3. Ссылки, функции, константность

Гит

Программирование и алгоритмизация

Практические занятия

БИВТ-24-17

Надежда Анисимова

ms teams m2102039@edu.misis.ru

Проверка себя

1. На что указывает тип?
2. Что будет в результате в переменных?

```
int a = 1;
int b = 5;

b = a++; // b = ?, a = ?
b = ++a; // b = ? a = ?
```

3. Какой это оператор >>

Ссылки
Reference

01

Напоминалка

- Переменная - именованное хранилище, именованный участок памяти под определенный тип

- Ссылка - альтернативное имя переменной (объекта), псевдоним

Оператор формата **&d**, где d - объявляемое имя

```
int value = 50;
int& refValue = value;
```

Требования для создания/ использования ссылок

- Ссылку необходимо сразу инициализировать

```
int& refValue = value; // ok
int& refValue; // ошибка
```

- Ссылка должна всегда ссылаться на какое-то место в памяти
- Ссылка не может быть null
- Тип ссылки должен совпадать с типом ссылаемого объекта

```
int value = 50;
float& refValue = value; // ошибка
```

Спецификатор const

02

const (qualifier const) -спецификатор, указывающий, что значение переменной не должно и не будет изменяться

Например, фиксированный размер буфера

```
const int bufferSize = 512;
```

- Компилятор обычно заменяет в коде константу на значение
- Нельзя изменить значение после инициализации;
- Но можно инициализоровать динамически (в ран-тайме):

```
int i = 0;
cin >> i;
int const k = i;
```

здесь инициализация в ран-тайме

Как константность ограничивает переменную?

! Можно использовать только операции, которые не изменяют объект

При этом значение константы можно присвоить другой переменной и манипулировать ей

```
const int val = 50;
const int val = 60; // ошибка

int differrentVal = val;
 differrentVal = 60; //ok
```

Когда использовать?

- Если важно, чтоб переменная далее не менялась
- Если нет необходимости изменять переменную далее в коде

- Для избежания багов – компилятор ругается если изменять константу
- Код лучше читается!
- Могут быть применены оптимизации компилятором

Ссылка на константу (reference to const)

Ссылка - альтернативное имя, поэтому сама по себе константной быть не может, но может ссылаться на объект, свойства которого станут константными

```
// ok
const int val = 50;
const int& ref = val;

int val = 50;
const int& ref = val;
```

```
// ошибка
const int val = 50;
int& ref = val;
```

L- value и R - value

- **L- value -** выражение, занимаемое определенное мето в памяти (есть адрес)

- **R- value -** то что не L value, то есть выражения без определенного места в памяти

```
int val = 8; // ok
8 = val; // ошибка
```

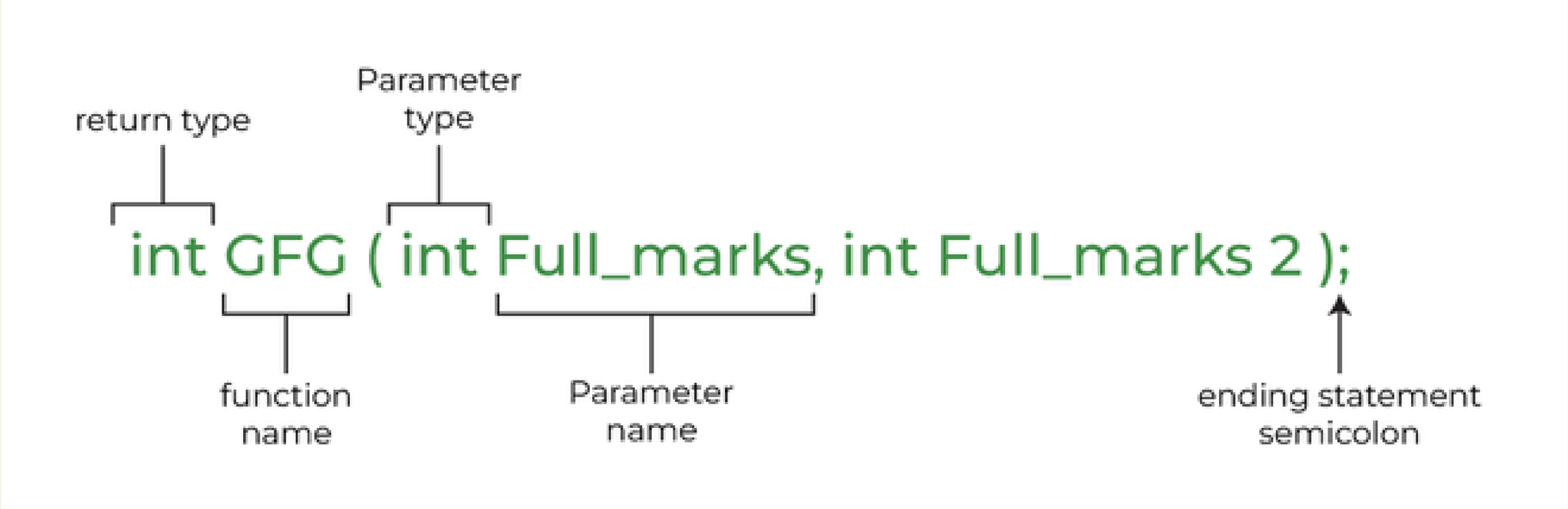
НО L-value - **это не просто выражение слева от присваивания**!

```
const int val = 10; // ok, val это L-value
a = 10; // ошибка, нельзя присвоить значение
```

Функции

03

Функции – это обособленный блоки кода, выполняющие определенные операции



- return type – тип возвращаемого значения, void – ничего
- function name – имя функции
- parameter type, name – тип и имя передаваемых в функцию параметров
- в фигурных скобках {} – тело функции

Когда выделить код в новую функцию?

- Когда текущая функции слишком большая, трудно понимать, что она делает, а так же у неё много зон ответственности

  Например, если в одной функции происходит
  - Сложное считывание данных
  - Чистка текста от ненужных символов
  - Обработка текста
  - …

- Хорошо-читабельный код – это самодокументируемый код

Если будут понятные и логичные имена переменных/функций, то код легко читать

Имена переменный – существительные

Имена функций – глаголы

Параметры функции

- Обычные переменные **(int value, myType value2)**
- Константные параметры **(const int value, const myType value2)** - неизменяемые
- Объекты переданные по сслыке **(int& value, myType& value2)** - так объект не копируется внутри функции и анипуляции происходят с тем же объектом, что и снаружи функции
- По константной ссылке **(const int& value, const myType& value2)** - если необходимо только чтение объекта, который снаружи (read-only access)

Не миксуйте способы передачи , если можно ввести в заблуждение

(std::vector<int> x, std::vector<int> const& y)

- Можно передать дефолтные параметры

(int value, myType value2 = 2) После дефолтного - все остальные дефолтные должны быть

Перегрузка функций (overloading)

Функции с одинаковым именем, но разными типами параметров

- Нельзя перегрузить только разным возвращаемым типом

```
int abs (int value);
int abs (double value);
```

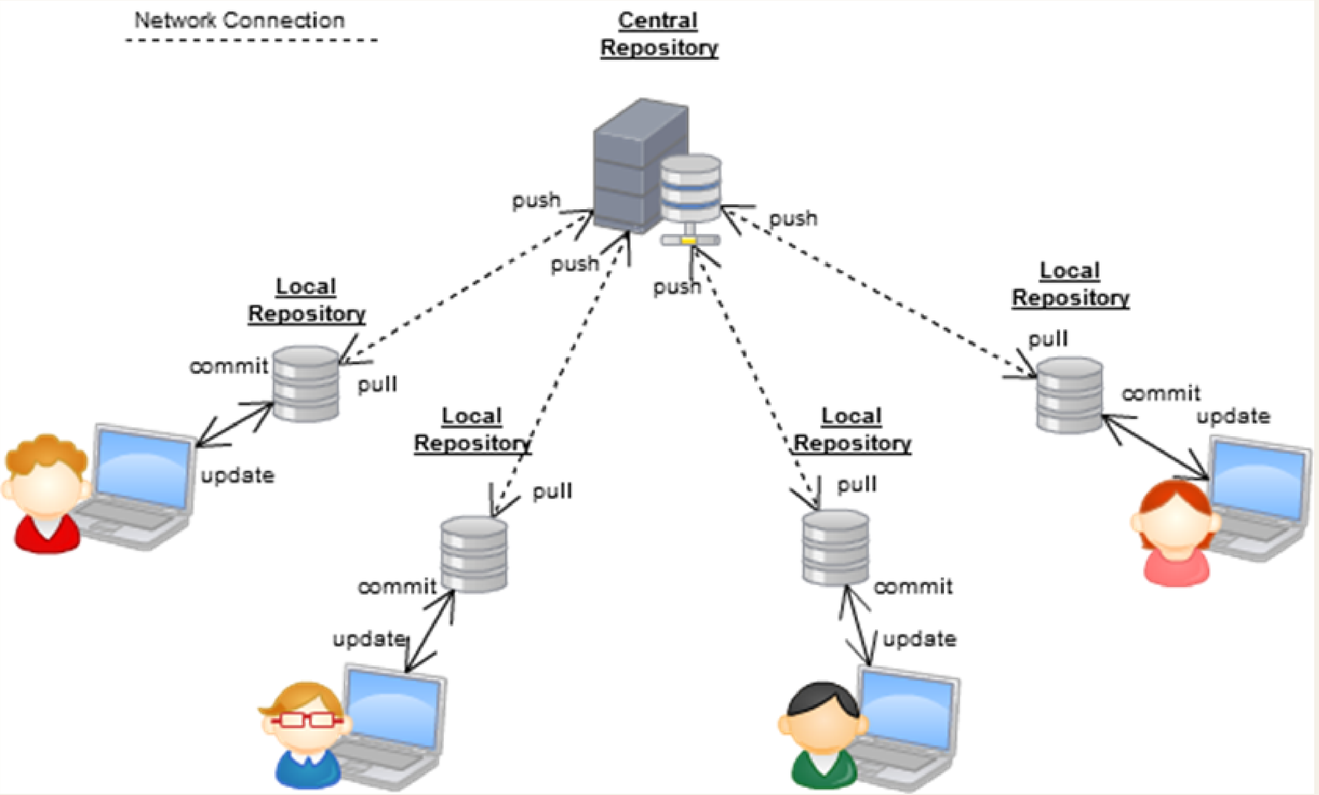
```
// вызов
res = abs(-10);
res = abs(-10.5);
```

Git

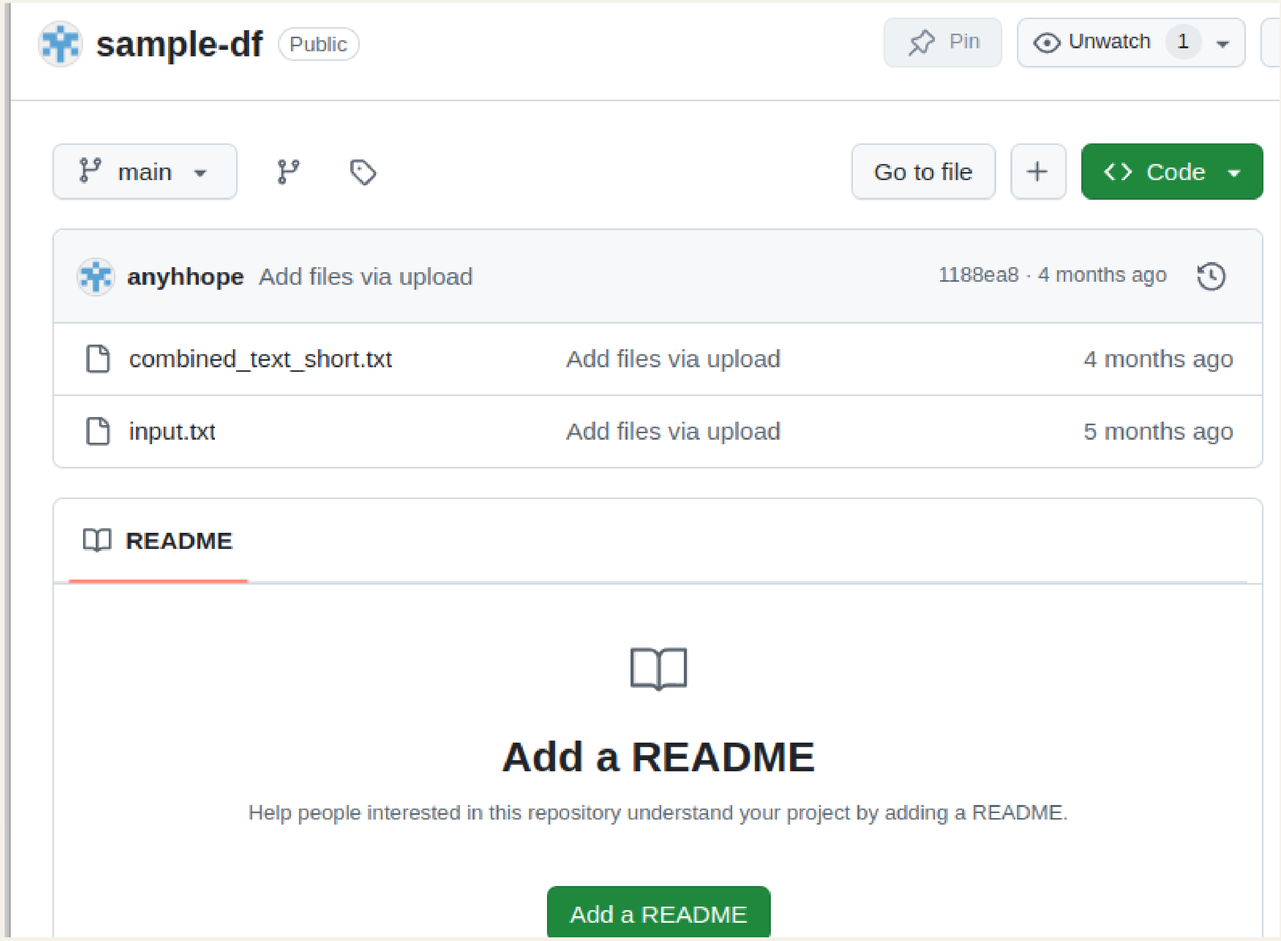
04

Git – система контроля версий, которая помогает отслеживать изменения в коде/других файлах

Github – сервис, основанный на системе контроля git



Репозиторий – хранилище с файлами проекта и информации про их изменение, по факту папка с файлами



Коммит(commit) – это команда в системе контроля версий Git, которая фиксирует изменения в репозитории

- При коммите делается снепшот теущего состояния проекта
- Как будто все файлы скопировали и вставили и зафиксировали, но лучше, ведь при коммите может только изменение фиксироваться

Ветка в Git — это последовательность коммитов, которые упорядочены по времени

Как создать репозиторий ?

1. Регистрация
GitHub https://github.com/

2. Создание репозитория

- private/public
- Добавить README.md - файл-описание, оформляется по правилам синтаксиса Markdown
- Добавить `.gitignore` (выбрать C++)

Как клонировать репозиторий ?

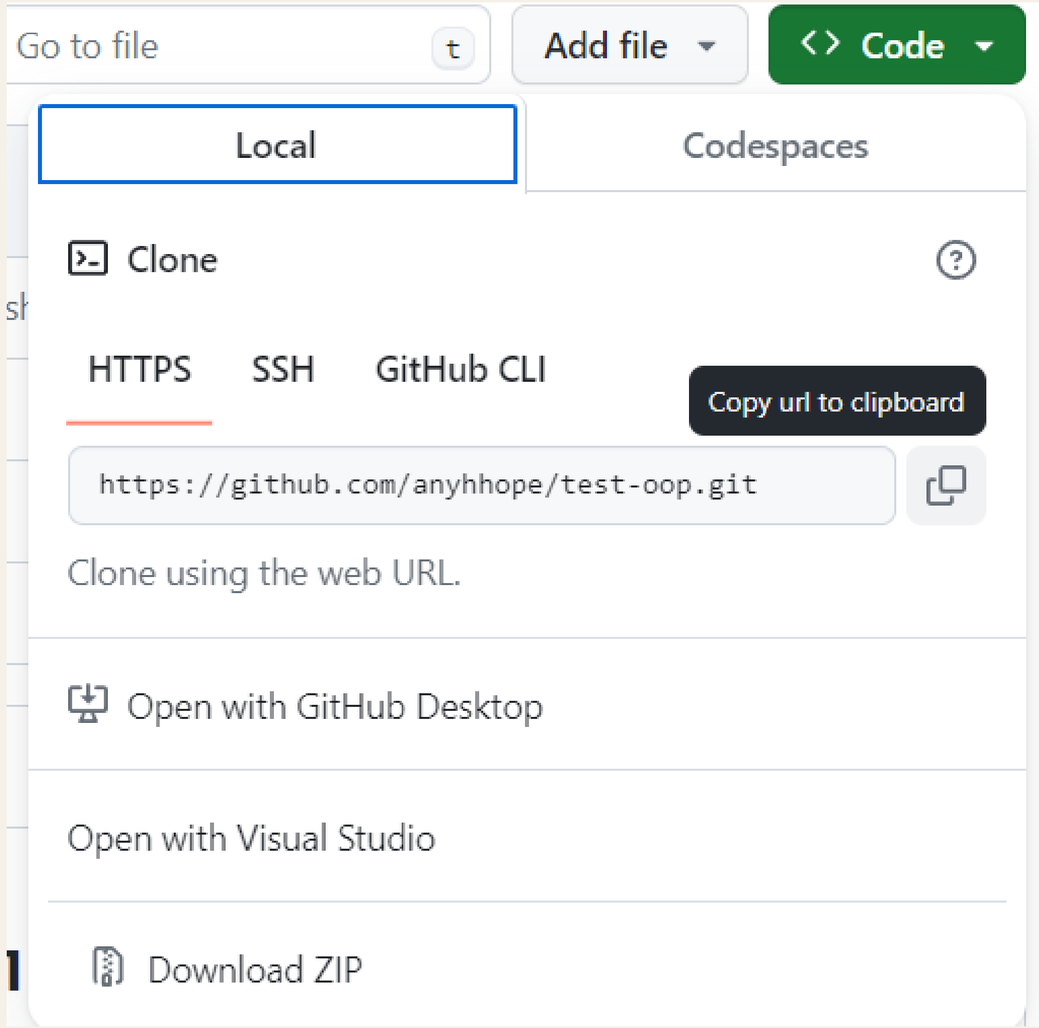
3. Клонирование репозитория на устройство

Репозиторий в виде папки у вас на компьютере называется локальный репозиторий.

Репозиторий, загруженный на GitHub, называется удалённый репозиторий.

Когда вы клонируете себе на компьютер репозиторий с GitHub, вы создаёте **локальную копию удалённого репозитория**.

Как клонировать репозиторий ?

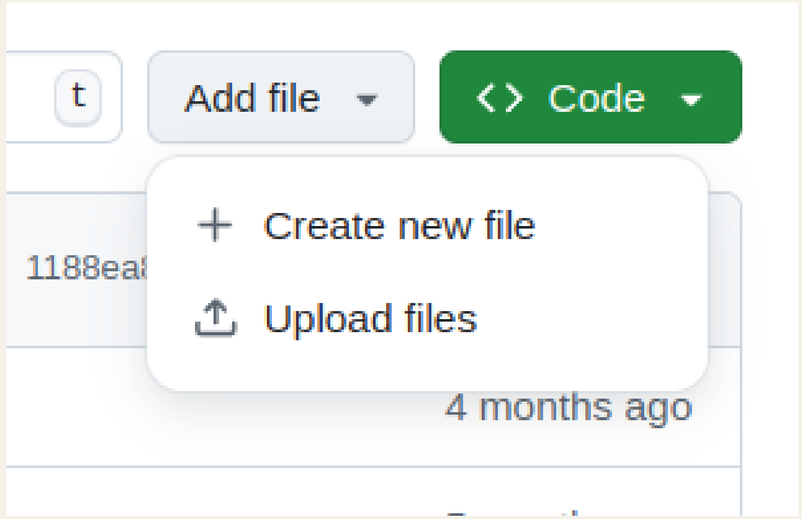


git clone https://github.com/<nickname>/<repository_name>.git

4. **Сохранить изменения и загрузить на** GitHub

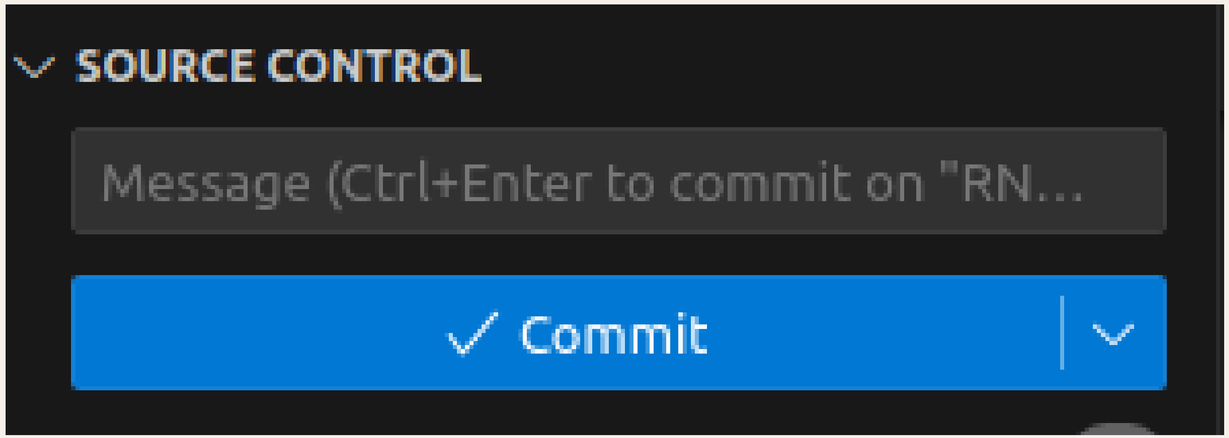
a) По кнопкам через сайт

- Add file справа вверху
- Выбираете Upload files
- Перетягиваете файлы и папки с проектом



Не делайте так, но опция есть

b) Через кнопки в IDE/ редакторе кода



Так многие делают, но лучше через терминал, чтобы навык оттачивать)

c) Через терминал

```
git clone https://github.com/<nickname>/<repository_name>.git
git add .
git commit -m 'Комментарий к коммиту'
git push origin main
```

- git add . (с точкой в конце) - Индексация изменения
- git commit -m "пишете что изменили" - Коммитите изменения (сохраняете версию кода)
- git push origin main - Загружаете файлы в репозиторий здесь main - название ветки репозитория, в которую пушите

Подробнее команды git

Домашнее задание

Контест ДЗ2