Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	**71230977**
Nama Lengkap	**MICHAEL HOSEA**
Minggu ke / Materi	**06 / PERCABANGAN DAN PERULANGAN KOMPLEKS**

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.

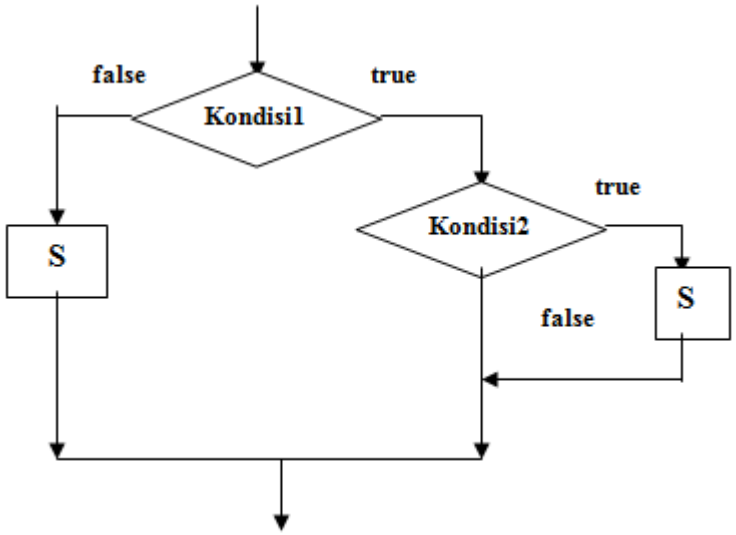
Materi 1: Struktur Percabangan Kompleks

Percabangan kompleks dalam pemrograman adalah struktur percabangan yang melibatkan lebih dari satu kondisi pemilihan dan juga lebih dari satu perintah yang dieksekusi berdasarkan kondisi-kondisi tersebut. Struktur ini memungkinkan pemrogram untuk mengontrol alur eksekusi program secara lebih fleksibel dan sesuai dengan berbagai skenario yang mungkin terjadi.

Percabangan kompleks sering kali melibatkan beberapa tingkat kondisi yang bersarang atau bertingkat, di mana setiap tingkatnya mengevaluasi kondisi tertentu dan menentukan jalur eksekusi yang sesuai berdasarkan hasil evaluasi tersebut.

Contohnya:

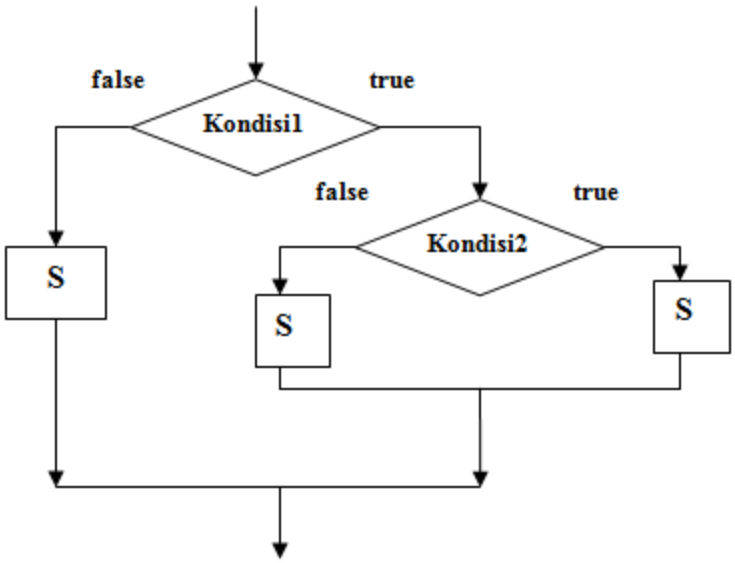
```
if kondisi1:
    if kondisi2:
        perintah1
        perintah2
else:
    perintah3
    perintah4
```



Contoh bentuk lainnya:

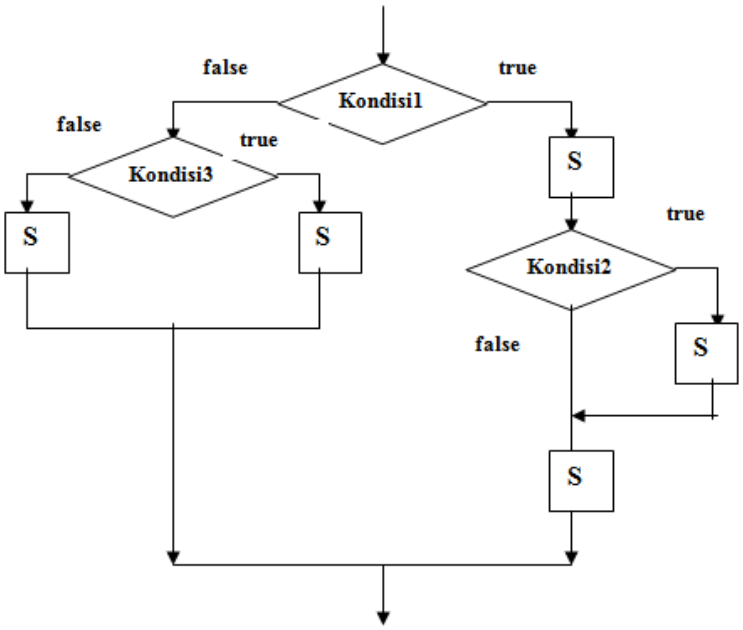
```
if kondisi1:
    if kondisi2:
        perintah1
```

```
        perintah2
    else:
        perintah3
        perintah4
else:
    perintah5
    perintah6
```



Contoh bentuk ketiga:

```
if kondisi1:
    perintah1
    if kondisi2:
        perintah2
        perintah3
else:
    if kondisi3:
        perintah4
        perintah5
    else:
        perintah6
        perintah7
```



Materi 2: Struktur Perulangan Kompleks bentuk “Break”

Break adalah pernyataan yang digunakan untuk menghentikan eksekusi dari sebuah loop atau switch-case secara paksa ketika kondisi tertentu terpenuhi. Saat break dieksekusi, kontrol program akan langsung keluar dari blok pengulangan atau switch-case, dan program akan melanjutkan eksekusi dari baris kode setelah blok tersebut.

Bentuk umum dari break adalah sebagai berikut:

```
while kondisi:
    pernyataan1
    pernyataan2
    if kondisi_untuk_break:
        break
    pernyataan3
```

Contoh dalam pemrograman Python:

```
i = 0
while i < 10:
    print(i)
    i += 1
    if i == 5:
        break
```

Pada contoh di atas, loop while akan dijalankan selama nilai i kurang dari 10. Ketika nilai i mencapai 5, pernyataan break akan dieksekusi, sehingga menghentikan eksekusi loop walaupun kondisi while masih terpenuhi. Sehingga hasilnya akan mencetak angka 0 hingga 4 saja.

MATERI 3: Struktur Perulangan Kompleks bentuk “Continue”

Continue adalah sebuah pernyataan dalam pemrograman yang digunakan untuk melanjutkan ke iterasi berikutnya dalam sebuah loop (misalnya, loop for atau while) tanpa menjalankan bagian kode yang berada setelah pernyataan continue pada iterasi saat ini. Dengan kata lain, ketika pernyataan continue dieksekusi, maka perulangan akan langsung melompat ke iterasi berikutnya, mengabaikan bagian kode yang tersisa dalam iterasi saat ini.

Bentuk umum dari pernyataan “Continue” adalah sebagai berikut:

```
for item in koleksi:
    if kondisi:
        continue
    pernyataan1
    pernyataan2
```

Berikut implementasinya dalam pemrograman Python:

```
for i in range(10):
    if i % 2 == 0:
        continue
    print(i)
```

Pada contoh di atas, loop “for” akan iterasi dari 0 hingga 9. Ketika “I” adalah bilangan genap (memiliki sisa pembagian 0 jika dibagi 2), pernyataan “continue” akan dieksekusi. Ini berarti bahwa kode “print(i)” tidak akan dieksekusi untuk bilangan genap, dan iterasi akan melompat ke nilai berikutnya dalam loop.

“Continue” sering digunakan dalam situasi di mana kita ingin melewati beberapa iterasi dalam sebuah loop berdasarkan suatu kondisi tertentu, tetapi tetap melanjutkan iterasi selanjutnya.

Perbedaan utama antara “continue” dan “break” adalah bahwa continue memungkinkan untuk melompati iterasi saat ini dan melanjutkan ke iterasi berikutnya, sedangkan “break” menghentikan loop secara keseluruhan.

MATERI 4: Penggunaan “break” dan “continue”

Perulangan bertingkat adalah penggunaan satu atau lebih perulangan di dalam perulangan lainnya. Dalam struktur ini, setiap iterasi dari perulangan luar akan berisi satu atau lebih iterasi dari perulangan dalam. Ini memungkinkan untuk mengulangi serangkaian tugas secara terstruktur dan terurut.

Bentuk umum dari perulangan bertingkat adalah sebagai berikut:

```
for variabel_luar in range(batas_luar):
    for variabel_dalam in range(batas_dalam):
        pernyataan
```

Berikut implementasinya dalam pemrograman Python:

```
for i in range(3): # Perulangan luar
    for j in range(2): # Perulangan dalam
        print("Nilai i:", i, "Nilai j:", j)
```

Hasil dari contoh di atas akan menjadi:

```
Nilai i: 0 Nilai j: 0
Nilai i: 0 Nilai j: 1
Nilai i: 1 Nilai j: 0
Nilai i: 1 Nilai j: 1
Nilai i: 2 Nilai j: 0
Nilai i: 2 Nilai j: 1
```

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

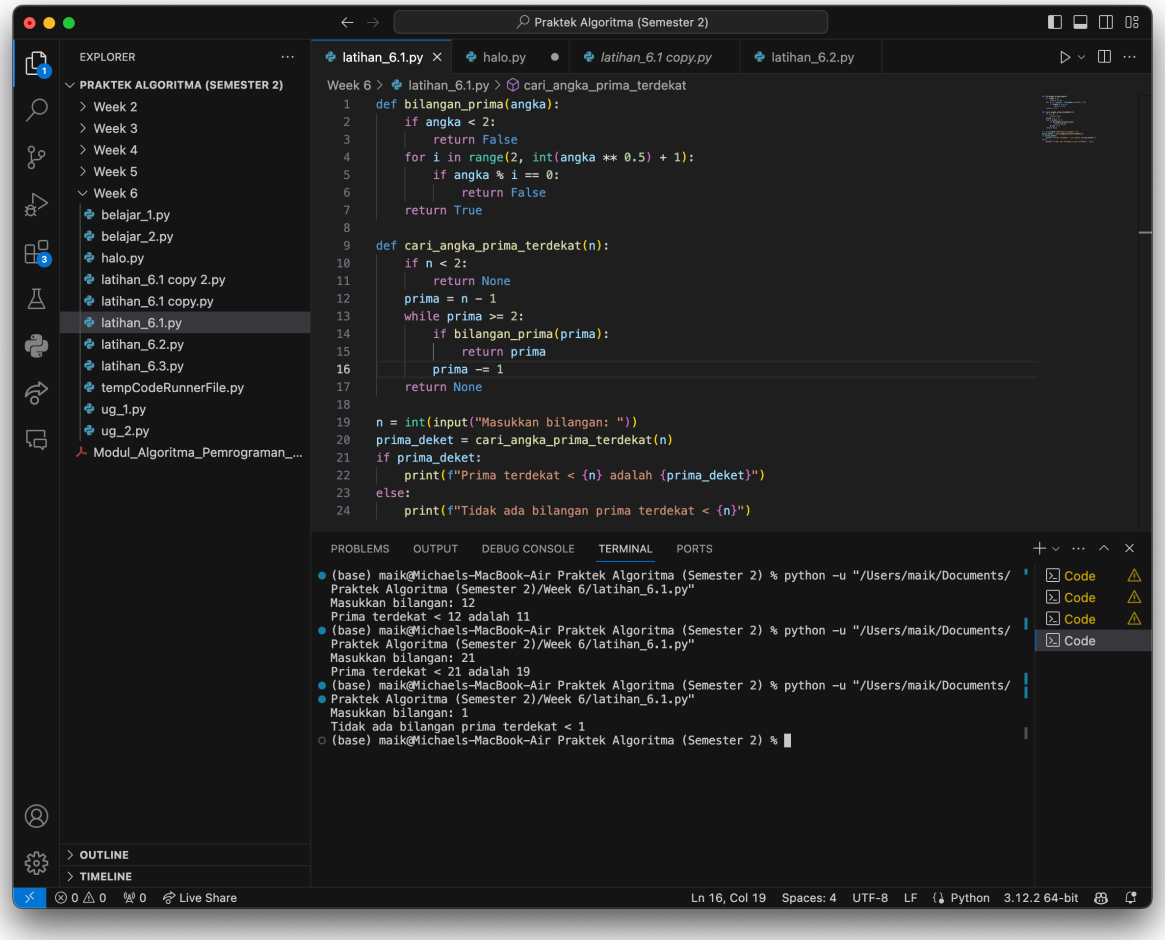
Buatlah program untuk mencari bilangan prima terdekat dari suatu bilangan yang diinputkan oleh pengguna (n) dan nilai bilangan prima tersebut < n. Contoh: input n=12, maka prima terdekat < 12 adalah 11 Contoh: input n=21, maka prima terdekat < 21 adalah 19.

- Source Code dan Output

```python
def bilangan_prima(angka):
    if angka < 2:
        return False
    for i in range(2, int(angka ** 0.5) + 1):
        if angka % i == 0:
            return False
    return True

def cari_angka_prima_terdekat(n):
    if n < 2:
        return None
    prima = n - 1
    while prima >= 2:
        if bilangan_prima(prima):
            return prima
        prima -= 1
    return None

n = int(input("Masukkan bilangan: "))
prima_deket = cari_angka_prima_terdekat(n)
if prima_deket:
    print(f"Prima terdekat < {n} adalah {prima_deket}")
else:
    print(f"Tidak ada bilangan prima terdekat < {n}")
```



- Penjelasan

1. Fungsi “bilangan_prima(angka)”:

- Fungsi ini bertugas untuk memeriksa apakah suatu bilangan adalah bilangan prima atau bukan.
- Jika bilangan kurang dari 2, maka fungsi akan mengembalikan “False”, karena bilangan prima haruslah lebih besar dari 1.
- Selanjutnya, dilakukan iterasi dari 2 hingga akar kuadrat dari bilangan tersebut, dan jika ditemukan bilangan pembagi dari bilangan yang diberikan, maka fungsi akan mengembalikan “False”. Jika tidak ditemukan bilangan pembagi, maka bilangan tersebut dianggap prima dan fungsi akan mengembalikan “True”.

2. Fungsi “cari_angka_prima_terdekat(n)”:

- Fungsi ini bertugas untuk menemukan bilangan prima terdekat yang lebih kecil dari bilangan yang diberikan (“n”).
- Jika “n” kurang dari 2, maka fungsi akan mengembalikan “None”, karena tidak mungkin ada bilangan prima yang lebih kecil dari 2.

- Selanjutnya, fungsi akan mencari bilangan prima terdekat yang lebih kecil dari “n” dengan cara mengurangi “prima” satu per satu, dimulai dari “n-1” hingga 2.
- Setiap “prima” yang dihasilkan akan diperiksa menggunakan fungsi “bilangan_prima(angka)”. Jika “prima” tersebut adalah bilangan prima, maka nilai “prima” tersebut akan dikembalikan. Jika tidak ditemukan bilangan prima, maka fungsi akan mengembalikan “None”.

3. Input dan Output:
- Pengguna diminta untuk memasukkan sebuah bilangan.
- Kemudian fungsi “cari_angka_prima_terdekat(n)” dipanggil untuk mencari bilangan prima terdekat yang lebih kecil dari bilangan yang dimasukkan.
- Hasilnya akan dicetak, yaitu bilangan prima terdekat yang ditemukan, atau pesan bahwa tidak ada bilangan prima terdekat yang lebih kecil dari bilangan yang dimasukkan.

SOAL 2

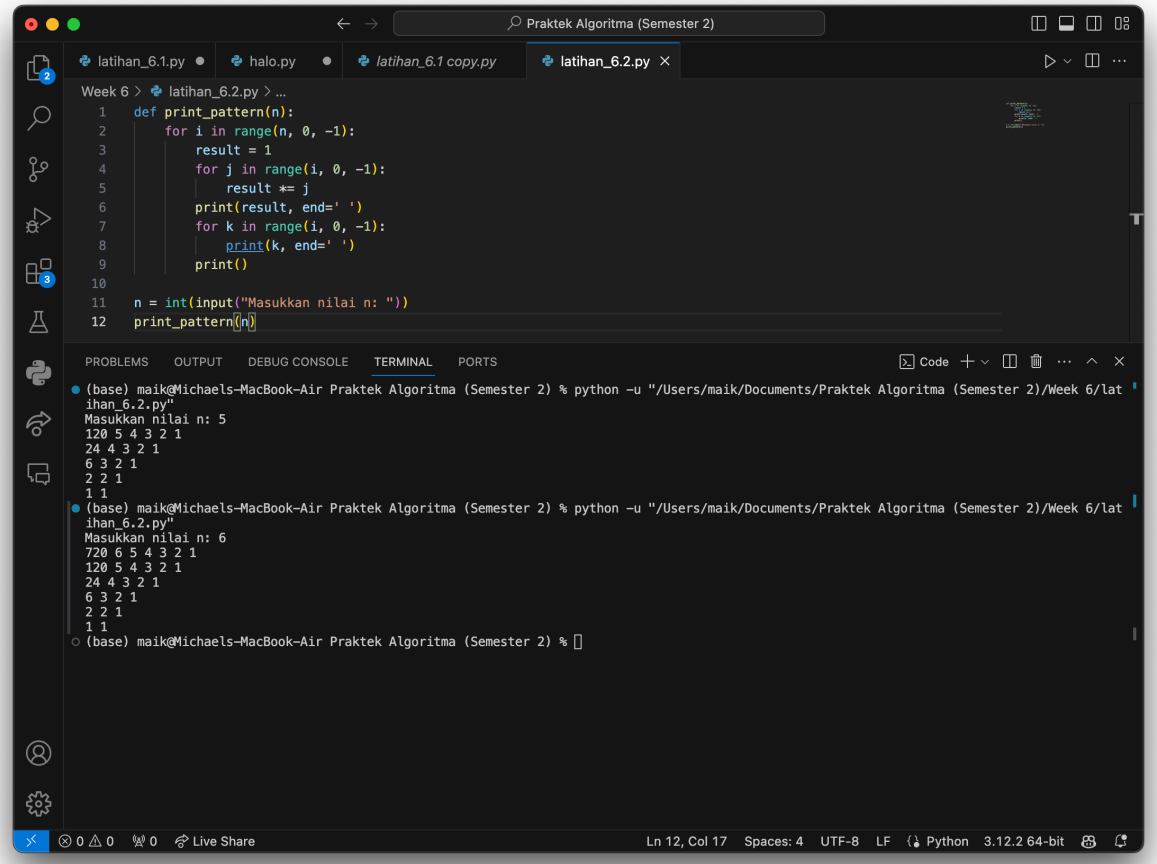
Buatlah program untuk menampilkan deret seperti di bawah ini. n diinputkan secara dinamis

contoh: n = 6
720 6 5 4 3 2 1
120 5 4 3 2 1
24 4 3 2 1
6 3 2 1
2 2 1
1 1

- Source Code dan Output

```
def print_pattern(n):
    for i in range(n, 0, -1):
        result = 1
        for j in range(i, 0, -1):
            result *= j
        print(result, end=' ')
        for k in range(i, 0, -1):
            print(k, end=' ')
        print()

n = int(input("Masukkan nilai n: "))
print_pattern(n)
```



- Penjelasan

  - Pada setiap baris ke-i, pertama-tama, itu mencetak hasil perkalian dari angka 1 hingga i (“i!”), diikuti dengan mencetak angka dari i hingga 1.
  - Ini diulangi dari n hingga 1.

1. “for i in range(n, 0, -1):”: Melakukan iterasi dari n hingga 1, menentukan baris pola.
2. “result = 1”: Inisialisasi variabel “result” sebagai 1.
3. “for j in range(i, 0, -1):”: Melakukan iterasi dari nilai i hingga 1 untuk menghitung hasil faktorial dari i.
   - “result *= j”: Mengalikan setiap nilai j ke variabel “result”.
4. “print(result, end=' ')”: Mencetak hasil faktorial dari i yang telah dihitung.
5. “for k in range(i, 0, -1):”: Melakukan iterasi dari nilai i hingga 1 untuk mencetak angka dari i hingga 1.
   - “print(k, end=' ')”: Mencetak setiap nilai k.
6. “print()”: Mencetak baris baru setelah mencetak angka dari i hingga 1 selesai.

SOAL 3

Buatlah program untuk menampilkan deret seperti di bawah ini. n diinputkan secara dinamis

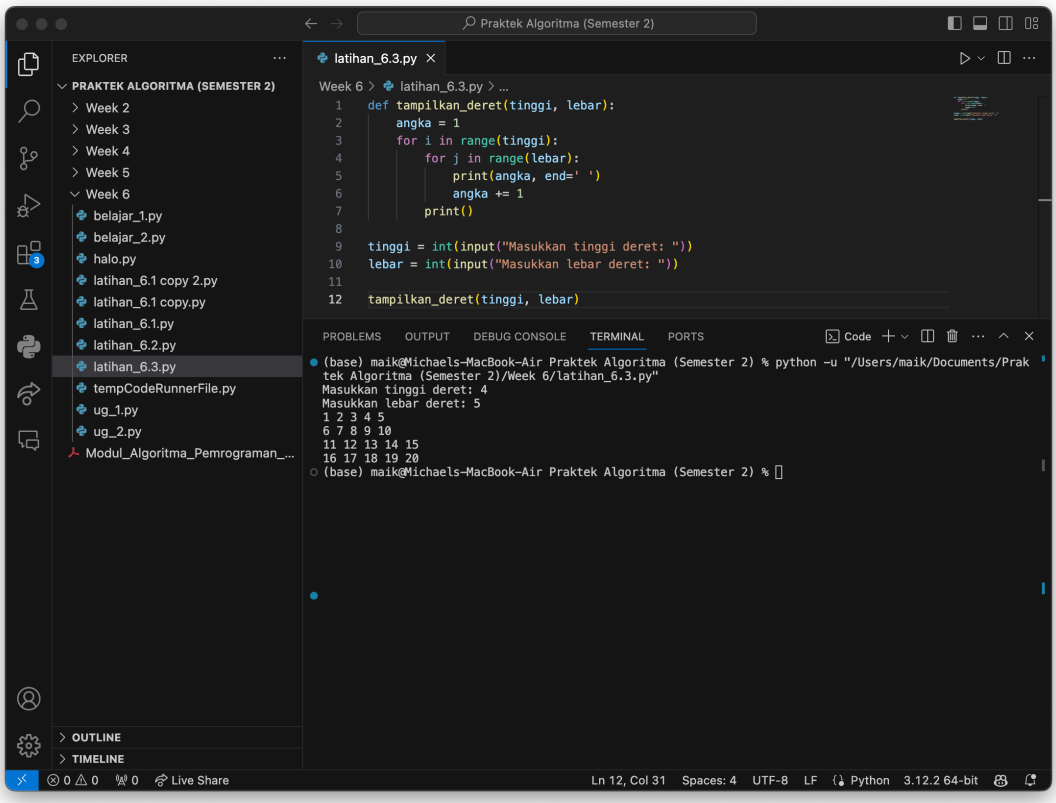
contoh: tinggi = 5, lebar = 4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
17 18 19 20

- Source Code dan Output

```python
def tampilkan_deret(tinggi, lebar):
    angka = 1
    for i in range(tinggi):
        for j in range(lebar):
            print(angka, end=' ')
            angka += 1
        print()

tinggi = int(input("Masukkan tinggi deret: "))
lebar = int(input("Masukkan lebar deret: "))

tampilkan_deret(tinggi, lebar)
```



- Penjelasan
  Fungsi “tampilkan_deret(tinggi, lebar)” ini bertujuan untuk mencetak deret angka dengan tinggi dan lebar tertentu.

  1. “angka = 1”: Inisialisasi variabel “angka” sebagai 1. Ini akan digunakan untuk mencetak deret angka dimulai dari 1.

  2. “for i in range(tinggi):”: Melakukan iterasi sebanyak “tinggi” kali untuk menentukan jumlah baris dalam deret.

  3. “for j in range(lebar):”: Melakukan iterasi sebanyak “lebar” kali untuk menentukan jumlah angka dalam setiap baris.

     - “print(angka, end=' ')”: Mencetak nilai “angka” dengan mengakhiri setiap angka dengan spasi.
     - “angka += 1”: Meningkatkan nilai “angka” untuk mencetak angka berikutnya.

  4. print()”: Mencetak baris baru setelah mencetak semua angka dalam satu baris.