C++ Essential

Дополнительные возможности классов

C++ Essential

Автор курса

Кирилл Чернега

После урока обязательно

Повторите этот урок в видеоформате на ITVDN.com

Проверьте, как Вы усвоили данный материал на TestProvider.com

Дополнительные возможности классов

Дополнительные возможности классов

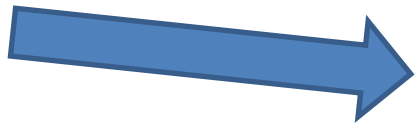
Содержание урока

1. Inline-функции и методы
2. Константные методы
3. Статические поля и методы
4. Абстрактные классы и чисто виртуальные методы
5. Дружественные классы, методы, функции
6. Перегрузка операторов

Дополнительные возможности классов

Inline-функции

```
inline void foo()
{
//some simple operations;
}
```



```
{
//some external usages
...
foo();
//some simple operations;

}
```

Дополнительные возможности классов

Константные методы

```
class A
{
public:
int sum() const {return a + b;}
// a и b не изменяются
public:
int a;
int b;
};
```

```
class A
{
public:
int sum() ~~const~~ {return a += b;}

public:
int a;
int b;
};
```

Дополнительные возможности классов

Статические поля и методы

```
class A
{
public:
int sum() const {return a + b;}
static int get_a() {return a;}
private:
static int a;
int b;
};
```

Использование:

```
A::a = 100; // инициализация вне класса

A::get_a(); //доступ через класс
A a;
a.get_a(); // доступ через объект
```

Дополнительные возможности классов

Абстрактный класс и интерфейс в C++

```
class MyAbstractClass
{
public:
virtual void function() = 0;
int some_func();
private:
int data;
};
```

Абстрактный класс (хотя бы один чисто виртуальный метод)

```
class MyInterface
{
public:
virtual void function() = 0;
virtual int sum() = 0;
};
```

Интерфейс (все члены класса – чисто виртуальные методы)

Дополнительные возможности классов

Ключевое слово friend

```
class AClass
{
private:
   int data;
   friend BClass;
// BClass является дружественным для класса AClass (связь не взаимная)
};
```

```
class BClass // имеет доступ ко
// всем данным класса AClass
{
public:
   void func(AClass& a)
   {
      cout << a.data;
   }
};
```

Дополнительные возможности классов

Перегрузка операторов

Является примером статического полиморфизма.

Не все операторы можно перегрузить.
Нельзя перегружать операторы:
1. ?: (тернарный оператор);
2. :: (оператор расширения области видимости);
3. . (доступ к полям);
4. .* (доступ к полям по указателю);
5. sizeof, typeid и операторы приведения (*_cast)

Количество операндов, порядок выполнения и ассоциативность операторов определяется стандартной версией.

Нельзя перегрузить несуществующий оператор (придумав свой, к примеру: !! Или &%)

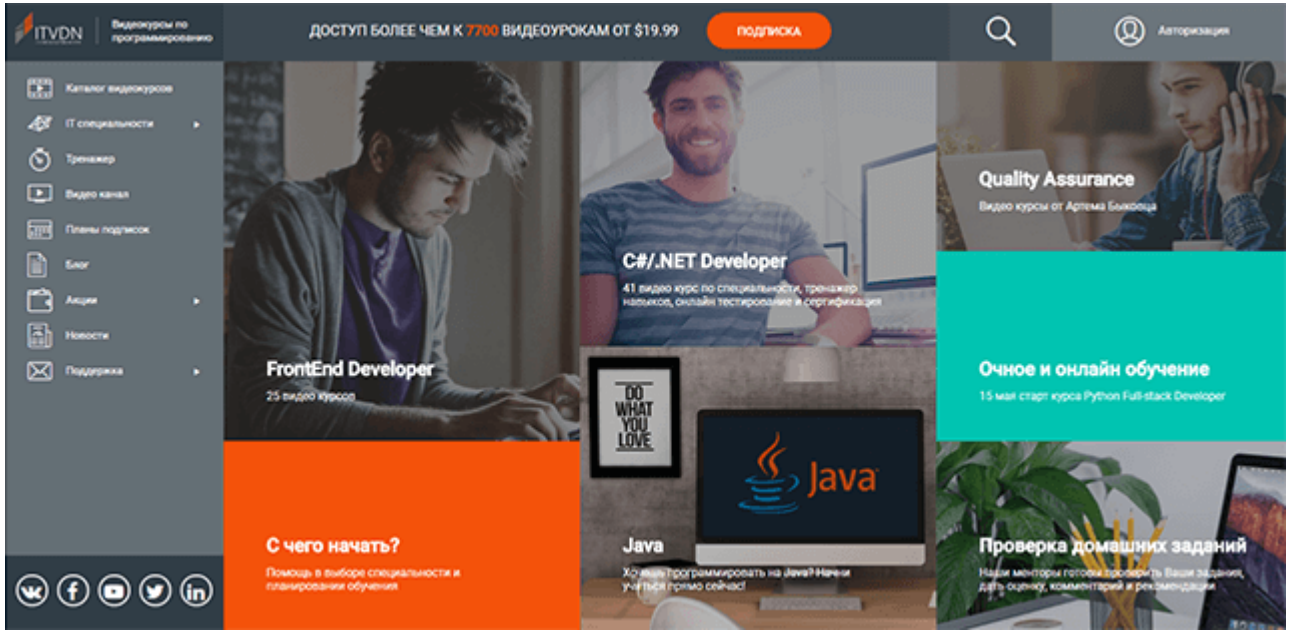
```
class Cat;
class Dog;
class Matrix;


//usage
Cat cat1, cat2;
Dog dog1, dog2,
Matrix a, b;

cat2 = cat1;
cout << cat1 << dog1;
cout << a * b + b;
```

Смотрите наши уроки в видеоформате

ITVDN.com



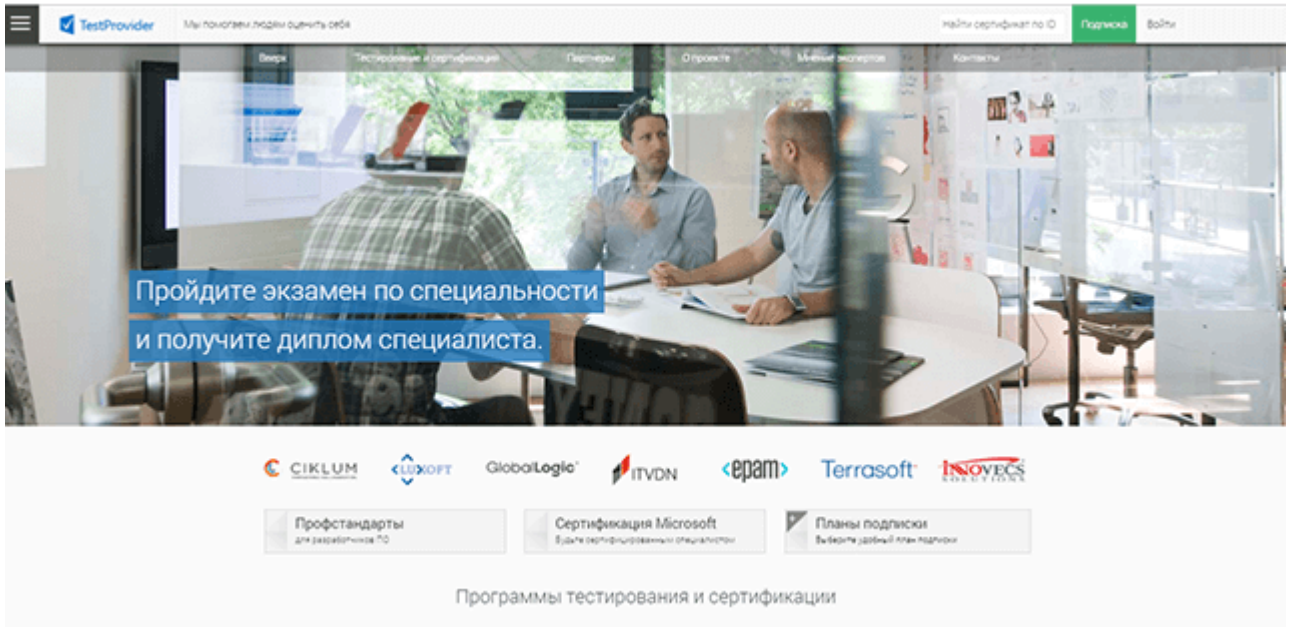
Посмотрите этот урок в видеоформате на образовательном портале ITVDN.com для закрепления пройденного материала.

Курсы записаны сертифицированными тренерами, которые работают в учебном центре CyberBionic Systematics, и другими высококвалифицированными разработчиками.

ITVDN
IT VIDEO DEVELOPERS NETWORK

Проверка знаний

TestProvider.com



TestProvider – это online-сервис проверки знаний по информационным технологиям. С его помощью Вы можете оценить Ваш уровень и выявить слабые места. Он будет полезен как в процессе изучения технологии, так и для общей оценки знаний IT-специалиста.

После каждого урока проходите тестирование для проверки знаний на TestProvider.com

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.

TestProvider

Q&A

Информационный видеосервис для разработчиков программного обеспечения

ITVDN

IT VIDEO DEVELOPERS NETWORK

TestProvider PROMETRIC TEST CENTER CyberBionic systematics Microsoft Partner Silver Learning Windows Azure Cloud Partner EBA European Business Association