

**LAPORAN PRAKTIKUM**  
**ALGORITMA DAN STRUKTUR DATA**  
**MODUL 1**



**DISUSUN OLEH :**

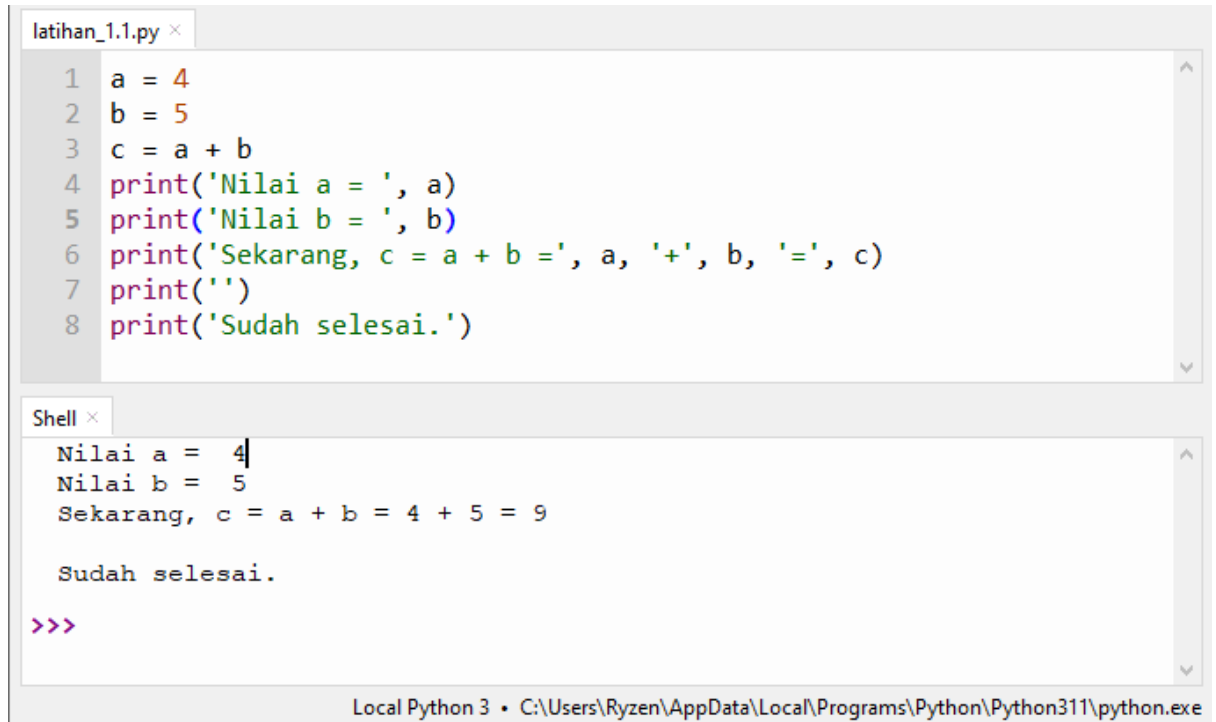
**NIM** : L200210021  
**NAMA** : Muhammad Irfan Abidin  
**KELAS** : B

**PROGRAM STUDI INFORMATIKA**  
**FAKULTAS KOMUNIKASI DAN INFORMATIKA**  
**UNIVERSITAS MUHAMMADIYAH SURAKARTA**

# TINJAUAN ULANG PYTHON

## Latihan

### 1.1



The screenshot shows a Python IDE with two tabs: 'latihan\_1.1.py' and 'Shell'. The 'latihan\_1.1.py' tab contains the following code:

```
1 a = 4
2 b = 5
3 c = a + b
4 print('Nilai a = ', a)
5 print('Nilai b = ', b)
6 print('Sekarang, c = a + b =', a, '+', b, '=', c)
7 print('')
8 print('Sudah selesai.')
```

The 'Shell' tab shows the output of the script:

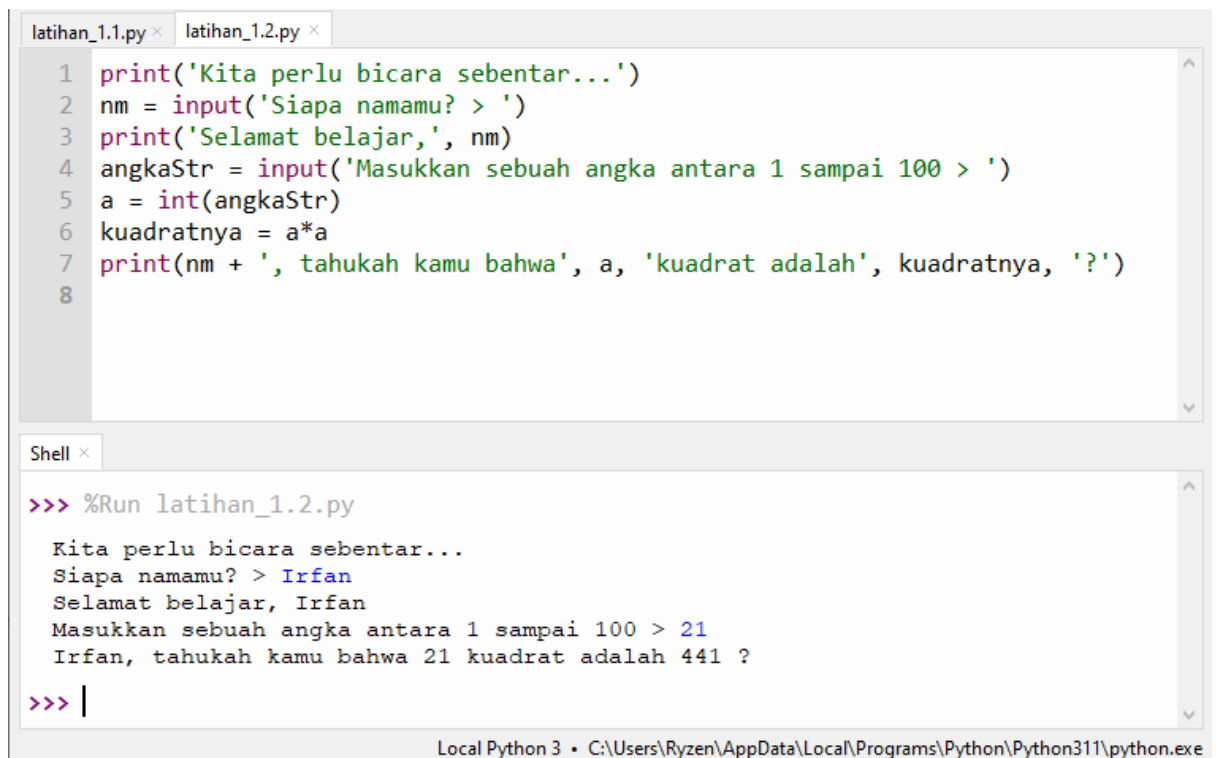
```
Nilai a = 4
Nilai b = 5
Sekarang, c = a + b = 4 + 5 = 9

Sudah selesai.

>>>
```

At the bottom of the IDE, the path 'Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe' is displayed.

### 1.2



The screenshot shows a Python IDE with two tabs: 'latihan\_1.1.py' and 'latihan\_1.2.py'. The 'latihan\_1.2.py' tab contains the following code:

```
1 print('Kita perlu bicara sebentar...')
2 nm = input('Siapa namamu? > ')
3 print('Selamat belajar,', nm)
4 angkaStr = input('Masukkan sebuah angka antara 1 sampai 100 > ')
5 a = int(angkaStr)
6 kuadratnya = a*a
7 print(nm + ', tahukah kamu bahwa', a, 'kuadrat adalah', kuadratnya, '?')
8
```

The 'Shell' tab shows the execution of the script:

```
>>> %Run latihan_1.2.py

Kita perlu bicara sebentar...
Siapa namamu? > Irfan
Selamat belajar, Irfan
Masukkan sebuah angka antara 1 sampai 100 > 21
Irfan, tahukah kamu bahwa 21 kuadrat adalah 441 ?

>>> |
```

At the bottom of the IDE, the path 'Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe' is displayed.

### 1.3



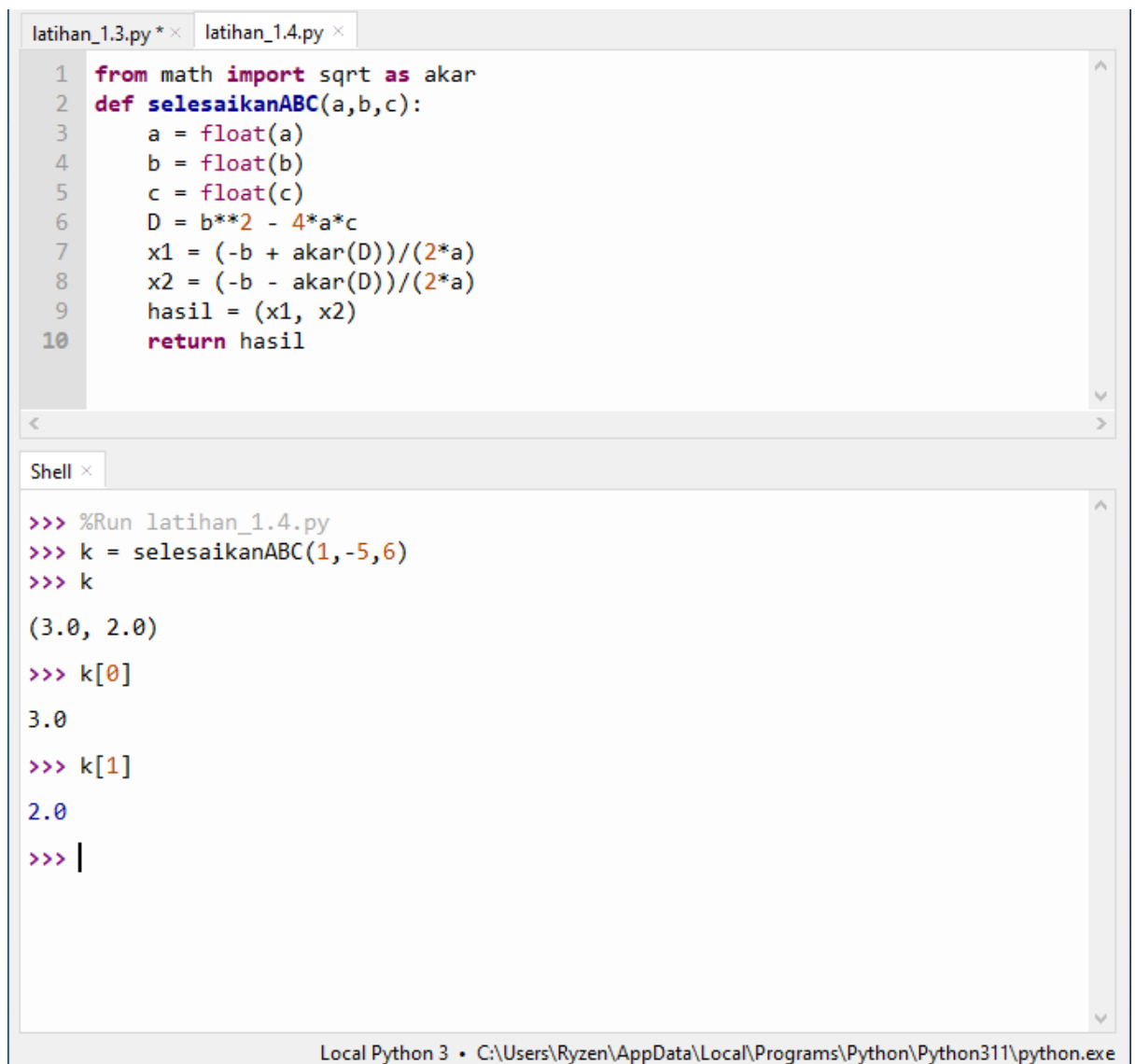
The image shows a Python IDE window with two panes. The top pane, titled 'latihan\_1.3.py \*', contains a Python script with three functions: `ucapkanSalam()`, `sapa(nama)`, and `kuadratkan(b)`. The bottom pane, titled 'Shell x', shows the execution of the script. The output of the script is as follows:

```
latihan_1.3.py * x
1 def ucapkanSalam():
2     print("Assalamu 'alikum!")
3
4 def sapa(nama):
5     ucapkanSalam()
6     print('Halo', nama)
7     print('Selamat belajar!')
8
9 def kuadratkan(b):
10     h = b*b
11     return h

Shell x
>>> %Run latihan_1.3.py
>>> ucapkanSalam()
    Assalamu 'alikum!
>>> sapa('Irfan')
    Assalamu 'alikum!
    Halo Irfan
    Selamat belajar!
>>> b = kuadratkan(5)
>>> b
25
>>> k = 9
>>> print('Bilangannya', k, ', kalau dipangkatkan dua jadinya', kuadratkan(k))
    Bilangannya 9 , kalau dipangkatkan dua jadinya 81
>>>
```

Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe

## 1.4



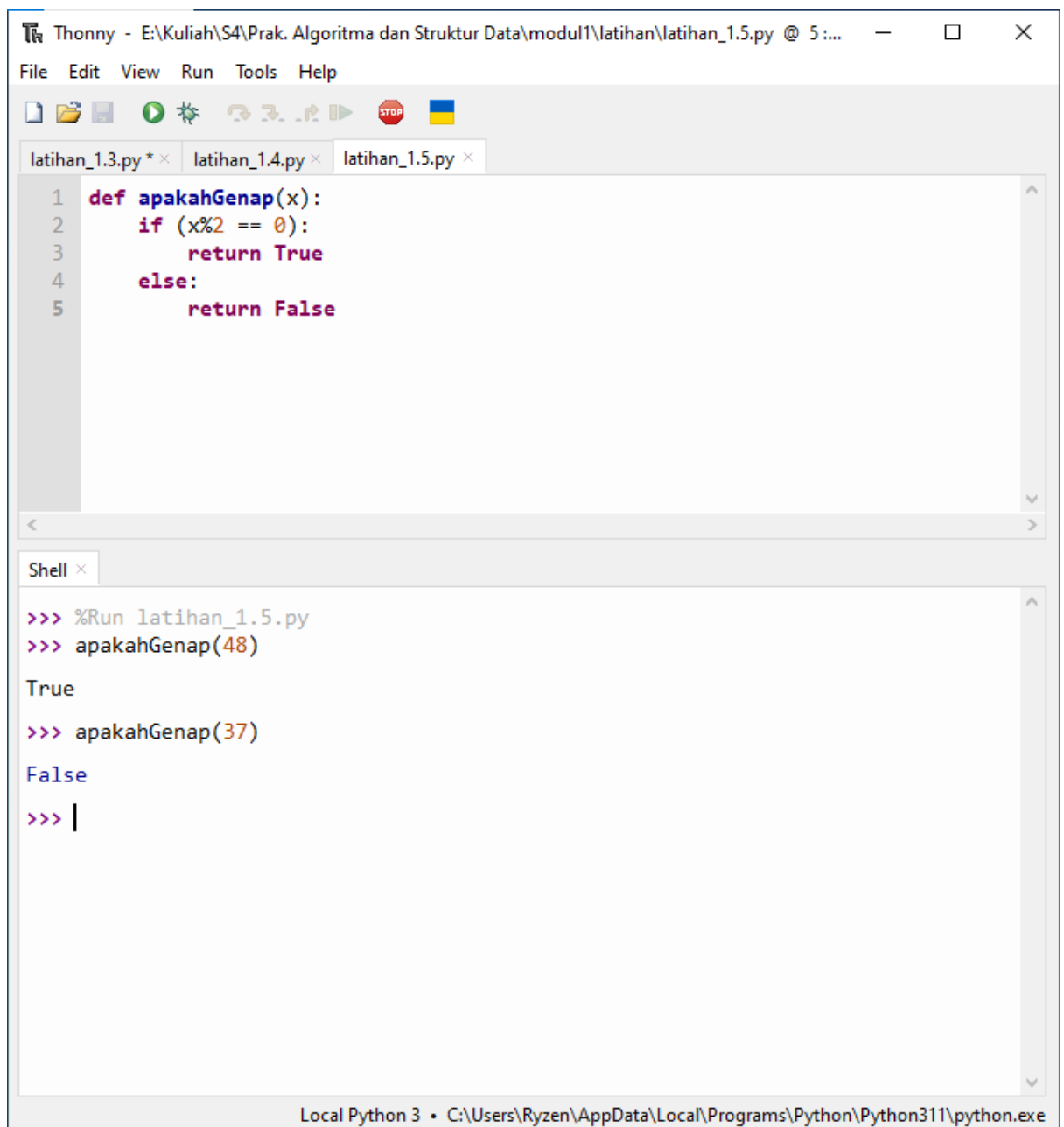
The image shows a Python IDE with two tabs: 'latihan\_1.3.py' and 'latihan\_1.4.py'. The 'latihan\_1.4.py' tab is active, displaying a function named 'selesaikanABC' that calculates the roots of a quadratic equation. Below the code editor is a 'Shell' window showing the execution of the function with parameters (1, -5, 6), resulting in the roots (3.0, 2.0). The status bar at the bottom indicates 'Local Python 3' is running from 'C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe'.

```
latihan_1.3.py × latihan_1.4.py ×
1  from math import sqrt as akar
2  def selesaikanABC(a,b,c):
3      a = float(a)
4      b = float(b)
5      c = float(c)
6      D = b**2 - 4*a*c
7      x1 = (-b + akar(D))/(2*a)
8      x2 = (-b - akar(D))/(2*a)
9      hasil = (x1, x2)
10     return hasil

Shell ×
>>> %Run latihan_1.4.py
>>> k = selesaikanABC(1,-5,6)
>>> k
(3.0, 2.0)
>>> k[0]
3.0
>>> k[1]
2.0
>>> |

Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe
```

## 1.5



The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named `latihan_1.5.py` with the following code:

```
1 def apakahGenap(x):  
2     if (x%2 == 0):  
3         return True  
4     else:  
5         return False
```

Below the editor is a Shell window showing the execution of the script:

```
>>> %Run latihan_1.5.py  
>>> apakahGenap(48)  
True  
>>> apakahGenap(37)  
False  
>>> |
```

The status bar at the bottom indicates the Python version and the interpreter path: Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe

## 1.6

```
latihan_1.6.py x
1 def tigaAtaulima(x):
2     if (x%3==0 and x%5==0):
3         print('Bilangan itu adalah kelipatan 3 dan 5 sekaligus')
4     elif (x%3==0):
5         print('Bilangan itu adalah kelipatan 3')
6     elif (x%5==0):
7         print('Bilangan itu adalah kelipatan 5')
8     else:
9         print('Bilangan itu bukan kelipatan 3 maupun 5')
10

Shell x
>>> %Run latihan_1.6.py
>>> tigaAtaulima(9)
    Bilangan itu adalah kelipatan 3
>>> tigaAtaulima(10)
    Bilangan itu adalah kelipatan 5
>>> tigaAtaulima(15)
    Bilangan itu adalah kelipatan 3 dan 5 sekaligus
>>> tigaAtaulima(17)
    Bilangan itu bukan kelipatan 3 maupun 5
>>> |
```

## 1.7

```
latihan_1.7.py x
1 staff = {'Santi' : 'santi@ums.ac.id', \
2         'Jokowi' : 'jokowi@solokab.go.id', \
3         'Endang' : 'Endang@yahoo.com', \
4         'Sulastri': 'Sulastri3@gmail.com'}
5
6 yangDicari = 'Santi'
7 if yangDicari in staff:
8     print('emailnya', yangDicari, 'adalah', staff[yangDicari])
9 else:
10    print('tidak ada yang namanya', yangDicari)

Shell x
>>> %Run latihan_1.7.py
    emailnya Santi adalah santi@ums.ac.id
>>> |
```

## 1.8



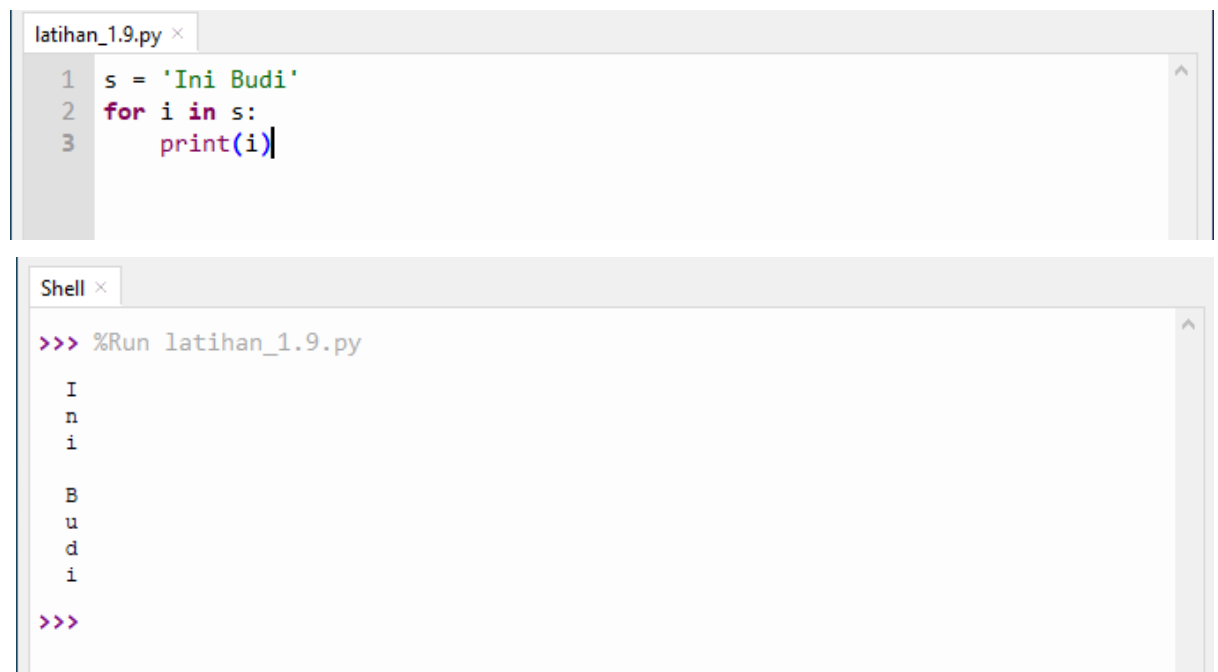
The screenshot shows a Python IDE with two panels. The top panel, titled 'latihan\_1.8.py', contains the following code:

```
1 for i in range(0,10):  
2     print(i)
```

The bottom panel, titled 'Shell', shows the execution of the script:

```
>>> %Run latihan_1.8.py  
  
0  
1  
2  
3  
4  
5  
6  
7  
8  
9  
  
>>>
```

## 1.9



The screenshot shows a Python IDE with two panels. The top panel, titled 'latihan\_1.9.py', contains the following code:

```
1 s = 'Ini Budi'  
2 for i in s:  
3     print(i)
```

The bottom panel, titled 'Shell', shows the execution of the script:

```
>>> %Run latihan_1.9.py  
  
I  
n  
i  
  
B  
u  
d  
i  
  
>>>
```

### 1.10

```
latihan_1.10.py ×
1 dd = {'nama':'joko', 'umur':21, 'asal': 'Surakarta'}
2 for kunci in dd:
3     print(kunci, '<---->', dd[kunci])

Shell ×
>>> %Run latihan_1.10.py

nama <----> joko
umur <----> 21
asal <----> Surakarta

>>>
```

### 1.11

```
latihan_1.11.py ×
1 bil = 0
2 while(bil*bil<200):
3     print(bil, bil*bil)
4     bil = bil + 1

Shell ×
>>> %Run latihan_1.11.py

0 0
1 1
2 4
3 9
4 16
5 25
6 36
7 49
8 64
9 81
10 100
11 121
12 144
13 169
14 196

>>>
```

Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe



## Soal - soal

1

```
tugas_1.1.py ×
1 def cetakSiku(x):
2     for i in range(x):
3         for j in range(i + 1):
4             print('*', end = '')
5             print()
6
7 cetakSiku(5)

Shell ×
*
**
***
****
*****

>>> |
```

2

```
tugas_1.2.py ×
1 def gambarlahPersegiEmpat(a, b):
2     for i in range(a):
3         if i == 0 or i == a - 1:
4             print('*' * b)
5         else:
6             print('*' + ' ' * (b-2) + '*')

< >

Shell ×
>>> %Run tugas_1.2.py
>>> gambarlahPersegiEmpat(4,5)

*****
*  *
*  *
*****

>>> |
```

```
tugas_1.3.py x
1 def jumlahHurufVokal(string):
2     jumlah_huruf = 0
3     jumlah_vokal = 0
4     vokal = 'aiueoAIUEO'
5
6     for char in string:
7         if char.isalpha():
8             jumlah_huruf += 1
9             if char in vokal:
10                 jumlah_vokal += 1
11
12     return [jumlah_huruf, jumlah_vokal]
13
14 def jumlahHurufKonsonan(string):
15     jumlah_huruf = 0
16     jumlah_konsonan = 0
17     vokal = 'aiueoAIUEO'
18
19     for char in string:
20         if char.isalpha():
21             jumlah_huruf += 1
22             if char not in vokal:
23                 jumlah_konsonan += 1
24     return [jumlah_huruf, jumlah_konsonan]
25 |

Shell x
>>> %Run tugas_1.3.py
>>> k = jumlahHurufVokal('Surakarta')
>>> k
[9, 4]
>>> k = jumlahHurufKonsonan('Surakarta')
>>> k
[9, 5]
>>>
```

4

```
tugas_1.4.py ×
1 def rerata(x):
2     hasil = sum(x) / len(x)
3     return hasil
4
5 def variance(x):
6     hasil = sum([(xi - rerata(x)) ** 2 for xi in x]) / len(x)
7     return hasil
8
9 def stdev(x):
10    import math
11    hasil = math.sqrt(variance(x))
12    return hasil

Shell ×
>>> %Run tugas_1.4.py
>>> x = [1, 2, 3, 4, 5]
>>> rerata(x)
3.0
>>> variance(x)
2.0
>>> stdev(x)
1.4142135623730951
>>> g = [3,4,5,4,3,4,5,2,2,10,11,23]
>>> rerata(g)
6.333333333333333
>>> variance(g)
32.72222222222222
>>> stdev(g)
5.720334100576838
>>>
```



The image shows a Python IDE with two tabs: 'tugas\_1.5.py' and 'tugas\_1.6.py'. The 'tugas\_1.5.py' tab is active, displaying a Python function named 'apakahPrima(n)'. The function checks if a number is prime by first testing against a list of small primes (2, 3, 5, 7, 11) and then testing divisibility by numbers from 2 up to the square root of n. Below the code editor is a 'Shell' window showing the execution of the script. The shell contains the command '%Run tugas\_1.5.py' followed by several calls to the 'apakahPrima' function with arguments 17, 97, 123, and 10. The outputs are 'True' for the first three calls and 'False' for the last one.

```
tugas_1.5.py x tugas_1.6.py x
1 from math import sqrt as sq
2 def apakahPrima(n):
3     n = int(n)
4     assert n >= 0
5     primaKecil = [2,3,5,7,11]
6     bukanPrKecil = [0,1,4,6,8,9,10]
7     if n in primaKecil:
8         return True
9     elif n in bukanPrKecil:
10        return False
11    else:
12        for i in range(2, int(sq(n))+1):
13            if n % i == 0:
14                return False
15        return True

Shell x
>>> %Run tugas_1.5.py
>>> apakahPrima(17)
True
>>> apakahPrima(97)
True
>>> apakahPrima(123)
True
>>> apakahPrima(10)
False
>>>
```

Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe

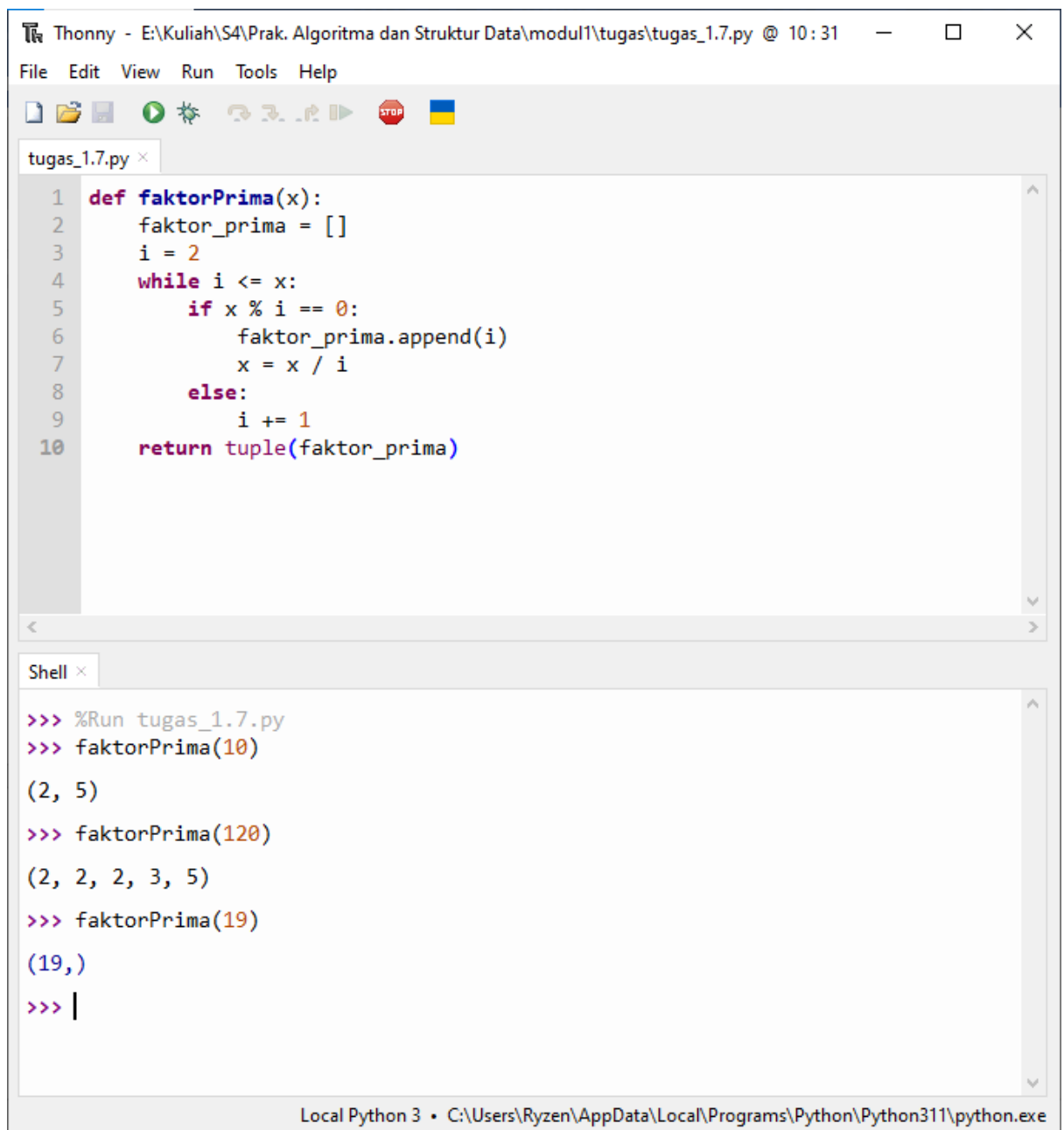
6

The image shows a Python IDE window with two panes. The top pane, titled 'tugas\_1.6.py', contains a Python script that iterates through numbers from 2 to 1000, checking for divisibility. The bottom pane, titled 'Shell', displays the output of the script, which is a list of prime numbers. The status bar at the bottom indicates the interpreter is 'Local Python 3' located at 'C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe'.

```
tugas_1.6.py ×
1 for num in range(2, 1001):
2     for i in range(2, num):
3         if (num % i) == 0:
4             break
5     else:
6         print(num)
```

```
Shell ×
823
827
829
839
853
857
859
863
877
881
883
887
907
911
919
929
937
941
947
953
967
971
977
983
991
997
>>>
```

Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe



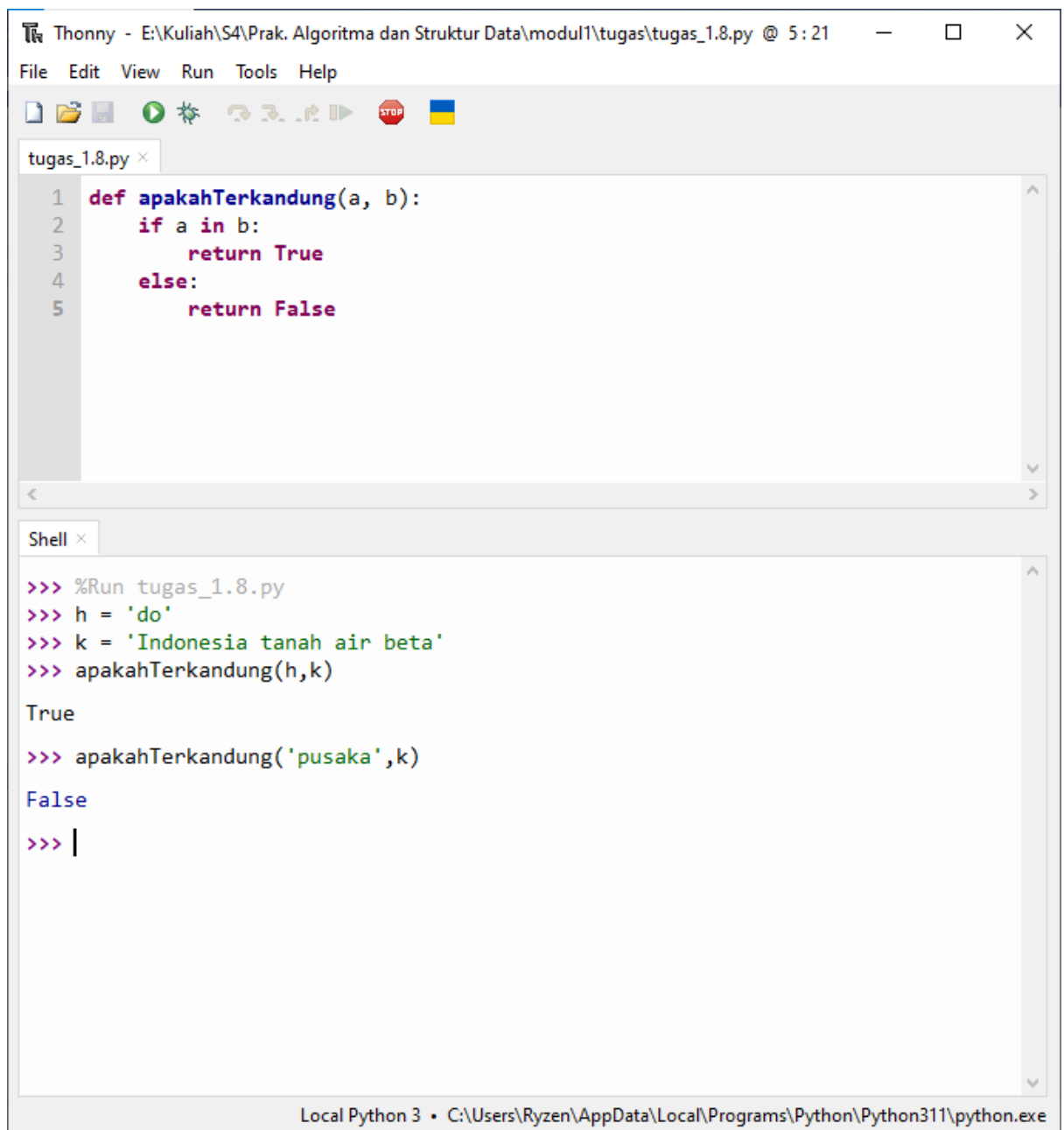
The screenshot shows the Thonny Python IDE interface. The top window, titled 'tugas\_1.7.py', contains a Python function `faktorPrima(x)` designed to find the prime factors of a given number `x`. The function initializes an empty list `faktor_prima` and starts with `i = 2`. It enters a `while` loop that continues as long as `i` is less than or equal to `x`. Inside the loop, it checks if `x` is divisible by `i` (`x % i == 0`). If true, it appends `i` to the list and divides `x` by `i`. If false, it increments `i` by 1. Finally, it returns the list of factors as a tuple.

The bottom window, titled 'Shell', shows the execution of the script. It starts with `%Run tugas_1.7.py`, followed by three function calls: `faktorPrima(10)` returning `(2, 5)`, `faktorPrima(120)` returning `(2, 2, 2, 3, 5)`, and `faktorPrima(19)` returning `(19,)`. The prompt `>>> |` indicates the shell is ready for further input.

```
1 def faktorPrima(x):
2     faktor_prima = []
3     i = 2
4     while i <= x:
5         if x % i == 0:
6             faktor_prima.append(i)
7             x = x / i
8         else:
9             i += 1
10    return tuple(faktor_prima)
```

```
>>> %Run tugas_1.7.py
>>> faktorPrima(10)
(2, 5)
>>> faktorPrima(120)
(2, 2, 2, 3, 5)
>>> faktorPrima(19)
(19,)
>>> |
```

Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe



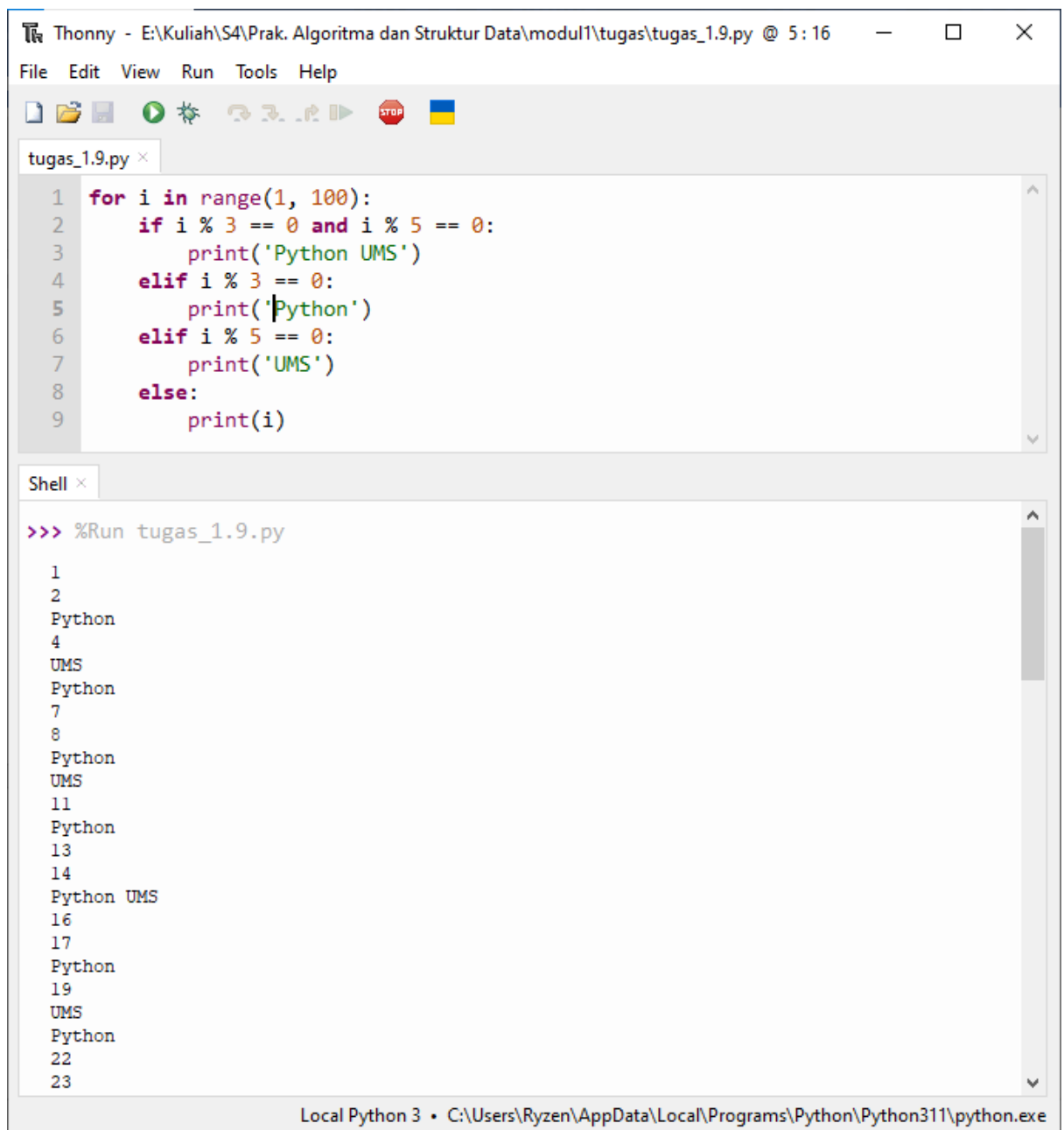
The screenshot shows the Thonny Python IDE interface. The title bar reads "Thonny - E:\Kuliah\S4\Prak. Algoritma dan Struktur Data\modul1\tugas\tugas\_1.8.py @ 5:21". The menu bar includes "File", "Edit", "View", "Run", "Tools", and "Help". The toolbar contains icons for opening files, saving, running, and other IDE functions. The editor window, titled "tugas\_1.8.py", contains the following Python code:

```
1 def apakahTerkandung(a, b):
2     if a in b:
3         return True
4     else:
5         return False
```

Below the editor is the "Shell" window, which shows the execution of the script:

```
>>> %Run tugas_1.8.py
>>> h = 'do'
>>> k = 'Indonesia tanah air beta'
>>> apakahTerkandung(h,k)
True
>>> apakahTerkandung('pusaka',k)
False
>>> |
```

The status bar at the bottom indicates "Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe".



The screenshot shows the Thonny Python IDE interface. The top window displays a Python script named `tugas_1.9.py` with the following code:

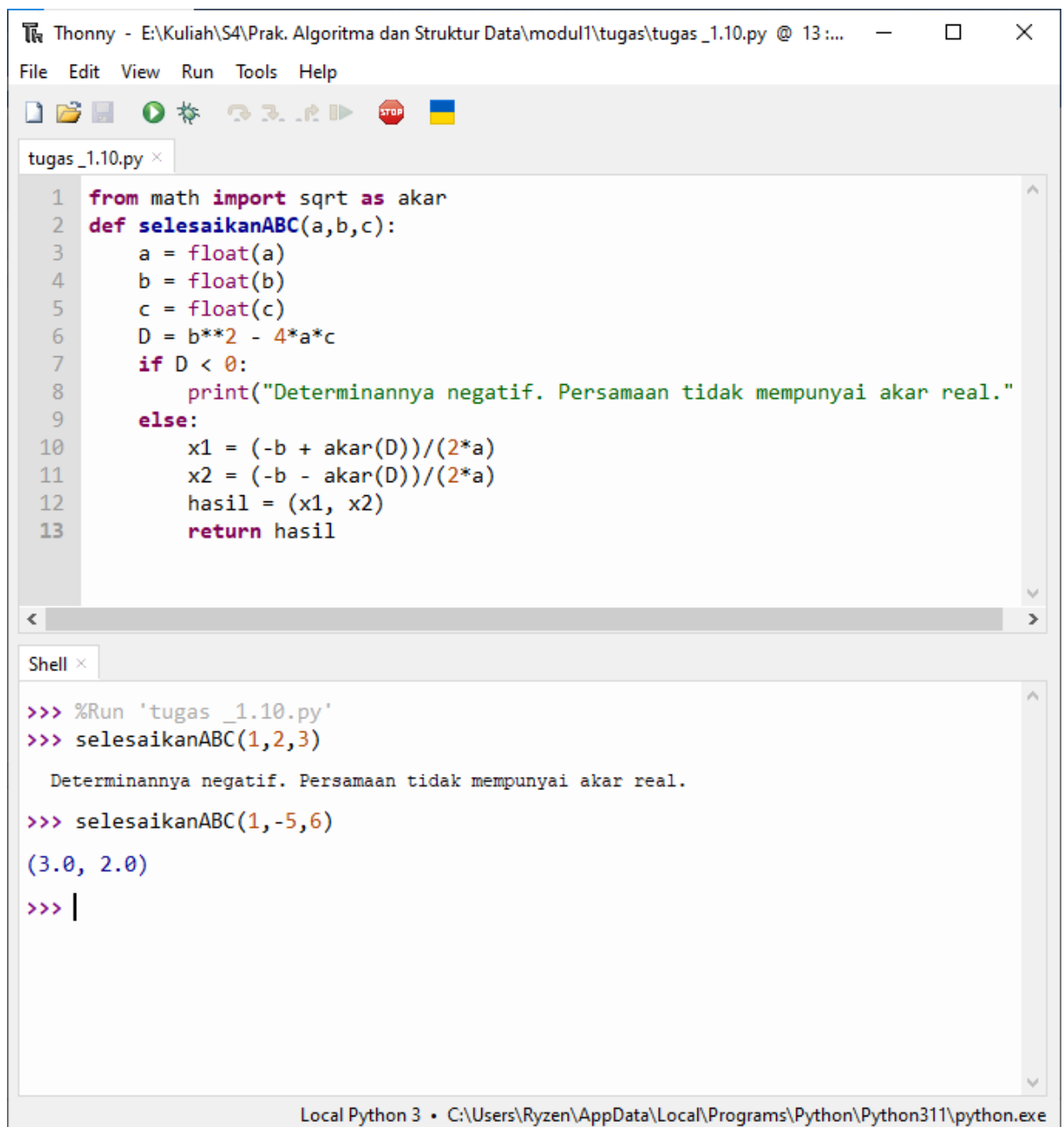
```
1 for i in range(1, 100):
2     if i % 3 == 0 and i % 5 == 0:
3         print('Python UMS')
4     elif i % 3 == 0:
5         print('Python')
6     elif i % 5 == 0:
7         print('UMS')
8     else:
9         print(i)
```

The bottom window, titled "Shell", shows the output of running the script. It displays the numbers 1 through 23, with the words "Python" and "UMS" printed at specific intervals corresponding to the logic in the script.

```
>>> %Run tugas_1.9.py
1
2
Python
4
UMS
Python
7
8
Python
UMS
11
Python
13
14
Python UMS
16
17
Python
19
UMS
Python
22
23
```

The status bar at the bottom indicates the Python version and the path to the Python executable: "Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe".





The screenshot shows the Thonny Python IDE interface. The main editor window displays a Python script named `tugas_1.10.py` that defines a function `selesaikanABC(a,b,c)` to solve quadratic equations. The function calculates the discriminant  $D = b^2 - 4ac$ . If  $D < 0$ , it prints a message indicating no real roots. Otherwise, it calculates the two roots  $x_1$  and  $x_2$  using the quadratic formula and returns them as a tuple.

```
1 from math import sqrt as akar
2 def selesaikanABC(a,b,c):
3     a = float(a)
4     b = float(b)
5     c = float(c)
6     D = b**2 - 4*a*c
7     if D < 0:
8         print("Determinannya negatif. Persamaan tidak mempunyai akar real.")
9     else:
10        x1 = (-b + akar(D))/(2*a)
11        x2 = (-b - akar(D))/(2*a)
12        hasil = (x1, x2)
13        return hasil
```

The Shell window below the editor shows the execution of the script. It runs the file `tugas_1.10.py` and then calls the function `selesaikanABC(1,2,3)`, which outputs the message "Determinannya negatif. Persamaan tidak mempunyai akar real.". Subsequently, it calls `selesaikanABC(1,-5,6)`, which outputs the tuple `(3.0, 2.0)`. The prompt `>>> |` indicates the shell is ready for further input.

```
>>> %Run 'tugas_1.10.py'
>>> selesaikanABC(1,2,3)
    Determinannya negatif. Persamaan tidak mempunyai akar real.
>>> selesaikanABC(1,-5,6)
(3.0, 2.0)
>>> |
```

At the bottom of the window, the status bar indicates the Python version and the interpreter path: "Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe".

```
tugas_1.11.py x
1 def apakahKabisat(tahun):
2     if tahun % 4 == 0:
3         if tahun % 100 == 0:
4             if tahun % 400 == 0:
5                 print(tahun, 'tahun kabisat (habis dibagi 400)')
6                 return True
7             else:
8                 print(tahun, 'bukan tahun kabisat (meski habis dibagi 4 dan 100 tapi tidak habis dibagi 400)')
9                 return False
10        else:
11            print(tahun, 'tahun kabisat (habis dibagi 4)')
12            return True
13    else:
14        print(tahun, 'bukan tahun kabisat(sudah jelas)')
15        return False
16    |

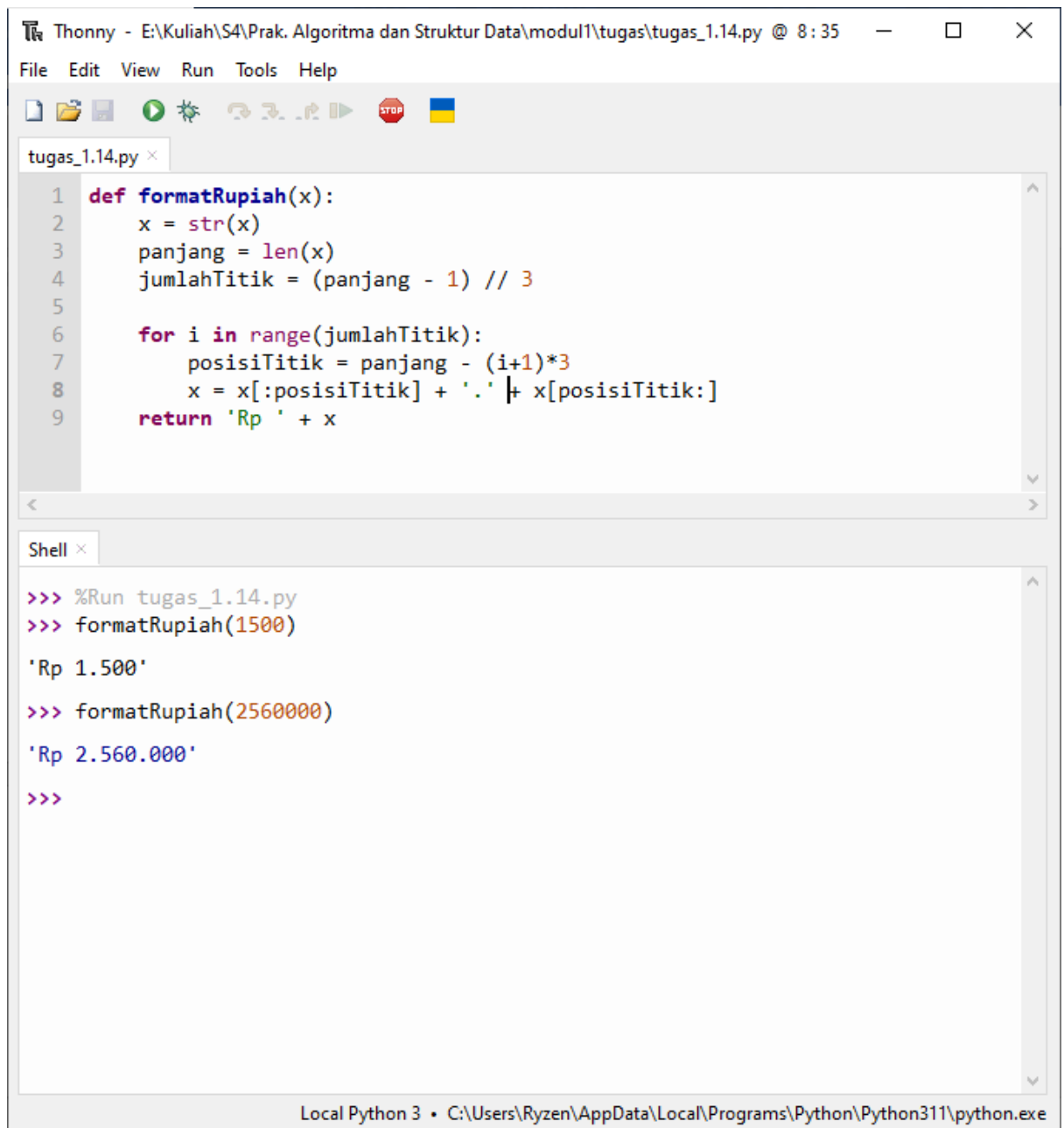
Shell x
>>> %Run 'tugas_1.11.py'
>>> apakahKabisat(1896)
    1896 tahun kabisat (habis dibagi 4)
True
>>> apakahKabisat(1897)
    1897 bukan tahun kabisat(sudah jelas)
False
>>> apakahKabisat(1900)
    1900 bukan tahun kabisat (meski habis dibagi 4 dan 100 tapi tidak habis dibagi 400)
False
>>> apakahKabisat(2000)
    2000 tahun kabisat (habis dibagi 400)
True
>>> apakahKabisat(2004)
    2004 tahun kabisat (habis dibagi 4)
True
>>> apakahKabisat(2100)
    2100 bukan tahun kabisat (meski habis dibagi 4 dan 100 tapi tidak habis dibagi 400)
False
>>> apakahKabisat(2400)
    2400 tahun kabisat (habis dibagi 400)
True
```

```
tugas_1.12.py x
1 import random
2 def tebakAngka():
3     angka = random.randint(1, 100)
4     jumlahTebakan = 0
5     maksTebakan = 7
6
7     print('Permainan tebak angka.')
8     print('Saya menyimpan sebuah angka bulat antara 1 sampai 100. Coba teba
9
10    while jumlahTebakan < maksTebakan:
11        tebak = int(input(f'Masukkan tebak ke-{jumlahTebakan+1}:> '))
12        jumlahTebakan += 1
13
14        if tebak < angka:
15            print('Itu terlalu kecil. Coba lagi.')
16        elif tebak > angka:
17            print('Itu terlalu besar. Coba lagi.')
18        else:
19            print('Ya. Anda benar')
20            return
21    print(f'Anda sudah gagal menebak {maksTebakan} kali. Angka yang benar a
22
23 print(tebakAngka())
```

```
Shell x
>>> %Run tugas_1.12.py
Permainan tebak angka.
Saya menyimpan sebuah angka bulat antara 1 sampai 100. Coba tebak.
Masukkan tebak ke-1:> 50
Itu terlalu besar. Coba lagi.
Masukkan tebak ke-2:> 30
Itu terlalu kecil. Coba lagi.
Masukkan tebak ke-3:> 35
Itu terlalu kecil. Coba lagi.
Masukkan tebak ke-4:> 40
Itu terlalu besar. Coba lagi.
Masukkan tebak ke-5:> 37
Itu terlalu besar. Coba lagi.
Masukkan tebak ke-6:> 36
Ya. Anda benar
None
>>> |
```

```
tugas_1.13.py ×
1 def katakan(angka):
2     if angka >= 1000000000:
3         return 'Maaf, Otak saya tidak sampai!!'
4
5     satuan = ['', 'satu', 'dua', 'tiga', 'empat', 'lima',
6               'enam', 'tujuh', 'delapan', 'sembilan', 'sepuluh', 'sebelas']
7
8     if angka < 12:
9         return satuan[angka]
10    elif angka < 20:
11        return katakan(angka - 10) + ' belas '
12    elif angka < 100:
13        return katakan(angka // 10) + ' puluh ' + katakan(angka % 10)
14    elif angka < 200:
15        return 'seratus ' + katakan(angka - 100)
16    elif angka < 1000:
17        return katakan(angka // 100) + ' ratus ' + katakan(angka % 100)
18    elif angka < 2000:
19        return 'seribu ' + katakan(angka - 1000)
20    elif angka < 1000000:
21        return katakan(angka // 1000) + ' ribu ' + katakan(angka % 1000)
22    elif angka < 1000000000:
23        return katakan(angka // 1000000) + ' juta ' + katakan(angka % 1000000)

Shell ×
>>> %Run tugas_1.13.py
>>> katakan(3125750)
'tiga juta seratus dua puluh lima ribu tujuh ratus lima puluh '
>>> katakan(1251000000)
'Maaf, Otak saya tidak sampai!!'
>>> |
```



The screenshot shows the Thonny Python IDE interface. The title bar reads "Thonny - E:\Kuliah\S4\Prak. Algoritma dan Struktur Data\modul1\tugas\tugas\_1.14.py @ 8:35". The menu bar includes "File", "Edit", "View", "Run", "Tools", and "Help". Below the menu is a toolbar with icons for opening files, saving, running, and debugging. The main editor window, titled "tugas\_1.14.py", contains the following Python code:

```
1 def formatRupiah(x):
2     x = str(x)
3     panjang = len(x)
4     jumlahTitik = (panjang - 1) // 3
5
6     for i in range(jumlahTitik):
7         posisiTitik = panjang - (i+1)*3
8         x = x[:posisiTitik] + '.' + x[posisiTitik:]
9     return 'Rp ' + x
```

Below the editor is a "Shell" window showing the execution of the script:

```
>>> %Run tugas_1.14.py
>>> formatRupiah(1500)
'Rp 1.500'
>>> formatRupiah(2560000)
'Rp 2.560.000'
>>>
```

The status bar at the bottom indicates "Local Python 3 • C:\Users\Ryzen\AppData\Local\Programs\Python\Python311\python.exe".