



# Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

<b>NIM</b>	<b>&lt;71230992&gt;</b>
<b>Nama Lengkap</b>	<b>&lt;Andriano Kurniawan Ladjeba&gt;</b>
<b>Minggu ke / Materi</b>	<b>12 / Set</b>

SAYA MENYATAKAN BAHWA LAPORAN PRAKTIKUM INI SAYA BUAT DENGAN USAHA SENDIRI TANPA MENGGUNAKAN BANTUAN ORANG LAIN. SEMUA MATERI YANG SAYA AMBIL DARI SUMBER LAIN SUDAH SAYA CANTUMKAN SUMBERNYA DAN TELAH SAYA TULIS ULANG DENGAN BAHASA SAYA SENDIRI.

SAYA SANGGUP MENERIMA SANKSI JIKA MELAKUKAN KEGIATAN PLAGIASI, TERMASUK SANKSI TIDAK LULUS MATA KULIAH INI.

PROGRAM STUDI INFORMATIKA  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS KRISTEN DUTA WACANA  
YOGYAKARTA  
2024

## BAGIAN 1: MATERI MINGGU INI (40%)

Pada bagian ini, tuliskan kembali semua materi yang telah anda pelajari minggu ini. Sesuaikan penjelasan anda dengan urutan materi yang telah diberikan di saat praktikum. Penjelasan anda harus dilengkapi dengan contoh, gambar/ilustrasi, contoh program (source code) dan outputnya. Idealnya sekitar 5-6 halaman.

### MATERI 1

#### Pengenalan dan Mendefinisikan Set

“Set” atau “himpunan” adalah salah satu tipe data Python yang dapat digunakan untuk menyimpan sekumpulan data yang semuanya berbeda. Salah satu sifat set Python adalah:

- Isi dari Set disebut sebagai anggota (member).
- Anggota dari Set harus bersifat immutable. Beberapa tipe data immutable pada Python: integer, float, string, tuple dan lain-lain. Dengan demikian list dan dictionary (mutable) tidak dapat dimasukkan ke dalam Set.
- Set sendiri bersifat mutable, artinya anda dapat menambah atau mengurangi isi dari sebuah Set. Karena itu Set tidak dapat dimasukkan ke dalam Set.

Untuk mendefinisikan Set, ada beberapa cara yang dapat dilakukan yaitu dengan menggunakan notasi {} dan fungsi set() seperti contoh berikut ini:

```
# dengan menggunakan {}
bilangan_genap = {2, 4, 6, 8, 10, 12}
bilangan_ganjil = {1, 3, 5, 7, 9, 11}
# dengan menggunakan fungsi set()
pernah_ke_bulan = set('Neil Armstrong', 'Buzz Aldrin')
```

Untuk mendefinisikan Set kosong anda tidak dapat menggunakan notasi {}, tetapi harus menggunakan fungsi set() seperti contoh berikut:

```
# dengan fungsi set()
pernah_ke_mars = set() # menghasilkan set kosong
data = {} # ini akan menghasilkan dictionary kosong
```

#### Pengaksesan Set

Set tidak memiliki indeks, karena itu anda tidak dapat mengakses anggota-anggota dari sebuah Set secara langsung menggunakan indeks. Perhatikan contoh program berikut ini:

```
nim = {'71200120', '71200195', '71200214'}
```

```

jumlah_nim = len(nim)
print(jumlah_nim) # akan menghasilkan output 3
# tampilkan isi set satu-persatu
for n in nim:
    print(n)

```

Output yang dihasilkan dari program tersebut adalah sebagai berikut:

```

3
71200120
71200195
71200214

```

Karena Set tidak memiliki indeks, sehingga tidak ada urutan posisi anggota, output yang dihasilkan urutannya berbeda dengan deklarasi Set sebelumnya. Ini karena posisi anggota tidak penting untuk tipe data Set.

Set adalah jenis data yang dapat diubah, yang berarti isinya dapat bertambah atau berkurang. Cara menggunakan fungsi add() untuk menambah anggota ke set ditunjukkan dalam program berikut:

```

plat_nomor = set() # definisikan sebuah set kosong
plat_nomor.add('AB 1890 XA') # tambahkan plat nomor 'AB 1890 XA'
plat_nomor.add('AD 6810 MT') # tambahkan plat nomor 'AD 6810 MT'
print(len(plat_nomor)) # jumlah anggota di dalam Set
plat_nomor.add('AB 1890 XA') # tambahkan plat yang sama sekali lagi
for plat in plat_nomor: # tampilkan semua plat nomor
    print(plat)

```

Output yang dihasilkan adalah sebagai berikut:

```

2
AB 1890 XA
AD 6810 MT

```

Set memiliki mekanisme untuk memastikan apakah anggota baru yang akan dimasukkan sudah ada di dalamnya (cek duplikasi), dan jika tidak, anggota tersebut dapat dimasukkan. Namun, jika anggota dengan nilai yang sama sudah ada di dalam Set, pemanggilan fungsi add() tidak akan menambah anggota ke dalam Set. Anda tidak perlu melakukan pengecekan duplikasi ini sendiri karena sudah ada di dalam fungsi add().

Ada beberapa cara untuk menghapus anggota dari set, seperti menggunakan fungsi discard(), remove(), pop(), dan clear(). Gambar 12.1 menunjukkan perbedaan antara empat fungsi tersebut. Program yang menunjukkan penghapusan anggota dari sebuah set dicontohkan di sini:

```

bilangan_prima = {13, 23, 7, 29, 11, 5} # hapus 5 dari set tersebut
bilangan_prima.remove(5)
print(bilangan_prima) # hapus 97 (tidak ada)
bilangan_prima.discard(97)
print(bilangan_prima) # ambil dan hapus salah satu
bilangan = bilangan_prima.pop()
print(bilangan)
print(bilangan_prima) # kosongkan set
bilangan_prima.clear()
print(bilangan_prima)

```

Ouputnya:

```

{29, 23, 7, 11, 13}
{29, 23, 7, 11, 13}
29
{23, 7, 11, 13}
set()

```

<b>discard()</b>	<b>remove()</b>	<b>pop()</b>	<b>clear()</b>
Menghapus satu elemen yang disebutkan	Menghapus satu elemen yang disebutkan	Mengambil salah satu dan menghapusnya dari set (tidak tentu)	Menghapus seluruh elemen di dalam set
Tidak ada error	Muncul error jika elemen yang dihapus tidak ada	Error jika set kosong	Tidak ada error

Gambar 12.1: Fungsi-fungsi untuk menghapus anggota dari Set.

Sementara fungsi `pop()` akan mengambil salah satu anggota secara acak dan mengeluarkannya dari Set, fungsi `discard()` tidak akan menghasilkan error jika anggota yang ingin dihapus tidak ada di dalam Set. Fungsi `pop()` berguna jika kita ingin memproses isi dari Set satu-persatu tanpa memperdulikan urutan atau posisi setiap anggota di dalam Set.

Apa yang terjadi jika Anda ingin mengubah nilai salah satu anggota dalam Set? Nilai anggota dalam set tidak dapat diubah secara langsung. Satu-satunya cara untuk melakukannya adalah melakukan operasi penggantian, atau penggantian, dengan menghapus anggota yang ingin diubah, kemudian memasukkan anggota baru dengan nilai yang diinginkan. Program berikut menunjukkan contohnya:

```
# Buat Set dari List
ikan = set(['koi', 'koki', 'kembung', 'salmon'])
print(ikan)
# ganti koi menjadi teri
ikan.remove('koi')
ikan.add('teri')
print(ikan)
```

Outputnya :

```
{'koki', 'koi', 'salmon', 'kembung'}
{'koki', 'teri', 'salmon', 'kembung'}
```

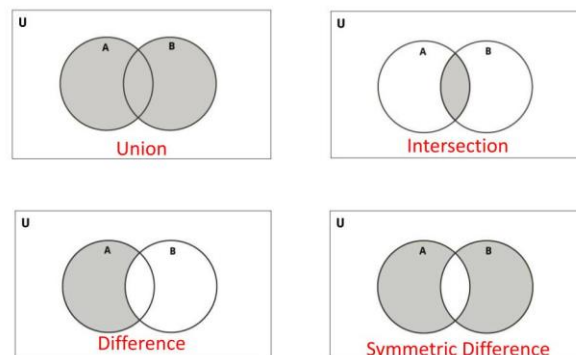
Untuk mengubah nilai "koi" menjadi "teri", pertama-tama harus menghapus anggota "koi" dan kemudian menambahkan anggota baru dengan nilai "teri". Sangat penting untuk diingat bahwa set tidak mengenal posisi atau urutan data, sehingga nilai "koi" tidak ada di output program dan digantikan oleh nilai "teri". Setelah itu, urutan isi Set ikan saat ditampilkan berubah. Urutan anggota dalam Set biasanya akan berubah setiap kali ada penambahan atau penghapusan.

## Operasi-Operasi pada Set

Operasi-operasi pada Set adalah operasi-operasi pada himpunan. Berikut ini adalah daftar operasi-operasi Set pada Python:

- Operator union dapat menggabungkan dua set menjadi satu dengan menggunakan operator `|` dan fungsi `union()`.
- Operator Intersection. Menghasilkan irisan dari dua Set. Dapat menggunakan operator `&` maupun fungsi `intersection()`.
- Operator Difference. Menghasilkan Set baru yang merupakan selisih dari dua Set yang dibandingkan. Dapat menggunakan operator `-` maupun fungsi `difference()`.
- Operator Symmetric Difference. Menghasilkan Set baru yang merupakan jumlah dari dua Set kecuali irisannya. Dapat menggunakan operator `^` maupun fungsi `symmetric_difference()`.

Gambar 12.2 menunjukkan operasi-operasi tersebut.



## Operator Union

Contoh penggunaan operator union dapat dilihat pada contoh program berikut ini:

```
merek_hp = {'Samsung', 'Apple', 'Xiaomi', 'Sony'}
merek_ac = {'LG', 'Samsung', 'Panasonic', 'Daikin', 'Sony'}
# union dari merek_hp dan merek_ac
gabungan = merek_hp | merek_ac
# bisa juga menggunakan gabungan = merek_hp.union(merek_ac)
print(gabungan)
```

Anggota dari Set merek\_hp dan anggota dari Set merek\_ac digabungkan oleh program, menghasilkan Set baru yang disebut gabungan. Ada duplikasi (anggota yang sama), jadi "Samsung" dan "Sony" hanya muncul satu kali di Set gabungan karena anggota harus unik. Hasilnya adalah sebagai berikut:

```
{'Apple', 'Xiaomi', 'Daikin', 'Samsung', 'Sony', 'LG', 'Panasonic'}
```

## Operator Intersection

Contoh penggunaan operator Intersection dapat dilihat pada program berikut:

```
renang = {'siti', 'mail', 'ikhsan', 'upin', 'ipin'}
tenis = {'joko', 'mail', 'ipin', 'upin', 'tejo'}
# suka renang dan tenis
renang_tenis = renang & tenis
print(renang_tenis)
```

Program akan menghasilkan output dari operasi intersection dari Set Renang dan Tennis, yang berarti anggota yang berada di kedua set, yaitu mail, upin, dan ipin, sekaligus. Operasi intersection menghasilkan output seperti:

```
{'upin', 'mail', 'ipin'}
```

## Operator Difference

Contoh penggunaan operator difference dapat dilihat pada program berikut:

```
# bisa berbahasa english
english = {'desi', 'tono', 'evan', 'miko', 'takashi', 'chaewon'}
# bisa berbahasa korea
korean = {'chaewon', 'yeona', 'erika', 'miko'}
```

```
# siapa yang hanya bisa bahasa korea?
only_korean = korean - english
print(only_korean)
# siapa yang hanya bisa bahasa english?
only_english = english - korean
print(only_english)
```

Set yang anggotanya adalah selisih dari dua set yang dibandingkan akan dihasilkan oleh operator perbedaan. Pada contoh ini, untuk mendapatkan anggota yang hanya bisa berbahasa Korea, Anda harus mencari selisih antara set Korea dan set English. Sebaliknya, jika Anda ingin mengetahui siapa saja yang hanya bisa berbahasa Inggris, Anda harus mencari selisih antara set Korea dan set English. Output berikut dihasilkan oleh program:

```
{'erika', 'yeona'}
{'evan', 'tono', 'takashi', 'desi'}
```

## Operator Symmetric Difference

Program berikut menunjukkan contoh penggunaan operator perbedaan simetris:

```
english = {'desi', 'tono', 'evan', 'miko', 'takashi', 'chaewon'}
korean = {'chaewon', 'yeona', 'erika', 'miko'}
# hanya bisa bicara satu bahasa saja
one_language = english ^ korean
print(one_language)
```

Operator perbedaan simetrik akan menghasilkan set baru yang merupakan gabungan dari dua set tetapi tidak termasuk irisannya. Contoh berikut menggunakannya untuk mendapatkan siapa saja yang hanya dapat berbicara dalam satu bahasa. Selain itu, biasanya dapat dihasilkan dari operasi berikut:

```
one_language = english.union(korean) - english.intersection(korean)
```

Output yang dihasilkan dari program tersebut adalah:

```
{'tono', 'desi', 'evan', 'erika', 'takashi', 'yeona'}
```

## MATERI 2

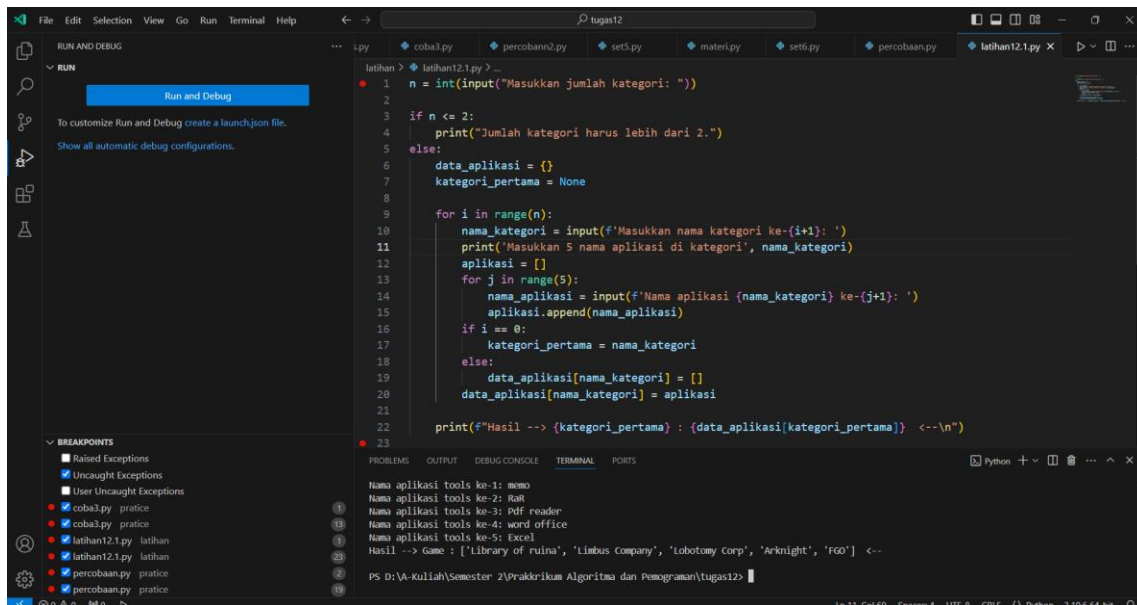
Penjelasan materi 2, dst... sesuai format ini.

## BAGIAN 2: LATIHAN MANDIRI (60%)

Pada bagian ini anda menuliskan jawaban dari soal-soal Latihan Mandiri yang ada di modul praktikum. Jawaban anda harus disertai dengan source code, penjelasan dan screenshot output.

### SOAL 1

#### Latihan 12.1



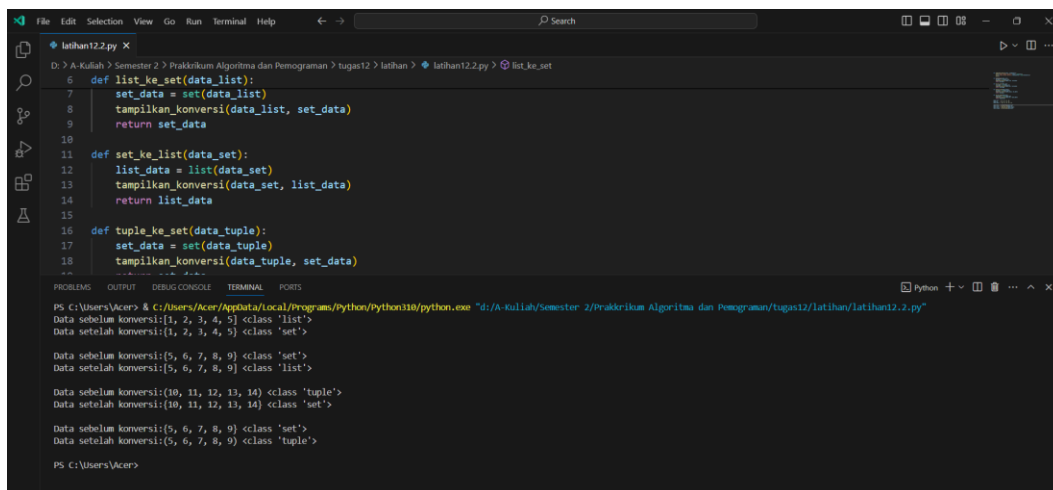
```
1 n = int(input("Masukkan jumlah kategori: "))
2
3 if n <= 2:
4     print("Jumlah kategori harus lebih dari 2.")
5 else:
6     data_aplikasi = {}
7     kategori_pertama = None
8
9     for i in range(n):
10        nama_kategori = input(f'Masukkan nama kategori ke-{i+1}: ')
11        print(f'Masukkan 5 nama aplikasi di kategori', nama_kategori)
12        aplikasi = []
13        for j in range(5):
14            nama_aplikasi = input(f'Nama aplikasi {nama_kategori} ke-{j+1}: ')
15            aplikasi.append(nama_aplikasi)
16            if i == 0:
17                kategori_pertama = nama_kategori
18            else:
19                data_aplikasi[nama_kategori] = []
20            data_aplikasi[nama_kategori] = aplikasi
21
22        print(f'Hasil --> {kategori_pertama} : {data_aplikasi[kategori_pertama]} <--\n")
23
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

Nama aplikasi tools ke-1: memo  
Nama aplikasi tools ke-2: RAR  
Nama aplikasi tools ke-3: Pdf reader  
Nama aplikasi tools ke-4: word office  
Nama aplikasi tools ke-5: Excel  
Hasil --> Game : ['Library of ruina', 'Limbus Company', 'Lobotomy Corp', 'Arknights', 'Igo'] <--

### SOAL 2

#### Latihan 12.2



```
6 def list_ke_set(data_list):
7     set_data = set(data_list)
8     tampilkan_konversi(data_list, set_data)
9     return set_data
10
11 def set_ke_list(data_set):
12     list_data = list(data_set)
13     tampilkan_konversi(data_set, list_data)
14     return list_data
15
16 def tuple_ke_set(data_tuple):
17     set_data = set(data_tuple)
18     tampilkan_konversi(data_tuple, set_data)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\Acer> & C:\Users\Acer\AppData\Local\Programs\Python\Python310\python.exe "d:/A-Kuliah/Semester 2/Praktikum Algoritma dan Pemrograman/tugas12/latihan/latihan12.2.py"

Data sebelum konversi:[1, 2, 3, 4, 5] <class 'list'>  
Data setelah konversi:{1, 2, 3, 4, 5} <class 'set'>

Data sebelum konversi:{5, 6, 7, 8, 9} <class 'set'>  
Data setelah konversi:[5, 6, 7, 8, 9] <class 'list'>

Data sebelum konversi:(10, 11, 12, 13, 14) <class 'tuple'>  
Data setelah konversi:{10, 11, 12, 13, 14} <class 'set'>

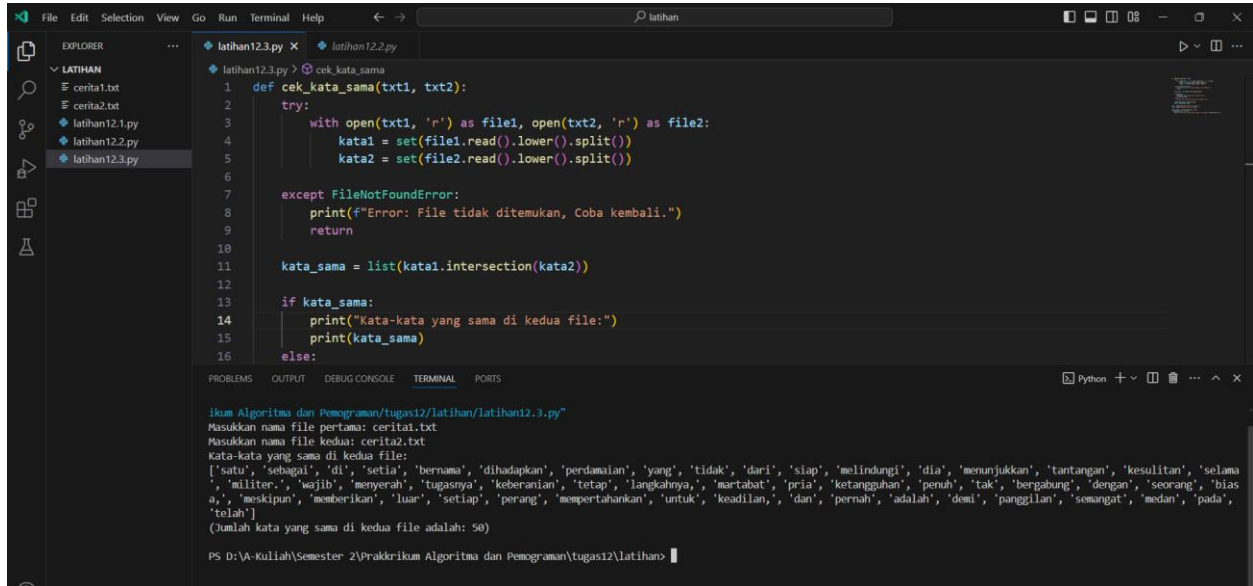
Data sebelum konversi:{5, 6, 7, 8, 9} <class 'set'>  
Data setelah konversi:(5, 6, 7, 8, 9) <class 'tuple'>

PS C:\Users\Acer>



## SOAL 3

### Latihan 12.3



The screenshot shows a Visual Studio Code editor window with a Python script named `latihan12.3.py`. The script defines a function `cek_kata_sama` that takes two file paths as arguments. It uses `try-except` blocks to handle file opening and reading. The words from both files are converted to sets, and their intersection is found using `intersection`. The result is printed as a list. The terminal output shows the execution of the script, which finds 50 common words between `cerita1.txt` and `cerita2.txt`.

```
def cek_kata_sama(txt1, txt2):
    try:
        with open(txt1, 'r') as file1, open(txt2, 'r') as file2:
            katal = set(file1.read().lower().split())
            kata2 = set(file2.read().lower().split())

    except FileNotFoundError:
        print(f"Error: File tidak ditemukan, Coba kembali.")
        return

    kata_sama = list(katal.intersection(kata2))

    if kata_sama:
        print("Kata-kata yang sama di kedua file:")
        print(kata_sama)
    else:
        print("Tidak ada kata yang sama di kedua file.")
```

ikun Algoritma dan Pemrograman\tugas12\latihan\latihan12.3.py"

Masukkan nama file pertama: cerita1.txt

Masukkan nama file kedua: cerita2.txt

Kata-kata yang sama di kedua file:

['satu', 'sebagai', 'di', 'setia', 'bernama', 'dihadapkan', 'perdamaian', 'yang', 'tidak', 'dari', 'siapa', 'melindungi', 'dia', 'menunjukkan', 'tantangan', 'kesulitan', 'selama', 'militer', 'wajib', 'menyerah', 'tugasnya', 'keberanian', 'tetap', 'langkahnya', 'martabat', 'pria', 'ketangguhan', 'penuh', 'tak', 'bergabung', 'dengan', 'seorang', 'bias', 'a', 'meskipun', 'memberikan', 'luan', 'setiap', 'perang', 'mempertahankan', 'untuk', 'keadilan', 'dan', 'pernah', 'adalah', 'demi', 'panggilan', 'semangat', 'medan', 'pada', 'telah']

(Jumlah kata yang sama di kedua file adalah: 50)

PS D:\A-Kuliah\Semester 2\Praktikum Algoritma dan Pemrograman\tugas12\latihan>

Github : <https://github.com/EchoGinDev/tugasAlpro12.git>