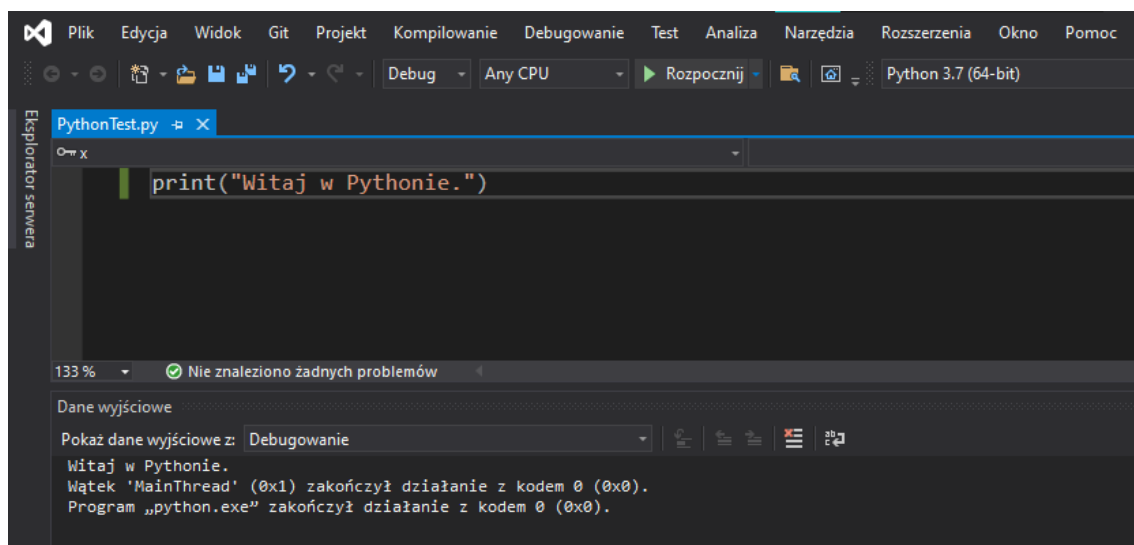
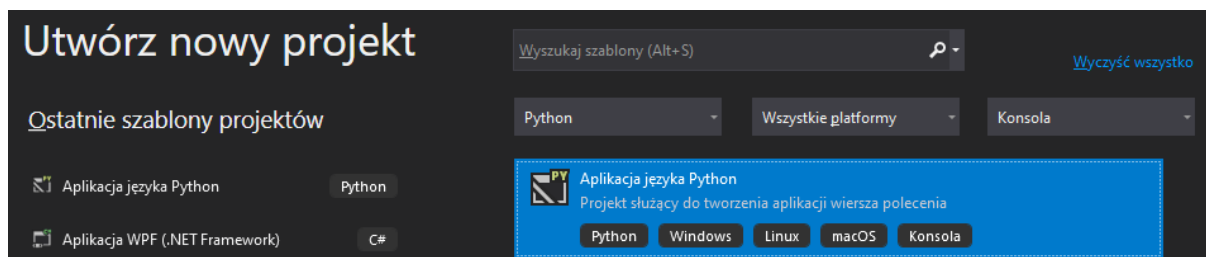


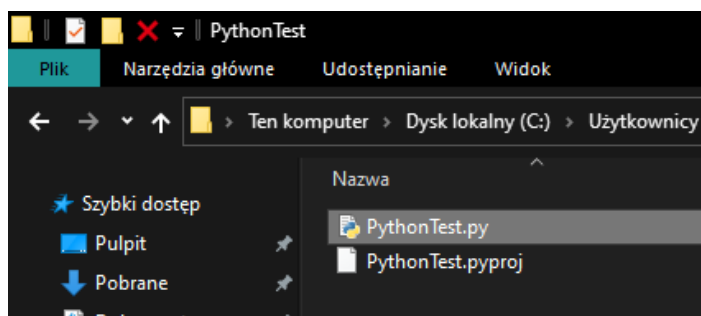
PYTHON

1. Instalacja środowiska (PyCharm, Python, Visual Studio, ...)
2. Test działania środowiska

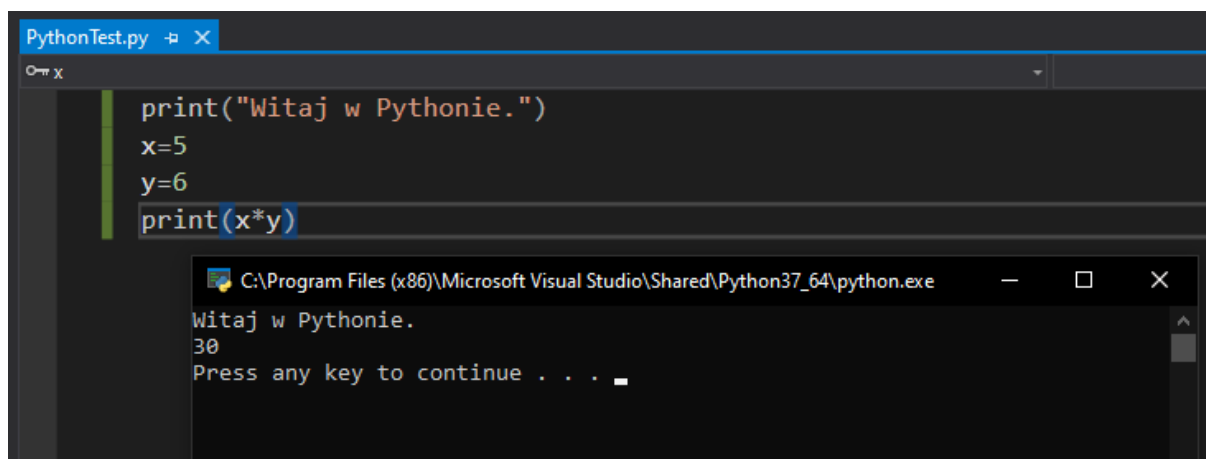
Na przykładzie Visual Studio Community:



Utworzony plik:



3. Tworzenie danych i wyświetlanie wyników operacji na danych



4. Komentowanie kodu

```
PythonTest.py*  ▢ ✕  
  
print("Witaj w Pythonie.")  
# --dane wejściowe po komentarzu liniowym  
x=5  
y=6  
'''  
Wyświetlenie wyniku po komentarzu blokowym  
'''  
print(x*y)
```

Komentarz blokowy zaczyna się

i kończy trzema apostrofami

5. Wprowadzanie różnych typów danych

```
TypyDanych.py*  ▢ ✕  
  
# typ integer /całkowity  
x = 200  
# typ float  
y = 3.5  
# dodawanie  
print (x + y)  
# odejmowanie  
print (x - y)  
# mnożenie  
print (x * y)  
# dzielenie  
print (x / y)  
# dzielenie całkowitoliczbowe  
print (x // y)  
# reszta z dzielenia  
print (x % y)  
# potęgowanie  
print (x ** y)
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Share  
203.5  
196.5  
700.0  
57.142857142857146  
57.0  
0.5  
113137084.9898476  
Press any key to continue . . .
```

Typ liczb zespolonych:

```
# liczba zespolona  
print(2.0+2.0j)
```

110 % Nie znaleziono żadnych problemów

Dane wyjściowe

Pokaż dane wyjściowe z: Debugowanie

```
0.5  
113137084.9898476  
(2+2j)  
Wątek 'MainThread' (0x1) zakończył działanie z kodem 0 (0x0).  
Program „python.exe” zakończył działanie z kodem 0 (0x0).
```

```
# liczba zespolona
print(2.0+2.0j)
# liczba sprzężona do liczby zespolonej
z = 2.0+2.0j
print(z)
print(z.conjugate())
```

```
(2+2j)
(2+2j)
(2-2j)
```

6. Typy sekwencyjne

```
TypySekwencyjne.py  X
# łańcuch typu string
str = 'żółw to gad'
print(str)
# łańcuch bajtów - tylko znaki ASCII
b = b'zolw to gad'
print(b)
#lista/tablica
l=[1,2,3]
print(l[0])
#krotka / rekord bazy danych
k=(1,2,3)
print(k)
```

Pokaż dane wyjściowe z: Debugowanie

```
żółw to gad
b'zolw to gad'
1
(1, 2, 3)
Wątek 'MainThread' (0x1) zakończył działanie z kodem 0 (0x0).
Program „python.exe” zakończył działanie z kodem 0 (0x0).
```

c.d.

```
TypySekwencyjne.py*  X
#konkatenacja '+'
print('ab'+ 'cd')
#operator należności 'in'
print('a' in ('a','b','c','d'))
#operator należności 'not in'
print('a' not in ('a','b','c','d'))
#dlugosc sekwencji
print(len('Kto to wie'))
# min i max wartość oraz suma
z = [3,9,1,0,-5,3,8]
print(max(z))
print(min(z))
print(sum(z))
#znajdowanie pierwszego znaku np. 'm'
print('Ala ma małego kota'.index('m'))
#zliczanie elementów danego typu np. 'm'
print('Ala ma małego kota'.count('m'))
```

C:\Program Files (x86)\Microsoft Visual St

```
abcd
True
False
10
9
-5
19
4
2
Press any key to continue . . .
```

7. Operacje na ciągach:

```
OperacjeNaCiągach.py  X
```

```
#nowa linia i tabulator
print("Tekst\nw nowej linii\tz tabulatorem")

# split - podział
tekst = 'Ala ma kota, a kot ma Ale'
print(tekst.split('a'))
print(tekst.split('a',2)) #do drugiego elementu
#rsplit - podział od końca
print(tekst.rsplit('a'))
print(tekst.rsplit('a',2)) #od końca

# join - łączenie
print(' '.join(['Ala','ma', 'kota','a',' ','kot','ma','Ale']))

# startswith lub endswith - zaczyna się lub kończy stringiem
print(tekst.startswith('Ala')) #zwraca true lub false
print(tekst.endswith('Ale'))  #zwraca true lub false

# czy litera, czy cyfra, czy litery i cyfry (T lub F)
tekst = 'Ala ma 2 koty'
print(tekst.isalpha()) # czy litery
print(tekst.isdigit()) # czy liczby
print('Ala ma 2 koty'.isalnum()) # czy litery i liczby
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared\Python37
Tekst
w nowej linii z tabulatorem
['Al', ' m', ' kot', ', ', ' kot m', ' Ale']
['Al', ' m', ' kota, a kot ma Ale']
['Al', ' m', ' kot', ', ', ' kot m', ' Ale']
['Ala ma kota, ', ' kot m', ' Ale']
Ala ma kota a , kot ma Ale
True
False
False
False
False
Press any key to continue . . .
```

```
OperacjeNaCiągach.py  X
```

```
# usuwanie białych znaków
tekst = ' Po dwie spacje '
print(tekst.strip()) # zobu stron
print(tekst.lstrip()) # zobu lewej strony
print(tekst.rstrip()) # zobu prawej strony

# zamiana ciągu nznaków na inny
print(tekst.replace('dwie', 'trzy'))
# usunięcie spacji
print(tekst.replace(' ', ''))
```

```
C:\Program Files (x86)\Microsoft Visual Studio\Shared
Po dwie spacje
Po dwie spacje
Po dwie spacje
Po trzy spacje
Po dwie spacje
Press any key to continue . . .
```

8. Listy

```
Listy.py X
```

```
# lista - append
lista = [2,4,6,8]
lista += [10] # bez append
lista.append(12)
print(lista)

# usuwa element listy - remove
lista.remove(2) # nie zadziała z liczbami
print(lista) # zadziała z liczbami
lista = ['as','lis','osa','kos','lis']
lista.remove('lis') # zadziała z tekstem
print(lista) # zadziała z tekstem

#usuwa element o danym indeksie
lista.pop(2)
print(lista)

# wstawia element na dane miejsce
lista.insert(2,'byk')
print(lista)

# metoda ZIP tworzy krotki
print(list(zip('abcdef',[1,2,3])))
```

```
C:\Program Files (x86)\Microsoft Visual Studio
[2, 4, 6, 8, 10, 12]
[4, 6, 8, 10, 12]
['as', 'osa', 'kos', 'lis']
['as', 'osa', 'lis']
['as', 'osa', 'byk', 'lis']
[('a', 1), ('b', 2), ('c', 3)]
Press any key to continue . . .
```