Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

Факультет информационных технологий и программирования

Дисциплина «Введение в цифровую культуру и программирование»

Лабораторная работа №1

Настройка IDE

Выполнил студент группы № M3109

Хлучин Георгий Владимирович.

Подпись:

Проверил:

Повышев Владислав Вячеславович

Санкт-Петербург

2021

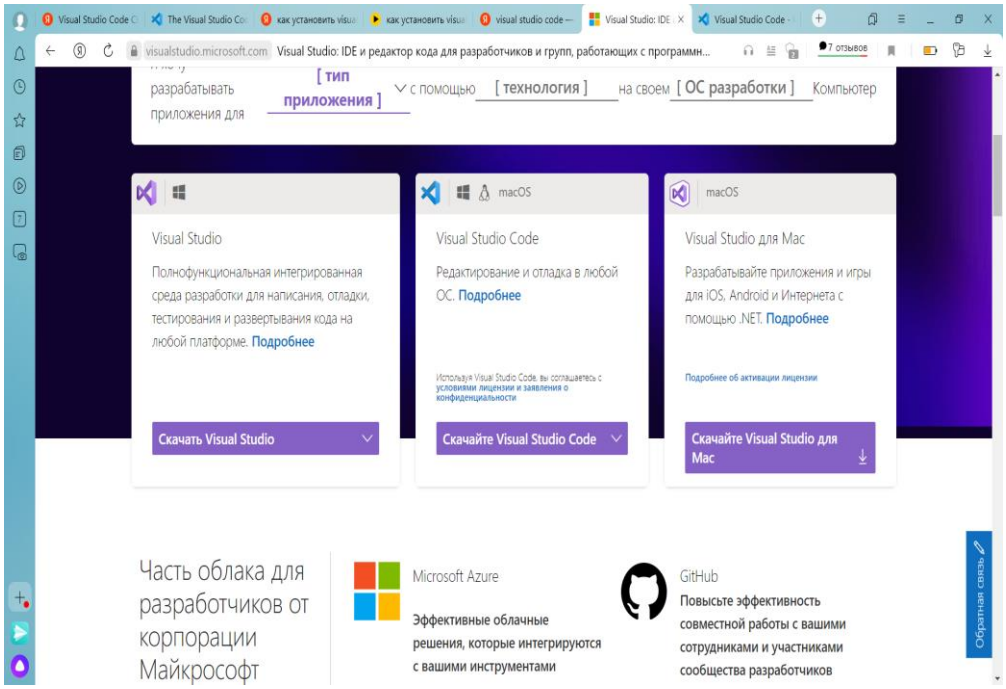
Лабораторная работа №1.

Введение в цифровую культуру и программирование.

Пункт №1.

(установка Visual Studio Code)

1. Заходим на официальный сайт → https://visualstudio.microsoft.com/ru/
2. Выбираем нужную нам среду разработки, а также нашу операционную систему.
3. Устанавливаем.

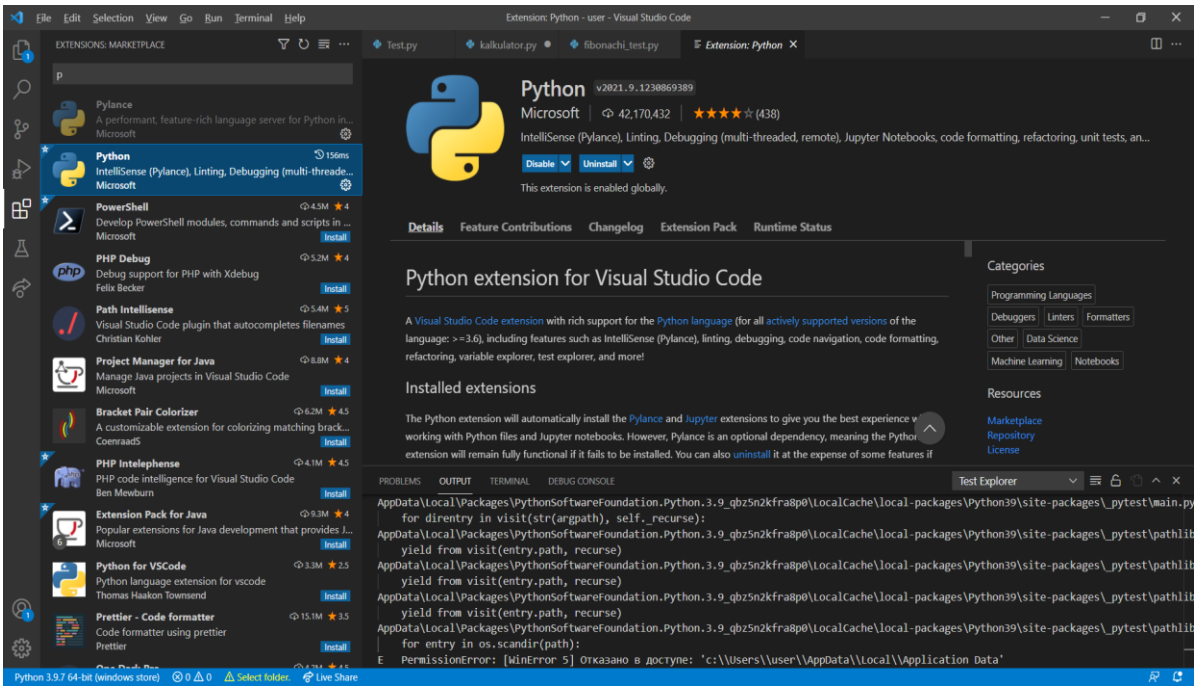


Пункт №2.

(Конфигурация для запуска и отладки одного языка программирования)

1. Думаем на каком языке мы будем писать проект. (Я выбрал Python).
2. Заходим в EXTENTIONS или нажимаем Ctrl + Shift + X.

3. Выбираем нужный плагин.



4. После того как мы сконфигурировали нужный нам язык программирования, приступаем к созданию проекта.

Пункт №3.

(написание проекта)

(в данном случае это будет консольный калькулятор)

1. Определяемся с проектом.
2. Пишем код. (мой код представлен ниже)
3. Смотрим что получилось.

Пункт №4.

(написание тестов к получившемуся проекту)

1. Думаем над тем какие данные может ввести пользователь. (*тесты можно начинать писать уже на третьем пункте, параллельно написанию проекта)
2. Пишем тесты.
3. Проверяем как программа проходит данные тесты.

Пункт №5.

(Visual Studio Code CLI и написание сниппетов)

1. Разберемся с тем, что такое Command Line Interface (CLI).

Итак, CLI – это интерфейс командной строки, который помогает управлять запуском редактора. С его помощью вы можете открывать файлы, устанавливать расширения, изменять язык отображения и выводить диагностику с помощью переключателей (параметров командной строки).

Какие могут быть команды:

1.1) code –help После ввода данной команды вы увидите версию, пример использования и список аргументов командной строки. Можно использовать сокращение code -h.

1.2) code –version После ввода данной команды вы увидите версию кода, идентификаторы фиксации на GitHub и архитектуры. Можно использовать сокращение code -v.

1.3) code –new-window (сокр. code -n) открывает новый сеанс VS code вместо восстановления предыдущего.

1.4) code –reuse-window (сокр. code -r) принудительно открывает файл или папку в последнем активном окне.

1.5) code –goto (сокр. code -g) При использовании с file:line[:character] открывает файл в указанной строке (символ указывать не обязательно)

1.6) code –diff (сокр. code -d) открывает редактор различий файлов (для выполнения этой команды вам потребуется два файла).

1.7) code –locale <locale> устанавливает язык отображения (языковой стандарт). (например: en-US или zh-TW).

1.8) code –file Указываете файл который нужно открыть, если файла не существует, то он будет создан и помечен как отредактированный. Также вы можете указать несколько файлов разделив их пробелами.

1.9) file:line[:character] используется с code -g.

1.10) folder Указываете имя папки которую нужно открыть. Если вы укажите насколько папок то будет создано новое рабочее пространство с несколькими корнями.

2. Разберёмся с тем, что такое сниппеты.
   Сниппет – это небольшой фрагмент исходного кода или текста, пригодный для многократного использования.

3. Напишем пять своих сниппетов для калькулятора: (все сниппеты в примерах будут написаны на языке программирования python, но надо иметь ввиду что сниппеты могут писаться сразу для нескольких языков программирования)

3.1) первый сниппет это сниппет функции

3.2) Второй сниппет это сниппет цикла for

3.3) Третий это обмен значениями переменных

3.4) Четвертый это конструкция if else

3.5) Пятый это конструкция if elif else

Пункт №6.

(HotKeys в Visual Studio Code)

Для ускоренной работы нам также могут пригодиться горячие клавиши.

Вот список наиболее распространенных горячих клавиш для работы в Visual Studio Code:

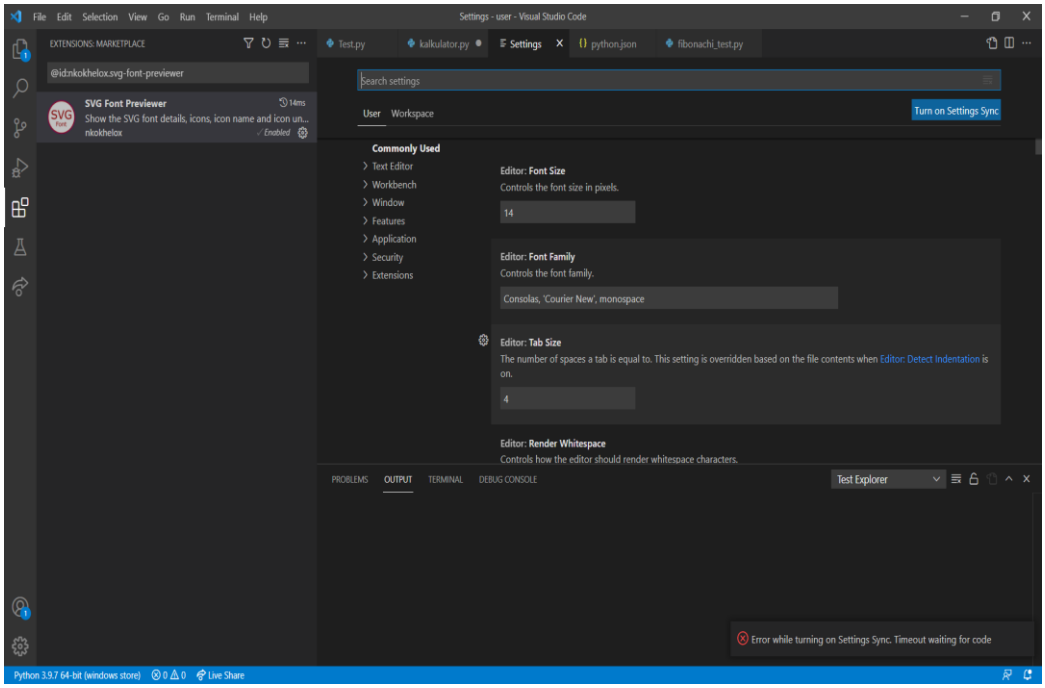
1) Ctrl + N --- Новый файл
2) Ctrl + Shift + N --- Новое окно редактора
3) Ctrl + W, Ctrl + F4 --- Закрыть окно редактора
4) Ctrl + O --- Открыть Файл
5) Ctrl + W --- Закрыть файл
6) Ctrl + S --- Сохранить
7) Ctrl + Shift + S --- Сохранить как
8) Ctrl + F --- Найти
9) Alt + F --- Форматировать документ
10) Ctrl + Shift + D --- Отладка

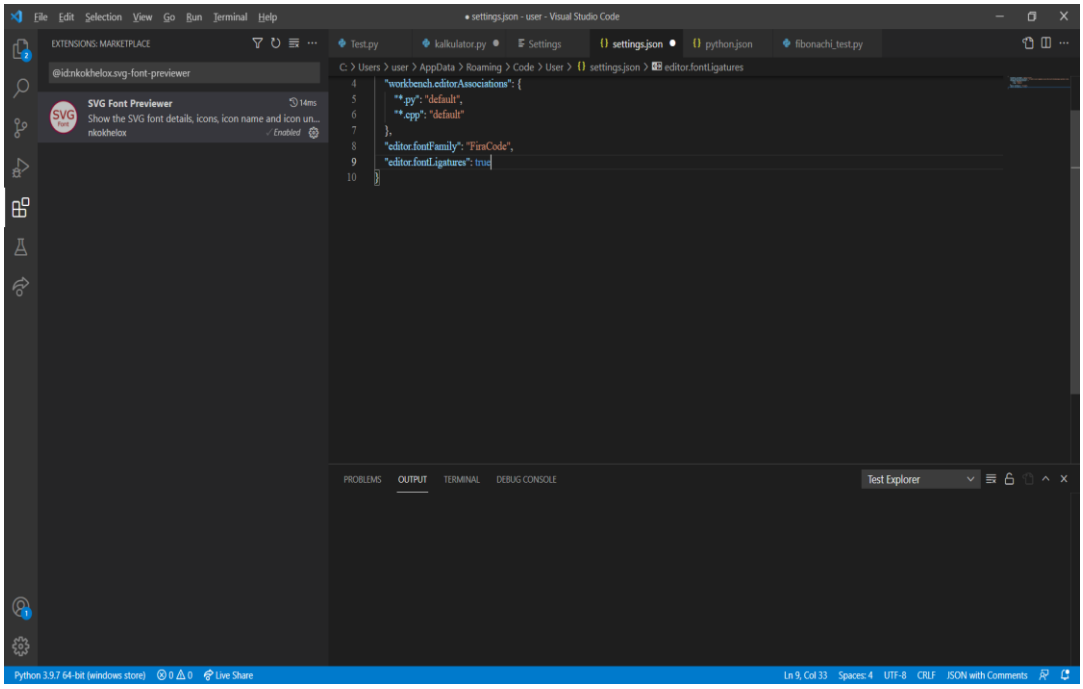
Пункт №7.

Документируем код.

Пункт №8.

Устанавливаем шрифт FiraCode.





Пункт №9.

Устанавливаем Live Share

Ниже представлены: код калькулятора, тест к калькулятору, а также сниппеты.

1. код калькулятора:

```
def funk(s):



    plus = s.count("+")
    minus = s.count('-')

    kl = len(s)

    num = 0

    a = []
    b = []
    c = []

    kc = 0

    for i in range(kl):
        if s[num] == "+":
            a.append(num)
            c.append(num)
            kc += 1
        elif s[num] == "-":
            b.append(num)
            c.append(num)
            kc += 1
        num += 1
```

```
kc1 = kc


f = 0
h = 0


sim = []


while kc > 0:
    kc -= 1
    if f == 0:
        ch = int(s[ : c[h] : ])
        sim.append(ch)
        f += 1
    if h != 1:
        ch = int(s[c[h] + 1 : c[h+1] : ])
        sim.append(ch)
        h += 1
    elif f > 0 and kc > 0:
        ch = int(s[c[f] + 1 : c[f+1] : ])
        sim.append(ch)
        f += 1
    elif kc == 0:
        ch = int(s[c[f] + 1 : : ])
        sim.append(ch)




sum = 0
p = 0
g = 0
```

```
si = 0
ij = 0
flag = 0

pk = len(a)
gk = len(b)

while kc1 > 0:
    if flag == 0:
        sum += sim[si]
        si += 1
        flag += 1
    if pk > 0:
        if a[p] == c[ij]:
            sum += sim[si]
            p += 1
            ij += 1
            kc1 -= 1
            si +=1
            pk -= 1
    if gk > 0:
        if b[g] == c[ij]:
            sum -= sim[si]
            g += 1
            ij += 1
            kc1 -= 1
            si += 1
            gk -= 1

return sum
```

''' функция вычисляющая значение вырвжения'''

def fibonachi(n):

    if n == 1:

        return 0

    elif n == 2:

        return 1

    else:

        return fibonachi(n-2)+fibonachi(n-1)

''' функция вычисляющая значение числа Фибоначчи'''

print('Hello world!')

print('Если вы хотите узнать информацию о том как работает калькулятор введите "yes" иначе введите "no"')

s=input()

n1=s.find("y")

n2=s.find("e")

n3=s.find("s")

n4=s.find("n")

n5=s.find("o")

n6=len(s)

if n1==0 and n2==1 and n3==2 and n6==3:

    print("Если Вы хотите провести операцию налд двумя числами, то введите данную операцию и нажмите =.")

    print("Выполнять несколько операций за один ход нельзя.")

elif n4==0 and n5==1 and n6==2:

    print('Hello')

else:

    print('Неверная команда')

```
x = 1



while x > 0:
    print("Введите операцию:")
    expres=input()
    if expres.count("+")==1 and expres.count("-")==0 and expres.count("*")==0 and expres.count("/")==0:
        d=expres
        plu = expres.find("+")
        firstSlag=int(expres[: plu:])
        secondSlag=int(expres[plu: : ])
        print("Результат:")
        print(firstSlag + secondSlag)


    elif expres.count("-")==1 and expres.count("+")==0 and expres.count("*")==0 and expres.count("/")==0 and expres.count("^")==0:
        d1 = expres
        minu = expres.find("-")
        reduc =    int(expres[: minu :])
        deduct = int(expres[minu +1 : : ])
        print("Результат:")
        print(reduc - deduct)


    elif expres.count("*")==1 and expres.count("+")==0 and expres.count("-")==0 and expres.count("/")==0:
        Mn = expres.find("*")
        firstMn = int(expres[: Mn :])
        secondMn = int(expres[Mn + 1 : : ])
        print("Результат:")
        print(firstMn * secondMn)
```

```
    elif expres.count("/")==1 and expres.count("*")==0 and expres.count("+")==0
and expres.count("-")==0:
        delt = expres.find("/")
        divisFirst = int(expres[: delt :])
        divisSecond = int(expres[delt + 1 : : ])
        if divisSecond !=0:
            print("Результат:")
            print(divisFirst / divisSecond)
        else:
            print("Неверная операция. На 0 делить нельзя!")


    if expres.count("^")==1 and expres.count("/")==0 and expres.count("*")==0 and
expres.count("+")==0 and expres.count("-")==0:
        step = expres.find("^")
        ch1 = int(expres[: step : ])
        ch2 = int(expres[step + 1 : : ])
        ch3 = 1
        if ch2 > 1:
            q = ch2
            while q != 0:
                ch3 = ch3 * ch1
                q = q - 1
        elif ch2 < 0:
            q1 = ch2
            while q != 0:
                ch3 = ch3 * ch1
                q = q + 1
            ch3 = 1 / ch3
        print("Результат:")
        print( ch3 )
```

```
    if expres.count("b") == 1 and expres.count("i") == 1 and expres.count("g") == 1
and expres.count("v") == 1:
        print("введите сложную операцию:")
        bigv = input()
        result = funk(bigv)
        print("Результат:")
        print(result)

    if expres.count("f") == 1 and expres.count("i") == 1 and expres.count("b") == 1:
        print("Введите номер числа Фибоначчи:")
        fib = int(input())
        resultfib = fibonachi(fib)
        print(fib, "число Фибоначчи =", resultfib)
''' основная программа калькулятора'''
```

2. тест к калькулятору(к функции, считающей числа Фиббоначи):

```
import unittest

from kalkulator import fibonachi

class TestFibonachi(unittest.TestCase):
    def test_zero(self):
        self.assertEqual(fibonachi(1), 0)
```

3. Сниппеты:

```
{
  // Place your snippets for python here. Each snippet is defined under a snippet
name and has a prefix, body and
```

```
  // description. The prefix is what is used to trigger the snippet and the body will be expanded and inserted. Possible variables are:
  // $1, $2 for tab stops, $0 for the final cursor position, and ${1:label}, ${2:another} for placeholders. Placeholders with the
  // same ids are connected.
  // Example:
  // "Print to console": {
  //         "prefix": "log",
  //         "body": [
  //                 "console.log('$1');",
  //                 "$2"
  //         ],
  //         "description": "Log output to console"
  // }

  "print to console":{
          "prefix": "f",
          "body": [
                  "def f(n):",
                  "",
                  "return "
          ],
  },

  "цикл for":{
          "prefix": "for",
          "body": [
                  "for i in range(n):",
                  ""
          ],
  },
```

```
"обмен двух значений":{
        "prefix": "sw",
        "body": [
                "def swap(a, b):",
                "return b, a"
        ],
},

"if else":{
        "prefix": "ifs",
        "body": [
                "if :",
                "",
                "else: ",
                ""
        ],
},

"if elif else":{
        "prefix": "ife",
        "body": [
                "if :",
                "",
                "elif :",
                "",
                "else:",
                ""
        ],
}
```

```
}
```