

Moduły i ich importowanie

Moduły są odpowiednikami zewnętrznych funkcji/bibliotek dołączanych do projektu. Moduły można utworzyć samemu lub użyć modułów wbudowanych. Jeżeli będziemy chcieli zaimportować nie istniejący moduł to dostaniemy komunikat o braku tego modułu:

```
Moduly.py*  ▸ ×
1  #Moduły
2  import random
3  import kukuryku
4
5
```

unresolved import 'kukuryku'

Przykład importu biblioteki oraz funkcji z biblioteki lub zmiennej z biblioteki:

```
1  #Moduły
2  import random
3  print(random.randint(1,5))
4  # można odwołać się do funkcji w danym module
5  # przy użyciu słowa 'from'
6  from random import randint
7  print(randint(1,10))
8  from math import pi
9  print(pi)
```

C:\Program Files (x86)\Microsoft Visual S...
4
1
3.141592653589793
Press any key to continue . . .

Można również wprowadzić własną nazwę:

```
1  #Moduły
2  import random
3  print(random.randint(1,5))
4  # można odwołać się do funkcji w danym module
5  # przy użyciu słowa 'from' :
6  from random import randint
7  print(randint(1,10))
8  from math import sqrt as pierwiastek
9  print(pi)
10 #zastosowanie własnej nazwy dla funkcji
11 from math import sqrt as pierwiastek
12 print(pierwiastek(4))
```

C:\Program Files (x86)\Microsoft Visual Stud...
4
8
3.141592653589793
2.0
Press any key to continue . . .

Inne funkcje modułu math:

Math Methods

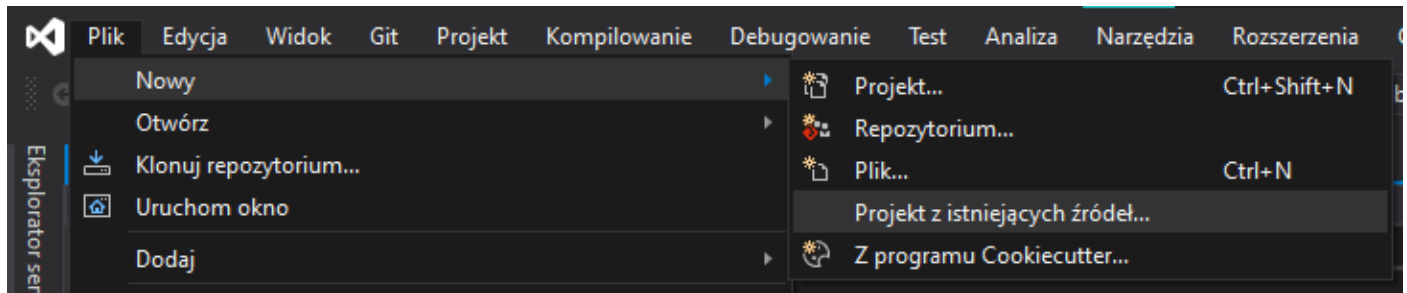
Method	Description
<u>math.acos()</u>	Returns the arc cosine of a number
<u>math.acosh()</u>	Returns the inverse hyperbolic cosine of a number
<u>math.asin()</u>	Returns the arc sine of a number
<u>math.asinh()</u>	Returns the inverse hyperbolic sine of a number
<u>math.atan()</u>	Returns the arc tangent of a number in radians
<u>math.atan2()</u>	Returns the arc tangent of y/x in radians
<u>math.atanh()</u>	Returns the inverse hyperbolic tangent of a number
<u>math.ceil()</u>	Rounds a number up to the nearest integer
<u>math.comb()</u>	Returns the number of ways to choose k items from n items without repetition and order
<u>math.copysign()</u>	Returns a float consisting of the value of the first parameter and the sign of the second parameter

<u>math.copysign()</u>	Returns a float consisting of the value of the first parameter and the sign of the second parameter
<u>math.cos()</u>	Returns the cosine of a number
<u>math.cosh()</u>	Returns the hyperbolic cosine of a number
<u>math.degrees()</u>	Converts an angle from radians to degrees
<u>math.dist()</u>	Returns the Euclidean distance between two points (p and q), where p and q are the coordinates of that point
<u>math.erf()</u>	Returns the error function of a number
<u>math.erfc()</u>	Returns the complementary error function of a number
<u>math.exp()</u>	Returns E raised to the power of x
<u>math.expm1()</u>	Returns $E^x - 1$
<u>math.fabs()</u>	Returns the absolute value of a number
<u>math.factorial()</u>	Returns the factorial of a number
<u>math.floor()</u>	Rounds a number down to the nearest integer
<u>math.fmod()</u>	Returns the remainder of x/y
<u>math.frexp()</u>	Returns the mantissa and the exponent, of a specified number
<u>math.fsum()</u>	Returns the sum of all items in any iterable (tuples, arrays, lists, etc.)

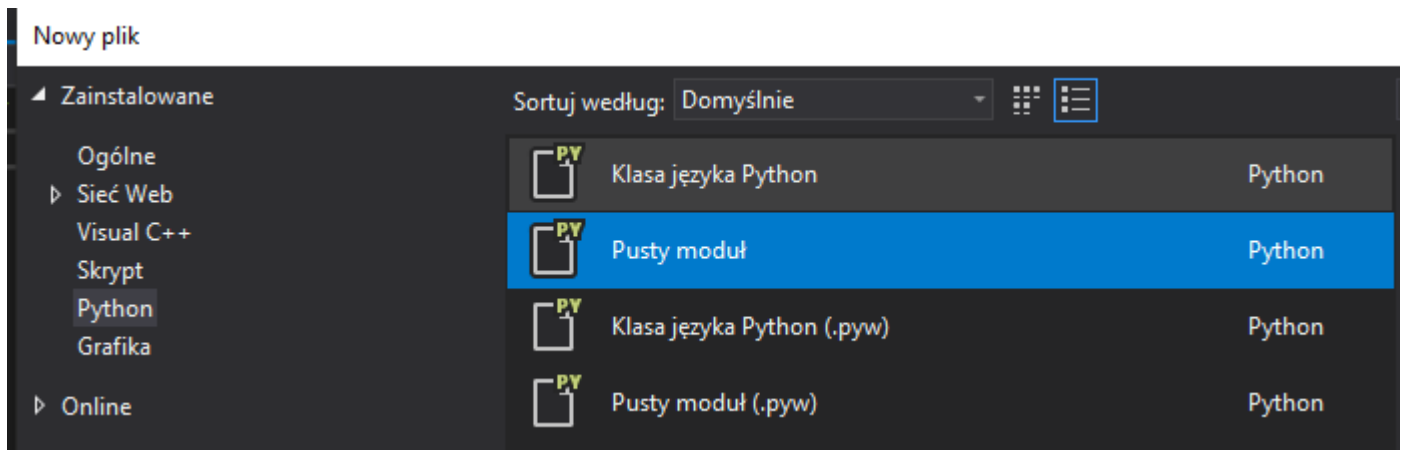
pozostałe: https://www.w3schools.com/python/module_math.asp

Stworzenie własnego modułu

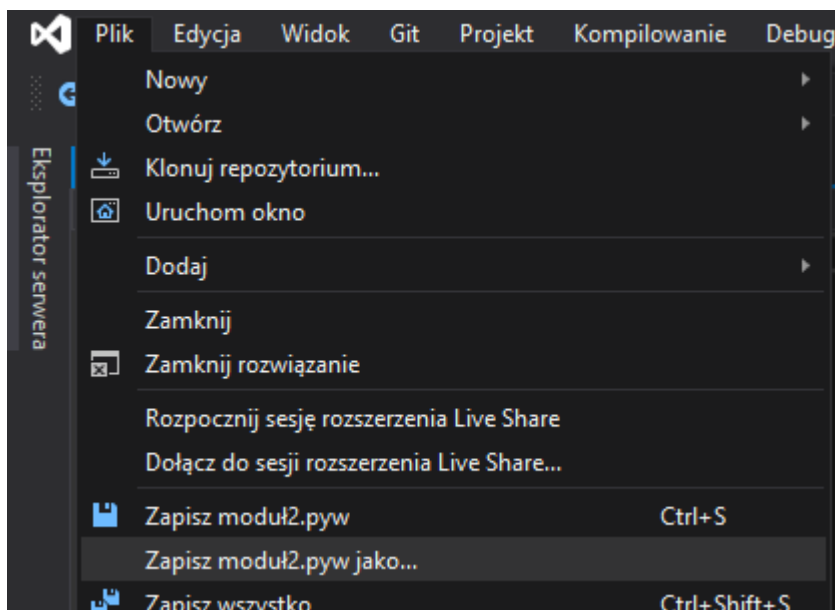
Do projektu dodajemy dodatkowy plik:



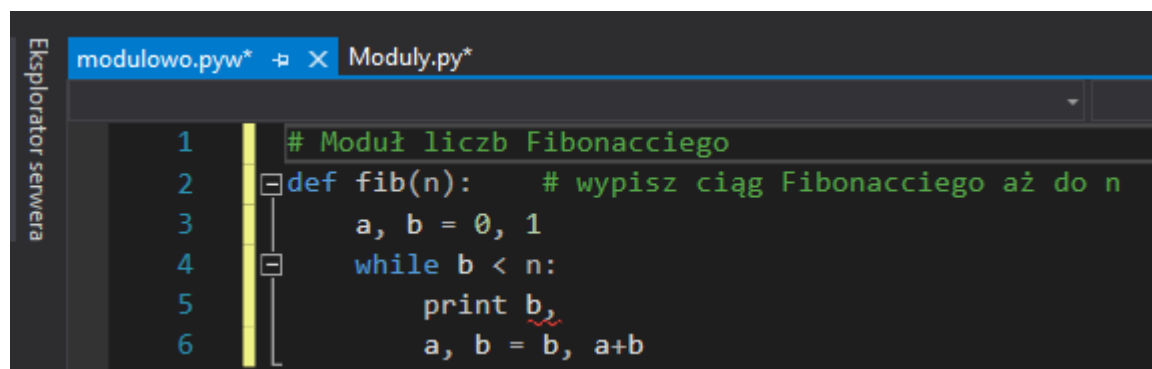
Wybieramy typ pliku



Zmieniamy nazwę pliku

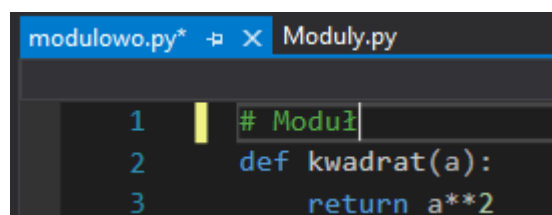


Aby sprawdzić możliwość importu modułu tworzymy w nim funkcję:



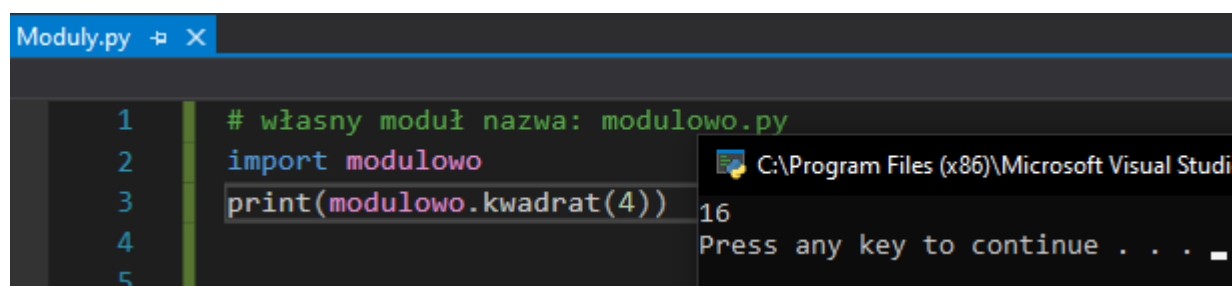
```
1 # Moduł liczb Fibonacciego
2 def fib(n): # wypisz ciąg Fibonacciego aż do n
3     a, b = 0, 1
4     while b < n:
5         print b,
6         a, b = b, a+b
```

Oraz importujemy moduł w pliku głównym:



```
1 # Moduł
2 def kwadrat(a):
3     return a**2
```

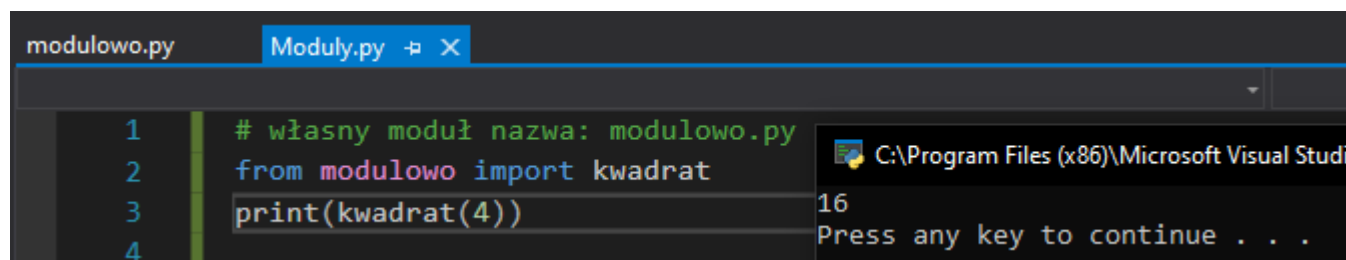
I wywołujemy funkcję:



```
1 # własny moduł nazwa: modulowo.py
2 import modulowo
3 print(modulowo.kwadrat(4))
4
5
```

16
Press any key to continue . . .

Lub:



```
1 # własny moduł nazwa: modulowo.py
2 from modulowo import kwadrat
3 print(kwadrat(4))
4
```

16
Press any key to continue . . .