**Python**

Язык программирования

**#**  комментарий

|  |  |  |  |
| --- | --- | --- | --- |
| **Pycharm**  Среда программирования | **Git**  Контролер кода (программы) | **Github**  Среда для обмена программами и совместной работы | Replit  Онлайн среда программирования |
|  |  | имя  аккаунта:  пароль: | имя  аккаунта:  пароль: |

**Как загрузить ваш проект на сайт Github?**

**1**. На главном экране находим кнопку “New” и нажимаем на нее. Если ее нет, то переходим на вкладку “Your repositories” через иконку вашего профиля и уже в этой вкладке нажимаем на кнопку “New”.

**2**. Откроется окошко для создания вашего репозитория;

**3**. Здесь необходимо дать любое название в строке “Repository name”, например, lesson0. Также добавить описание в строке “Description”. Описание требуется для того, чтобы было легче понять для какого задания был сделан тот или иной репозиторий, когда вы их просматриваете;

**4**. Затем выбираем слово “Public” - публичный репозиторий для того, чтобы вы могли поделиться им с преподавателем.

**5**. В конце нажимаем на “Create repositоry” , тем самым заканчивая создание репозитория;

**6**. Следующим шагом мы должны прикрепить в него уже существующий файл: в строке с адресом созданного репозитория нажимаем на “uploading an existing file”.

**7**. Откроется окошко для загрузки файла, нажимаем на “choose your files” и выбираем любой файл, который нас интересует.

**8**. Нажимаем на “Commit changes”, тем самым ваш файл добавится в репозитоий.

Теперь вы можете скопировать ссылку на репозиторий с адресной строки и прикрепить ее к ответу на домашнее задание, либо поделиться ею с товарищами.

Если возникает необходимость дополнить этот репозиторий, нажимаем “Add file” и далее “Upload files” и снова повторяем процедуру с загрузкой.

**typ()**  - определение типа данных

**Int** –целое число

**float** – дробное число

**str** – строка, буквы, символы, предложения

**bool** – логические (**true**-истина, **false**-ложно)

**list** - список

**+** - сложение (переменных и строк)

**-** - вычитание (переменных)

**\*** - умножение

**/** - деление(в результате дробное число) # 6/3=2.0

**\*\***- возведение в степень

**=** - присваивание, приравнивание, назначение # a = 4 x=y

**//**- целочисленное деление (в результате целое число) # 5//2=2

**%** - получение остатка от деления # 5%2=1

**< > <= >=** - сравнение переменных # результат типа bool

**==** - равны ли переменные # результат типа bool

**!=** - не равны переменные # результат типа bool

**and** – логическое **И** # результат типа bool

**or** – логическое **ИЛИ** # результат типа bool

**int(str)** – превращение строки в число # только если цифры

**str(int)** – превращение числа в строку

**Переменные**

Название переменной может содержать буквенные и цифровые символы, а также подчеркивания, но начинаться оно обязательно должно именно с буквы или подчеркивания

Snake case, в нем для разделения слов используются нижние подчеркивания

# peremen\_iz\_neskol\_slov

Camel case. При таком стиле слова разделяются верхним регистром, то есть каждое новое слово пишется с заглавной буквы

# PeremenIzNeskolSlov

**Динамическая типизация** - тип данных внутри переменной может меняться

# пример

Name = ‘Denis’

Name = 587

**Print**

Оператор языка для вывода информации на экран # напечатает только то что в скобках

# x = 27 y = 13 name = ‘Denis’ age = ‘17’

Print() -- # пустая строка

Print(x) -- 27

Print(y) -- 13

Print(name) -- Denis

Print(age) -- 17

Print(x,y,name,age) -- 27 13 Denis 17

Переменные должны быть разделены запятыми, при выводе автоматически разделяются пробелом.

set=” ” – вставка специального разделителя (знака, пробела, убрать пробел и т.п.)

Print(x,y,name,age,set=”,”) – 27, 13, Denis, 17

Print(x,y,’name’,age,set=”,”) – 27, 13, name, 17 # кавычки выделяют текст который НАДО напечатать, без кавычек это переменная, значит печатает её значение

Print(x,y,’Denis’,age,set=”,”) – 27, 13, Denis, 17

Print(f”Теперь пишем так как хотим получить на экране а переменные берем в фигурные скобки! {x}, {y} name: {name}, возраст {age}”) – Теперь пишем так как хотим получить на экране а переменные берем в фигурные скобки! 27, 13 name: Denis, возраст 17

Внутри скобок можно проводить различные математические и логические действия над переменными, слияние строк, преобразование типов данных.

**Функция**

# подпрограмма

**Функция** - это именованный участок кода на языке программирования, который можно вызывать по имени.

def – оператор обьявления функции

def *имя\_функции*():

оператор1

оператор2

…

оператор№

оператор

закончился отступ – закончилась фунция. Для вызова функции в программе просто пишем  *имя\_функции*(). Со скобками!

Если переменная создана вне функции, она называется **глобальной**, а если внутри - **локальной**.

**# это глобальная переменная**

**a = 5**

**def some\_function():**

**# это локальная переменная**

**b = 5**

При работе с переменными следует помнить об **области видимости переменных**:

1. Локальные переменные видны *только внутри функции*, в которой они были созданы;
2. Глобальные переменные видны внутри (и вне) всех функций;
3. Нельзя *менять* глобальную переменную внутри функции.
4. Если все-таки нужно поменять глобальную переменную внутри функции, можно использовать ключевое слово global, но делать так в больших проектах **не стоит**. Чуть позже мы обсудим почему, а пока позволим себе немного "шалости".

**4. Используем ключевое слово global.**

**a = 5**

**def some\_function():**

**global a**

**a = a + 10**

**some\_function()**

**# выведет 15**

**print(a)**