Конспект 14.02

HotKeys:

```
browser-like:   chrome, vscode
browser:        chrome, firefox
editors:        chrome, word
code-editors:   vscode, PyCharm
vscode:         vscode
```

- Windows

```
win + d  - закрыть, открыть все окна
win + e  - проводник
win + r  - быстрый промпт в консоль (например запуск ssh)
win + P  - тип работы с вторым монитором(проектором)
```

- HotKeys

```
ctrl+shift+n - новое окно (browser-like)
ctrl+s - сохранить (editors)
alt+tab - переключить окно (win)
shift + arrows - выделение (editors)
ctrl + arrows - указатель через слово (editors)
ctrl + shift + arrows - выделить слово (editors)
shift + alt + arrows up/down - скопировать строку (code-editors)
ctrl + delete - удалить слово после указ. (editors)
ctrl + backspace - удалить слово перед указ. (editors)
ctrl + / - разбить окно на два (vscode)
ctrl + tab - переключить окно (broswer-like)
alt + arrows - последнее/следующее изменение (code-editors)
alt + arrows - следующая/предыдущяя страница в истории окна (browsers)
ctrl + w - закрыть окно (broswer-like)
ctrl + b - закрыть/открыть левую панель (vscode)
ctrl + shift + p - открыть панель быстрого доступа (vscode)
ctrl + ` - открыть консоль (vscode)
ctrl + l - выделить строку (vscode)
shift + delete - удалить всю строку (vscode)
ctrl + H  - поиск + замена (editors) (в vscode есть regex)
alt + click - несколько курсоров (vscode)
ctrl + arrows up/down - пролистать вверх/вниз (vscode)
```

// код → прекомп. → ассемб. → линковка → exe

```
clang++-18 main.cpp -O main && ./main
```

// код → прекомп. → ассемб. → линковка → exe

```
#include <bits/stdc++.h> — 0,3 сек.
```

↪ импорт ВСЕХ библиотек

ООП:

```
struct Pair {
    int a, b;
};
```

структура

```
class OrientPair {
public:
    int a, b;
    int first {
        return a;
    }
    int second {
        return b;
    }
}
```

класс

поля класса

методы класса

КЛАСС:

```
class Citizen {
private:
    bool is_alife;
    std::string name;
    City* home;
public:

    Citizen(std::string name_) : name(name_), is_alife(true), home(nullptr) {
        std::cout << "We created a person!\n";
    }

void print() const {
    if (is_alife) {
        std::cout << name;
        if (home) {
            std::cout << " is live in city " << home->GetTitle();
        }
    } else {
        std::cout << name << " is Dead";
    }
    std::cout << '\n';
}

void SetHome(City *new_home) {
    if (home == new_home) {
        std::cout << "No relocation\n";
    }
    home = new_home;
}
```

поля класса

конструктор

методы

```
~Citizen() {
    is_alife = false;
    std::cout << name << " is Dead\n";
}
```

} деструктор

Инкапсуляция (сокрытие реализации):

- private – объект "снаружи" не доступен, изменять его можно только через методы класса
- объявление перед определением

  объявление – объект становится видимым для компилятора

  определение – создание "тела" функции
- разделение объявления и определения
- "геттеры" и "сеттеры"