void Show();
};
void MyClass::Show() {
    cout << "Привет !" << endl;
}

void Show() {
    cout << "Пока !" << endl;
}
```

```
int main() {
    MyClass ob;
    ob.Show();
    Show();
    return 0;
}
```



Лекция 13. Классы в C++.

Методы класса

Пример

```
class MyClass {
int a;
public:
    MyClass(); //конструктор с параметром по умолчанию
    ~MyClass(); //деструктор
    void Show() {
        cout << "Привет !" << endl;
    }
};
void Show() {
    cout << "Пока !" << endl;
}
```

```
int main() {
    MyClass ob;
    ob.Show();
    Show();
    return 0;
}
```



Лекция 13. Классы в C++.

Встраиваемые функции

Пример

```
class MyClass {
int a;
public:
    MyClass() //конструктор с параметром по умолчанию
    {
        cout << "Работает конструктор" << endl; //отладочный вывод
        a = x;
    }
    ~MyClass(); //деструктор
    void Show() {
        cout << "Привет !" << endl;
    }
};
void Show() {
    cout << "Пока !" << endl;
}

int main() {
    MyClass ob;
    ob.Show();
    Show();
    return 0;
}
```



Лекция 13. Классы в C++.

Указатель **this** указывает на объект, делающий вызов метода.

Пример

```
class MyClass {
int a;
public:
    MyClass(int x = 5) //конструктор с параметром по умолчанию
    {
        cout << "Работает конструктор" << endl; //отладочный вывод
        a = x;
        cout << "a=" << a << endl; //отладочный вывод
    }
    ~MyClass(); //деструктор
    void Show() {
        cout << "Привет !" << endl;
    }
};
```

```
int main() {
    MyClass ob;
    return 0;
}
```



Лекция 13. Классы в C++.

Указатель **this** указывает на объект, делающий вызов метода.

Пример

```
class MyClass {
int a;
public:
    MyClass(int x = 5) //конструктор с параметром по умолчанию
    {
        cout << "Работает конструктор" << endl; //отладочный вывод
        this->a = x;
        cout << "a=" << a<<"this ="<<this<< endl; //отладочный вывод
    }
    ~MyClass(); //деструктор
    void Show() {
        cout << "Привет !" << endl;
    }
};
```

```
int main() {
    MyClass ob,ob1;
        cout<<&ob;
    return 0;
}
```



Лекция 13. Классы в C++.

Задания

1. Создать класс, содержащий 3 конструктора (с параметром, с параметром по умолчанию, без параметров)
2. Создать деструктор.
3. Проиллюстрировать в программе использование всех конструкторов и деструктора.

