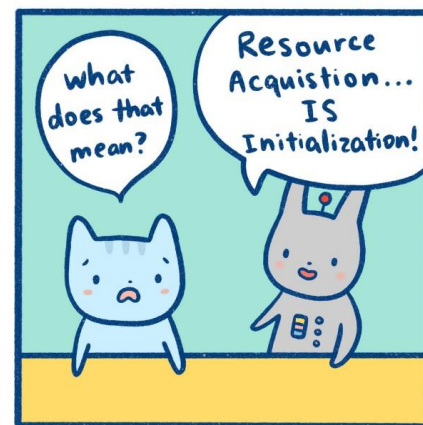
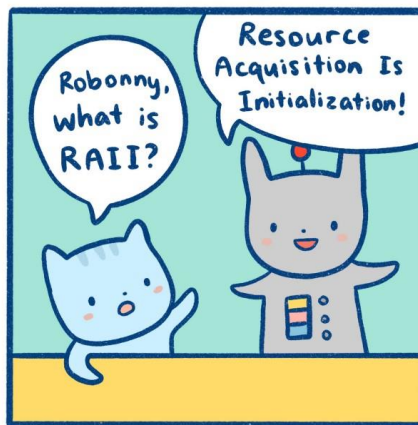


Contoso Ltd.

# CP 5

Деструктор. Освобождение ресурса.



# Деструктор

## Понятие деструктора

- Деструктор – это особый метод класса, который вызывается при уничтожении объекта класса. Мы вправе не объявлять деструктор в классе, тогда С++ применит к объекту тривиальный деструктор (или деструктор по-умолчанию). Тривиальный деструктор будет один за другим уничтожать все поля класса.
- При этом, если у класса уничтожаемого объекта есть свои «родители», то деструктор уничтожает поля класса от наследника – к родителям (конструкторы при наследовании вызываются в обратном порядке – сначала создаются родители, а потом - наследники).
- У самого базового класса **деструктор обязан быть виртуальным**.  
О том, зачем это нужно, уже сказано в лекции к СР№6.

## Деструкторы и наследование

Порядок вызова деструкторов при уничтожении объекта (сверху вниз, от частного – к общему). Первым будет вызван деструктор класса «Тигр», затем будут убиты «Кошачьи» и уже в конце – «Млекопитающие». Заметим, что для того, чтобы деструкторы **всегда** вызывались корректно, у класса «Млекопитающие» деструктор должен быть виртуальным.



# Деструктор

## RAII

- Нетривиальный деструктор нужен, если Вы хотите изменить логику уничтожения объекта. Обычно бывает достаточно освобождения памяти, но что будет, если при создании объекта конструктор захватывает некоторый ресурс?
- В C++ существует принцип RAII (Resource Acquisition Is Initialization – захват ресурса есть инициализация). Суть принципа RAII в том, что мы привязываем время жизни ресурса, который был запрошен при создании объекта (в конструкторе), к времени жизни объекта, к которому этот ресурс привязан, ресурс освобождается вместе с уничтожением объекта.  
При этом запрашиваемый ресурс нам обычно выдаёт ОС.
- Ресурс – понятие довольно расплывчатое, в нашей лабораторной работе ресурсом является файл.

```
public:
    // Default constructor
    CoffeeMachine(const char* fileName = "coffee.txt")
    {
        // Resource acquisition (open file)
        file.open(fileName);
        if (file.is_open()) {
            cout << "Initialization from file" << endl;
            // Initialize object using data from file
            file >> pressure;
            file >> temperature;
            file >> colour;
            file >> numberOfCups;
            file >> canMakeCappuccino;
            file >> price;
            file >> model;
        } else {
            // Default values for initialization
            cout << "File not found. Default initialization." <<
endl;

            canMakeCappuccino = true;
            price = 5000;
            pressure = 15;
            colour = "black";
            temperature = 0;
            numberOfCups = 1;
            model = "Vitek VT-1525";
        }
    }
}
```



# Деструктор

## RAII

- В данном примере я добавил ещё одно поле для класса CoffeeMachine –

```
fstream file; // Type of resource for  
acquisition.
```

Т.е. одним из полей класса теперь является файловый поток

**Примечание:** не будем писать методы для валидации входных данных и проверки их целостности, поскольку в регламенте это не требуется.

```
public:  
    // Default constructor  
    CoffeeMachine(const char* fileName = "coffee.txt")  
    {  
        // Resource acquisition (open file)  
        file.open(fileName);  
        if (file.is_open()) {  
            cout << "Initialization from file" << endl;  
            // Initialize object using data from file  
            file >> pressure;  
            file >> temperature;  
            file >> colour;  
            file >> numberOfCups;  
            file >> canMakeCappuccino;  
            file >> price;  
            file >> model;  
        } else {  
            // Default values for initialization  
            cout << "File not found. Default  
initialization." << endl;  
            canMakeCappuccino = true;  
            price = 5000;  
            pressure = 15;  
            colour = "black";  
            temperature = 0;  
            numberOfCups = 1;  
            model = "Vitek VT-1525";  
        }  
    }  
}
```

# Деструктор

## Освобождение ресурса

- Освобождение ресурса происходит в деструкторе. Деструктор объявляется следующим образом
- ```
public:  
...  
~ClassName()  
{  
...  
}
```
- Где **ClassName** — имя класса  
Деструктор должен быть публичным.

```
// Release acquired resource while calling  
destructor  
~CoffeeMachine(){  
    // Close file stream to release resource  
    if (file.is_open()) file.close();  
}
```

