



# Laporan Praktikum Algoritma dan Pemrograman

Semester Genap 2023/2024

NIM	71231051
Nama Lengkap	Amida A Ronsumbre
Minggu ke / Materi	05/ Struktur Kontrol Perulangan

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.

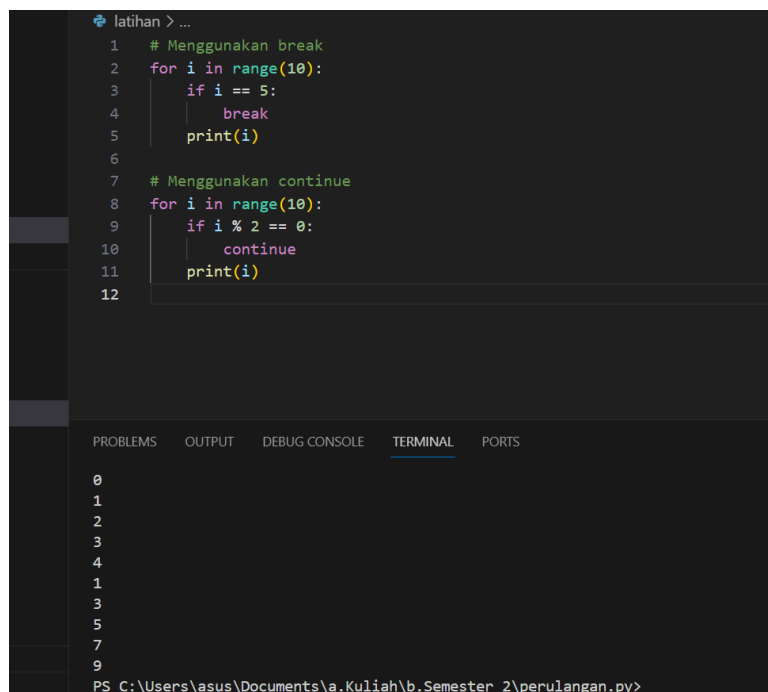
### MATERI 1

#### Penjelasan materi 1

#### • 5.3.1 Definisi Perulangan

Digunakan untuk mengulangi kode sejumlah iterasi yang diketahui atau untuk mengiterasi melalui item dalam suatu urutan seperti list, tuple, dictionary, atau string. Perulangan While: Digunakan untuk mengeksekusi blok kode selama kondisi tertentu bernilai True.

Menghentikan Loop: break digunakan untuk keluar dari loop sepenuhnya. continue digunakan untuk melewati iterasi saat ini dan melanjutkan ke iterasi berikutnya dalam loop.



```
latihan > ...
1  # Menggunakan break
2  for i in range(10):
3      if i == 5:
4          break
5      print(i)
6
7  # Menggunakan continue
8  for i in range(10):
9      if i % 2 == 0:
10         continue
11     print(i)
12
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
0
1
2
3
4
1
3
5
7
9
PS C:\Users\asus\Documents\A.Kuliah\b.Semester 2\perulangan.py>
```

Pilih antara for atau while tergantung pada skenario tertentu dan apakah jumlah iterasi diketahui atau tidak. Perulangan sangat berguna ketika memproses data atau melakukan tugas yang sama berulang kali.

#### • 5.3.2 Bentuk Perulangan for

Perulangan for pada Python digunakan ketika jumlah perulangan sudah diketahui sebelumnya atau saat melakukan operasi pada rentang data atau nilai tertentu. Fungsi range() memudahkan dalam menentukan rentang nilai yang akan diulang.

Fungsi Range:

range(stop): Membuat rentang dari 0 sampai stop-1.

range(start, stop, [step]): Membuat rentang dari start hingga stop dengan penambahan sebesar step.

```
1 for i in range(2, 101, 2):
2     print(i)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PO

```
82
84
86
88
90
92
94
96
98
```

Dengan menggunakan range() dan perulangan for, pengulangan dapat dilakukan dengan mudah dan efisien, baik untuk iterasi atas urutan nilai maupun untuk operasi pada rentang tertentu.

### • 5.3.3 Bentuk Perulangan While

Perulangan while digunakan ketika jumlah iterasi tidak diketahui sebelumnya. Contohnya, untuk memastikan input pengguna adalah bilangan genap:

```
latihan > ...
1  bilangan = 0
2  genap = False
3  while not genap:
4      bilangan = int(input('Masukkan bilangan genap: '))
5      if bilangan % 2 == 0:
6          genap = True
7      print(bilangan, 'adalah bilangan genap')
```

PS C:\Users\asus\Documents\A.Kuliah\B.Semester 2\perulangan.  
Semester 2\perulangan.py\latihan"  
Masukkan bilangan genap:

Program akan terus meminta input dari pengguna sampai bilangan yang dimasukkan adalah bilangan genap. Perulangan while sangat cocok untuk kasus di mana jumlah iterasi tidak dapat diprediksi sebelumnya. Pernyataan tersebut menjelaskan penggunaan perulangan `while` dalam situasi di mana jumlah iterasi tidak diketahui sebelumnya. Contohnya, dalam kode tersebut, program meminta pengguna untuk memasukkan bilangan genap. Namun, karena tidak diketahui berapa kali pengguna akan memasukkan bilangan ganjil sebelum memasukkan bilangan genap, maka perulangan `while` digunakan.

bilangan = 0: Variabel bilangan diinisialisasi dengan nilai 0.

genap = False: Variabel genap diinisialisasi dengan nilai False, menunjukkan bahwa kondisi bilangan genap belum terpenuhi.

while not genap:: Loop while akan terus berjalan selama genap bernilai False, artinya program akan terus berjalan selama pengguna belum memasukkan bilangan genap.

bilangan = int(input('Masukkan bilangan genap: ')): Program meminta input dari pengguna dan mengonversinya menjadi bilangan bulat.

if bilangan % 2 == 0: Program memeriksa apakah bilangan yang dimasukkan pengguna adalah bilangan genap.

genap = True: Jika bilangan yang dimasukkan pengguna adalah bilangan genap, variabel genap diubah menjadi True, dan loop while akan berhenti.

print(bilangan, 'adalah bilangan genap'):: Program mencetak pesan bahwa bilangan yang dimasukkan pengguna adalah bilangan genap.

Dengan demikian, kode ini memastikan bahwa program akan terus meminta input dari pengguna sampai bilangan yang dimasukkan adalah bilangan genap, sesuai dengan yang diminta oleh program.

### • 5.3.4 Penggunaan Break dan Continue

Perbedaan antara break dan continue adalah sebagai berikut:

break digunakan untuk menghentikan perulangan sepenuhnya saat kondisi tertentu terpenuhi. Setelah break dieksekusi, program akan keluar dari perulangan dan melanjutkan eksekusi pada baris kode setelah perulangan tersebut.

Contoh pengguna Break:

```
latihan > ...
1  for i in range(1, 11):
2      if i == 5:
3          break
4      else:
5          print(i)
6  print('Selesai')
```

continue digunakan untuk melewati iterasi saat ini dan melanjutkan ke iterasi berikutnya dalam perulangan. Jika continue dieksekusi, kode di bawahnya dalam blok perulangan tidak akan dijalankan, dan perulangan akan melanjutkan dengan iterasi berikutnya.

Contoh Pengguna Continue:

```
latihan > ...  
1   for i in range(1, 11):  
2       if i == 6:  
3           continue  
4       else:  
5           print(i)
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  
emester 2\perulangan.py\latihan"  
1  
2  
3  
4  
5  
7  
8  
9  
10  
PS C:\Users\asus\Documents\a.Kuliah\b.Semest
```

Dalam contoh di atas, perulangan akan menampilkan angka dari 1 sampai 10. Namun, dalam penggunaan continue, angka 6 dilewati dan tidak ditampilkan dalam output.

### • 5.3.5 Konversi dari Bentuk for Menjadi Bentuk while

Anda benar, perulangan for dapat dikonversi menjadi perulangan while dengan mempertimbangkan langkah-langkah berikut:

Tetapkan nilai awal.

Tentukan kondisi berhenti (nilai akhir).

Tentukan langkah atau penyesuaian yang diperlukan dalam setiap iterasi.

Contoh konversi dari perulangan for menjadi while:

```
python  
latihan > ...  
1   # Perulangan for  
2   for i in range(1, 11):  
3       print(i)  
4  
5   # Konversi ke perulangan while  
6   i = 1 # Nilai awal  
7   while i <= 10: # Kondisi berhenti  
8       print(i)  
9       i = i + 1 # Penyesuaian nilai setiap iterasi (step)
```

Dalam kasus ini, perulangan for mulai dari 1, berakhir di 10, dan langkahnya adalah 1. Konversi ke perulangan while menghasilkan hasil yang sama dengan mempertimbangkan nilai awal, kondisi berhenti, dan penyesuaian nilai setiap iterasi.

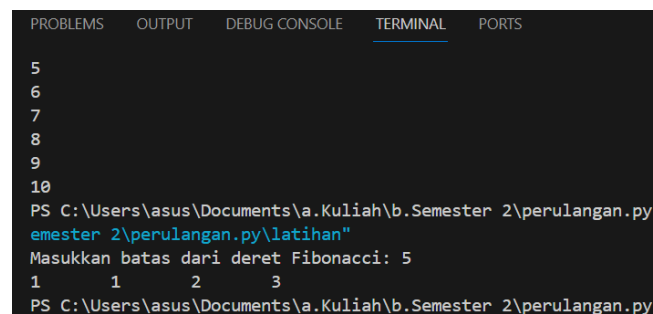
## MATERI 2

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

### • 5.4.1 Deret Bilangan

Berikut adalah program untuk menampilkan deret bilangan Fibonacci hingga batas tertentu yang dimasukkan oleh pengguna:

```
1 def fibonacci(batas):
2     bil1 = 1
3     bil2 = 1
4     # Tampilkan dua suku Fibonacci pertama
5     if bil1 < batas:
6         print(bil1, end='\t')
7         print(bil2, end='\t')
8     # Suku-suku berikutnya dari bil1 + bil2
9     suku_baru = bil1 + bil2
10    while suku_baru < batas:
11        print(suku_baru, end='\t')
12        # Geser bil1 dan bil2
13        bil1, bil2 = bil2, suku_baru
14        # Hitung lagi suku berikutnya
15        suku_baru = bil1 + bil2
16
```



```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
5
6
7
8
9
10
PS C:\Users\asus\Documents\A.Kuliah\B.Semester 2\perulangan.py
emester 2\perulangan.py\latihan"
Masukkan batas dari deret Fibonacci: 5
1      1      2      3
PS C:\Users\asus\Documents\A.Kuliah\B.Semester 2\perulangan.py
```

Untuk menjalankan program tersebut, pengguna diminta untuk memasukkan batas atas dari deret Fibonacci yang ingin ditampilkan.

Output program akan menampilkan deret bilangan Fibonacci mulai dari 1 hingga batas yang dimasukkan oleh pengguna.

Program tersebut akan meminta pengguna untuk memasukkan suku pertama dari deret konvergen, dan kemudian akan menampilkan deret bilangan konvergen yang dihasilkan. Program akan berhenti ketika mencapai nilai 1. Program pertama adalah program untuk menampilkan deret bilangan Fibonacci hingga batas tertentu yang dimasukkan oleh pengguna. Berikut penjelasannya:

Program menginisialisasi fungsi fibonacci(batas) yang akan menerima parameter batas, yang merupakan batas atas dari deret Fibonacci yang akan ditampilkan.

Di dalam fungsi fibonacci(), dua variabel bil1 dan bil2 diinisialisasi dengan nilai 1, sesuai dengan dua suku pertama dari deret Fibonacci.

Program memeriksa apakah dua suku pertama dari deret Fibonacci sudah kurang dari batas yang dimasukkan oleh pengguna. Jika ya, program akan mencetak kedua suku tersebut.

Selanjutnya, program menggunakan loop while untuk menghitung suku-suku berikutnya dari deret Fibonacci sampai suku tersebut melebihi batas yang dimasukkan oleh pengguna. Dalam setiap iterasi, suku berikutnya dihitung dengan menjumlahkan dua suku sebelumnya.

Setiap kali suku berikutnya dihitung, program mencetaknya.

Iterasi berlanjut sampai suku terakhir yang dihasilkan melebihi batas yang dimasukkan pengguna.

## • 5.4.2 Penggunaan Break

Program yang Anda sebutkan dirancang untuk menghitung rata-rata dari sejumlah nilai yang diinput oleh pengguna. Program akan terus meminta nilai input sampai pengguna memasukkan bilangan negatif atau nol. Berikut adalah penjelasan lebih detail dari program tersebut:

1. Fungsi `average()`: Program memulai dengan mendefinisikan sebuah fungsi `average()` yang tidak menerima parameter apa pun.
2. Inisialisasi Variabel: Di dalam fungsi, dua variabel diinisialisasi: `total` diatur ke 0 untuk menyimpan total semua input yang valid, dan `count` juga diatur ke 0 untuk menghitung berapa banyak input yang valid telah diberikan oleh pengguna.
3. Loop `while True`: Program kemudian masuk ke dalam loop `while` yang akan berjalan tanpa henti (`True` menandakan bahwa loop tidak memiliki kondisi berhenti yang eksplisit).
4. Meminta Input Pengguna: Dalam setiap iterasi loop, program meminta input dari pengguna. Input ini diubah menjadi tipe data integer dan disimpan dalam variabel `input_user`.
5. Mengecek Kondisi Berhenti: Program memeriksa apakah `input_user` kurang dari 1 (yaitu, nol atau negatif). Jika kondisi ini terpenuhi, loop akan dihentikan dengan perintah `break`.
6. Menghitung Total dan Jumlah Input: Jika input positif, nilai tersebut ditambahkan ke `total`, dan `count` (jumlah input) diinkremen.
7. Menghitung Rata-Rata: Setelah loop selesai (yaitu, pengguna memasukkan nol atau negatif), program memeriksa apakah `count` lebih besar dari 0 untuk menghindari pembagian dengan nol. Jika ya, program mengembalikan rata-rata dari total input yang valid dibagi dengan jumlah input yang valid. Jika tidak, program mengembalikan 0.
8. Menampilkan Rata-Rata: Di luar fungsi `average`, program memanggil fungsi tersebut, menyimpan hasilnya dalam variabel `hasil`, dan menampilkannya ke pengguna.

Ini adalah cara efisien untuk menghitung rata-rata dari input yang tidak diketahui jumlahnya dari awal, dengan menggunakan loop tak terbatas dan kondisi berhenti berbasis input pengguna.

```
latihan > ...
1  def average():
2      total = 0
3      count = 0
4      while True:
5          input_user = int(input('Masukkan nilai (nol atau negatif untuk berhenti): '))
6          if input_user <= 0:
7              break
8          else:
9              total += input_user
10             count += 1
11     if count > 0:
12         return total / count
13     else:
14         return 0
15
16 # Bagian utama program
17 hasil = average()

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS

1      1      2      3
PS C:\Users\asus\Documents\A.Kuliah\B.Semester 2\perulangan.py> python -u "c:\Users\asus\Documents\A.Kuliah\B.Semester 2\perulangan.py\latihan"
Masukkan nilai (nol atau negatif untuk berhenti): 3
Masukkan nilai (nol atau negatif untuk berhenti): 70
Masukkan nilai (nol atau negatif untuk berhenti): 80
Masukkan nilai (nol atau negatif untuk berhenti): 100
Masukkan nilai (nol atau negatif untuk berhenti): 48
Masukkan nilai (nol atau negatif untuk berhenti): 0
Rata-rata: 60.2
```

Dalam program ini, pengguna diminta untuk memasukkan nilai secara berulang. Program akan terus meminta input pengguna sampai pengguna memasukkan bilangan nol atau negatif. Setiap nilai yang dimasukkan oleh pengguna yang lebih besar dari nol akan ditambahkan ke total dan jumlah input yang valid akan dihitung. Setelah pengguna memasukkan bilangan negatif atau nol, program akan menghitung rata-rata dari semua input yang valid dan menampilkannya kepada pengguna.

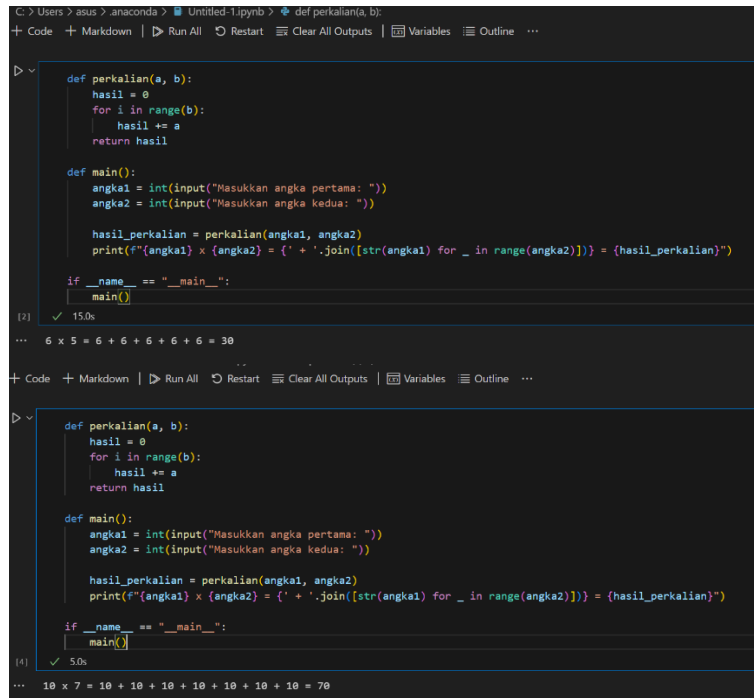


## BAGIAN 2: LATIHAN MANDIRI (60%)

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

Tulis jawaban anda untuk soal nomor 1 di sini.



```
C:\Users> asus > anaconda > Untitled-1.ipynb > def perkalian(a, b):
+ Code + Markdown | ▶ Run All ⏸ Restart 🗑 Clear All Outputs | 📄 Variables 📄 Outline ...

def perkalian(a, b):
    hasil = 0
    for i in range(b):
        hasil += a
    return hasil

def main():
    angka1 = int(input("Masukkan angka pertama: "))
    angka2 = int(input("Masukkan angka kedua: "))

    hasil_perkalian = perkalian(angka1, angka2)
    print(f"{angka1} x {angka2} = {' + '.join([str(angka1) for _ in range(angka2)] )} = {hasil_perkalian}")

if __name__ == "__main__":
    main()

[2] ✓ 15.0s
... 6 x 5 = 6 + 6 + 6 + 6 + 6 = 30

+ Code + Markdown | ▶ Run All ⏸ Restart 🗑 Clear All Outputs | 📄 Variables 📄 Outline ...

def perkalian(a, b):
    hasil = 0
    for i in range(b):
        hasil += a
    return hasil

def main():
    angka1 = int(input("Masukkan angka pertama: "))
    angka2 = int(input("Masukkan angka kedua: "))

    hasil_perkalian = perkalian(angka1, angka2)
    print(f"{angka1} x {angka2} = {' + '.join([str(angka1) for _ in range(angka2)] )} = {hasil_perkalian}")

if __name__ == "__main__":
    main()

[4] ✓ 5.0s
... 10 x 7 = 10 + 10 + 10 + 10 + 10 + 10 + 10 = 70
```

Dalam program ini, terdapat fungsi `perkalian()` yang menerima dua parameter `a` dan `b`, dan mengembalikan hasil perkalian dari kedua parameter tersebut dengan menggunakan penjumlahan. Fungsi `main()` digunakan untuk menerima input dari pengguna, memanggil fungsi `perkalian()` untuk menghitung hasilnya, dan kemudian mencetak hasilnya sesuai dengan format yang diminta.

Dengan demikian, program ini melakukan perkalian dengan menggunakan penjumlahan sebanyak yang diminta, dan kemudian menampilkan hasilnya sesuai dengan format yang diinginkan.

### SOAL 2

Tulis jawaban anda untuk soal nomor 2 di sini. Format untuk soal nomor 3 dan seterusnya juga sama.

```
C:\Users> asus > anaconda > def ganjil(bawah, atas):
+ Code + Markdown | Run All | Restart | Clear All Outputs | Variables | Outline ...
def ganjil(bawah, atas):
    if bawah < atas:
        return [x for x in range(bawah, atas + 1) if x % 2 != 0]
    else:
        return [x for x in range(bawah, atas - 1, -1) if x % 2 != 0]

def main():
    bawah = int(input("Masukkan batas bawah: "))
    atas = int(input("Masukkan batas atas: "))

    hasil = ganjil(bawah, atas)
    if bawah < atas:
        print("Karena bawah < atas, berarti dari kecil ke besar, maka hasilnya adalah: {}".format(', '.join(map(str, hasil))))
    else:
        print("Karena bawah > atas, berarti dari besar ke kecil, maka hasilnya adalah: {}".format(', '.join(map(str, hasil))))

if __name__ == "__main__":
    main()

[5] ✓ 370s
... Karena bawah < atas, berarti dari kecil ke besar, maka hasilnya adalah: 11, 13, 15, 17, 19, 21, 23, 25, 27, 29.
```

```
Code + Markdown | Run All | Restart | Clear All Outputs | Variables | Outline ...
def ganjil(bawah, atas):
    if bawah < atas:
        return [x for x in range(bawah, atas + 1) if x % 2 != 0]
    else:
        return [x for x in range(bawah, atas - 1, -1) if x % 2 != 0]

def main():
    bawah = int(input("Masukkan batas bawah: "))
    atas = int(input("Masukkan batas atas: "))

    hasil = ganjil(bawah, atas)
    if bawah < atas:
        print("Karena bawah < atas, berarti dari kecil ke besar, maka hasilnya adalah: {}".format(', '.join(map(str, hasil))))
    else:
        print("Karena bawah > atas, berarti dari besar ke kecil, maka hasilnya adalah: {}".format(', '.join(map(str, hasil))))

if __name__ == "__main__":
    main()

[6] ✓ 215s
... Karena bawah > atas, berarti dari besar ke kecil, maka hasilnya adalah: 97, 95, 93, 91, 89, 87, 85, 83.
```

a. Fungsi ganjil(bawah, atas):

Fungsi ini menerima dua parameter, bawah dan atas, yang merupakan batas bawah dan batas atas deret bilangan ganjil.

Jika bawah lebih kecil dari atas, fungsi ini mengembalikan daftar bilangan ganjil dari bawah hingga atas.

Jika bawah lebih besar dari atas, fungsi ini mengembalikan daftar bilangan ganjil dari bawah hingga atas secara terbalik.

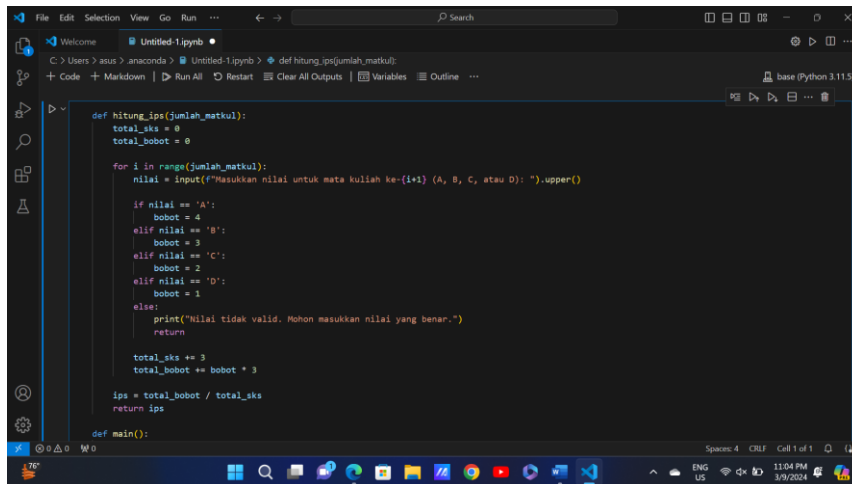
b. Fungsi main():

Fungsi ini menerima input dari pengguna untuk batas bawah dan batas atas.

Kemudian, fungsi ganjil() dipanggil dengan batas bawah dan batas atas yang dimasukkan oleh pengguna.

Hasilnya kemudian dicetak dengan format yang sesuai dengan kondisi perbandingan antara batas bawah dan batas atas.

## Soal 3



```
def hitung_ips(jumlah_matkul):
    total_sks = 0
    total_bobot = 0

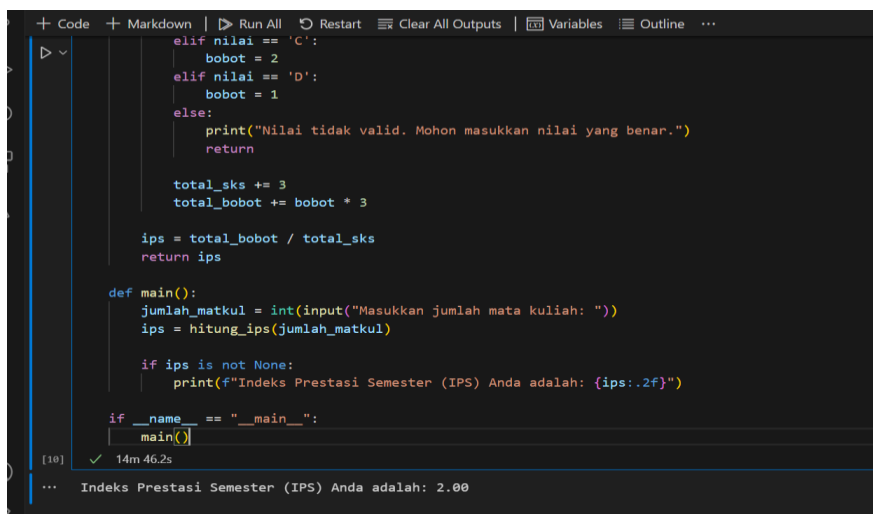
    for i in range(jumlah_matkul):
        nilai = input(f'Masukkan nilai untuk mata kuliah ke-{i+1} (A, B, C, atau D): ').upper()

        if nilai == 'A':
            bobot = 4
        elif nilai == 'B':
            bobot = 3
        elif nilai == 'C':
            bobot = 2
        elif nilai == 'D':
            bobot = 1
        else:
            print("Nilai tidak valid. Mohon masukkan nilai yang benar.")
            return

        total_sks += 3
        total_bobot += bobot * 3

    ips = total_bobot / total_sks
    return ips

def main():
```



```
        elif nilai == 'C':
            bobot = 2
        elif nilai == 'D':
            bobot = 1
        else:
            print("Nilai tidak valid. Mohon masukkan nilai yang benar.")
            return

        total_sks += 3
        total_bobot += bobot * 3

    ips = total_bobot / total_sks
    return ips

def main():
    jumlah_matkul = int(input("Masukkan jumlah mata kuliah: "))
    ips = hitung_ips(jumlah_matkul)

    if ips is not None:
        print(f"Indeks Prestasi Semester (IPS) Anda adalah: {ips:.2f}")

    if __name__ == "__main__":
        main()

[10] ✓ 14m 46.2s
... Indeks Prestasi Semester (IPS) Anda adalah: 2.00
```

a. Fungsi `hitung_ips(jumlah_matkul)`:

Fungsi ini menerima parameter `jumlah_matkul` yang merupakan jumlah mata kuliah.

Dalam perulangan, pengguna diminta untuk memasukkan nilai untuk setiap mata kuliah.

Kontrol percabangan digunakan untuk menentukan bobot dari setiap nilai yang dimasukkan.

Total SKS dan total bobot dihitung untuk seluruh mata kuliah.

IPS kemudian dihitung dengan rumus:  $IPS = \text{total bobot} / \text{total SKS}$ .

b. Fungsi `main()`:

Fungsi ini menerima input dari pengguna untuk jumlah mata kuliah.

Memanggil fungsi `hitung_ips()` untuk menghitung IPS berdasarkan input pengguna.

Jika IPS telah dihitung, maka hasilnya dicetak.

