



Cvičení C++

12.11.2018

faltin@ksi.mff.cuni.cz



Píškivorky feedback

- Nepoužívat not, and, or místo !, &&, | |
- Používat unsigned typy, kde to má smysl
 - size_t, uint32_t, unsigned int, ...
- Rozdělovat na hlavičkové (*.hpp/*.h) a zdrojové (*.cpp) soubory
 - Třída ~ jedna dvojice souborů
- const funkce
- Nepoužívat continue/break/goto/exit
- **Zkuste sám naprogramovat**



Domácí úkol

- **do 18.11:** přihlásit do ReCodexu - <https://recodex.mff.cuni.cz>
 - Pokud jsou problémy, napsat mail



R-value reference (1/2)

```
class Movable {  
    std::unique_ptr<LargeObject> ptr;  
public:  
    Movable(int arg1, const std::string &arg2) :  
        ptr(std::make_unique<LargeObject>(arg1, arg2)) {}  
  
    Movable(Movable &&rhs) : ptr(std::move(rhs.ptr)) {}  
  
    Movable &operator=(Movable &&rhs) {  
        if (*this != rhs) {  
            this->ptr = std::move(rhs.ptr);  
        }  
        return *this;  
    }  
  
    Movable(const Movable &) = delete;  
    Movable &operator=(const Movable &) = delete;  
};
```



R-value reference (2/2)

```
class matrix_holder {  
    std::vector<std::unique_ptr<matrix_type>> data_;  
  
public:  
    void push_back(std::unique_ptr<matrix_type> &&matrix) {  
        data_.push_back(std::move(matrix));  
    }  
};  
  
auto matrix1 = std::make_unique<matrix_type>(size_x, size_y);  
matrix_holder mh;  
mh.push_back(std::move(matrix1));
```



Úkoly

■ Vlastní implementace `std::vector`

1. Pouze pro `std::string`
 1. `push_back()`, `size()`, `pop_back()`, `front()`, `back()`, `insert()`, `clear()`
 2. Automaticky správně reallokuje
2. Pro libovolné objekty které dědí od interface `storable`
3. Pomocí templates vytvořit obecný `std::vector<T>`
4. Vytvořit `std::array<T, size_t>`
5. `std::vector` pro libovolné objekty optimalizovaný na rychlost