VERSION 1.1

AGUSTUS 10, 2023

[PRAKTIKUM PEMROG. FUNGSIONAL]

MODUL 1 – Konsep pemrograman fungsional menggunakan bahasa python.

DISUSUN OLEH :
Fildzah Lathifah
Hania Pratiwi Ningrum

DIAUDIT OLEH
Fera Putri Ayu L., S.Kom., M.T.

PRESENTED BY: TIM LAB-IT

UNIVERSITAS MUHAMMADIYAH MALANG

PERSIAPAN MATERI

Praktikan diharapkan paham konsep fungsi dalam matematika serta dasar pemrograman (tipe data, variabel, percabangan, dan perulangan).

TUJUAN PRAKTIKUM

1. Praktikan mampu memahami paradigma pemrograman fungsional.
2. Praktikan mampu memahami tipe data dan variabel pada python.
3. Praktikan mampu memahami dan mengimplementasikan perulangan dan percabangan pada python.
4. Praktikan mampu memahami dan mengimplementasikan konsep fungsi pada paradigma pemrograman fungsional menggunakan python.

TARGET MODUL

Penguasaan materi:

1. Konsep Fungsi dalam Paradigma Fungsional
2. Tipe Data Sequence

PERSIAPAN SOFTWARE/APLIKASI

- Komputer/Laptop
- Sistem operasi Windows/Linux/Mac OS
- Pycharm/Google Collab/ Jupyter Notebook

MATERI POKOK

Pada pemrograman fungsional, modul juga dapat diakses melalui google collab agar lebih interaktif melalui tautan ini.

1 Paradigma Pemrograman Fungsional

Pada semester sebelumnya, kalian telah mempelajari mengenai pemrograman berbasis objek (PBO/OOP). Pada pemrograman tersebut, kalian menggunakan objek sebagai komponen untuk menulis sebuah program. Jika pada PBO kalian familiar dengan penggunaan kelas dan objek, pada pemrograman fungsional ini kalian akan mengenal lebih dalam mengenai fungsi.

1.1 Konsep Fungsi Dalam Matematika

Dalam matematika, fungsi adalah hubungan antara suatu himpunan input (domain) dengan himpunan output (codomain). Setiap elemen dalam domain dipetakan ke elemen yang unik dalam codomain oleh fungsi tersebut.

Pada pemrograman fungsional, konsep fungsi dalam matematika ini diterjemahkan dengan menggunakan fungsi sebagai komponen utama dalam penulisan program. Fungsi-fungsi dalam pemrograman fungsional berperan sebagai blok bangunan untuk memecahkan masalah dengan menghubungkan input dengan output secara deterministik, mirip dengan konsep fungsi dalam matematika.

1.2 Fitur Pemrograman Fungsional

Pemrograman Fungsional (Functional Programming) adalah suatu pendekatan dalam pemrograman yang berfokus pada penggunaan fungsi (function) sebagai komponen utama dalam menulis program. Pemrograman fungsional beranggapan bahwa program dapat dianggap sebagai kumpulan fungsi yang menerima input dan menghasilkan output tanpa memiliki keadaan (stateless). Artinya, untuk input yang sama, fungsi akan selalu mengembalikan output yang sama pula, tanpa mempengaruhi variabel atau data di luar fungsi tersebut. Dalam Pemrograman Fungsional, fungsi tersebut dikenal dengan istilah Pure Function (fungsi murni).

Selain Pure Function, terdapat beberapa fitur kunci yang membedakan paradigma fungsional dari paradigma pemrograman lainnya. Beberapa fitur tersebut meliputi Recursion (Rekursi), Higher-Order Functions, Lambda Functions, dll. Kita akan membahas fitur-fitur tersebut secara lebih detail dan bagaimana mengimplementasikannya menggunakan bahasa pemrograman Python pada modul selanjutnya.

Sedangkan pada modul 1 kali ini, kita akan belajar konsep fungsi pada pemrograman fungsional menggunakan bahasa python. Jika dalam PBO, kalian menggunakan bahasa pemrograman *Java*. Maka dalam pemrograman fungsional ini kita akan menggunakan bahasa pemrograman *python*. Selain *python*, sebenarnya banyak bahasa pemrograman yang bisa digunakan untuk menerapkan pemrograman fungsional yaitu Javascript, Clojure, Haskell, Ruby,

dll.

2 Bahasa Pemrograman Python

Python adalah bahasa pemrograman yang interpretatif dengan pendekatan Object Oriented Programming dan semantik dinamis. Ia menggabungkan kemampuan, sintaksis jelas, dan fungsionalitas pustaka standar yang besar. Python memiliki fitur menarik seperti tata bahasa yang mudah dipelajari, sistem pengelolaan data dan memori otomatis, serta pembaruan modul yang konsisten. Python banyak digunakan di berbagai sistem operasi, termasuk Linux, Microsoft Windows, Mac OS, Android, dan lainnya.

Selain itu, Python juga merupakan bahasa pemrograman yang sangat populer di dunia industri dan komunitas pengembangan perangkat lunak. Hal ini membuatnya menjadi pilihan yang tepat untuk dipelajari, karena banyaknya sumber daya, dukungan, dan komunitas yang dapat membantu dalam proses pembelajaran dan pengembangan. Untuk sejarah dan detail lebih lanjut tentang Python dapat ditemukan di link berikut.

2.1 Instalasi Python di pycharm IDE (optional)

Sebelum melakukan pemrograman menggunakan python, terlebih dahulu harus melakukan instalasi python pada laptop/komputer anda. Untuk langkah instalasi python, dapat mengikuti tutorial pada link berikut. Setelah berhasil menginstall python, selanjutnya anda membutuhkan IDE untuk menulis dan menjalankan program python yang dibuat. Terdapat banyak IDE yang dapat anda gunakan, tetapi kami merekomendasikan untuk menggunakan IDE pycharm. Untuk link panduan instalasi IDE Pycharm bisa kalian lihat disini.

2.2 Google Collab

Collaboratory, atau disingkat “Collab”, adalah produk dari Google Research. Collab memungkinkan siapa saja untuk menulis dan mengeksekusi kode python arbitrer melalui browser, dan sangat cocok untuk pembelajaran mesin, analisis data, dan pendidikan. Secara lebih teknis, Collab adalah layanan notebook Jupyter yang dihosting yang tidak memerlukan penyiapan untuk digunakan, sekaligus memberikan akses gratis ke sumber daya komputasi termasuk GPU. Oleh karena itu Collab sangat direkomendasikan untuk kalian yang memiliki komputer / pc dengan spesifikasi yang rendah.

Dengan menggunakan google collab, kalian dapat mengakses kode yang sudah kalian tulis, dimana saja dan kapan saja selama kalian punya akses internet.untuk memulai menggunakan google collab, kalian bisa klik tombol connect / hubungkan pada bagian kanan atas dari halaman ini. Setelah tanda centang berwarna hijau sudah muncul, berarti google collab siap untuk digunakan.

3 Tipe Data pada Python

Python memiliki banyak tipe data. Diantaranya adalah berikut :

Text Type:	`str`
Numeric Types:	`int`, `float`, `complex`
Sequence Types:	`list`, `tuple`, `range`
Mapping Type:	`dict`
Set Types:	`set`, `frozenset`
Boolean Type:	`bool`
Binary Types:	`bytes`, `bytearray`, `memoryview`

untuk mengetahui tipe data yang sedang digunakan, dapat menggunakan function type(). Untuk lebih jelasnya, kalian bisa menjalankan baris kode dibawah ini:

3.1 Text Type

3.1.1 String

```
[ ]  # coba jalankan kode dibawah ini agar lebih memahami tipe data
     print("Hello World")
     print(type("Hello Colab"))
```

```
Hello World
<class 'str'>
```

pada baris pertama pada kode diatas, kita melakukan output dengan menggunakan fungsi print() yang kita isikan dengan sebuah string “Hello Colab”, sehingga ketika kita membungkus “Hello Colab” dengan fungsi type() akan menampilkan tipe data dari “Hello Colab”, yaitu string (str)

3.2 Numeric Type

3.2.1 int dan float

```
[ ]  # tipe data numerik
     print(type(10)) #int
     print(type(10.0)) #float
```

```
<class 'int'>
<class 'float'>
```

Beberapa bahasa pemrograman seperti C/C++ atau java memiliki tipe data dengan ukuran yang berbeda seperti short untuk bilangan bulat dengan ukuran lebih kecil dari int, dan long untuk bilangan bulat dengan ukuran lebih besar dari int. Namun, pada Python, ukuran tipe data int akan disesuaikan secara dinamis berdasarkan nilai bilangan yang kita berikan. Kita bisa

mengecek hal tersebut dengan kode berikut:

```
[ ]  import sys
     print(sys.getsizeof(1))
     print(sys.getsizeof(12345))
     print(sys.getsizeof(1234567890))
     print(sys.getsizeof(12345678901234567890))
```

```
28
28
32
36
```

For our information, berikut adalah sedikit keterangan tentang size/ukuran suatu penyimpanan data di python 3:

```
Bytes  type        scaling notes
28     int         +4 bytes about every 30 powers of 2
37     bytes       +1 byte per additional byte
49     str         +1-4 per additional character (depending on max width)
48     tuple       +8 per additional item
64     list        +8 for each additional
224    set         5th increases to 736; 21nd, 2272; 85th, 8416; 341, 32992
240    dict        6th increases to 368; 22nd, 1184; 43rd, 2280; 86th, 4704; 171st, 9320
136    func def    does not include default args and other attrs
```

Tahu kan sekarang seperti apa maksud dari ukuran tipe data akan disesuaikan secara dinamis.. Bahkan tidak hanya tipe data, fungsi juga ada ukurannya lho, coba cek sekali lagi.

3.3 Sequence Type

sequence merupakan kumpulan dari sebuah nilai yang bertipe data sama (seperti array) namun, di python, tipe data sekuen sedikit berbeda dengan data array pada umumnya. Data sequence lebih dinamis daripada array, kita dapat menambahkan elemennya hingga berapapun yang kita inginkan (tidak ada batasan ukuran sebagaimana di array). Selain itu, sekuen juga dapat menyimpan tipe data yang berbeda. Mari kita pelajari beberapa jenis sekuen berikut,

3.3.1 List

Dalam bahasa pemrograman Python, struktur data yang paling dasar adalah sebuah daftar atau list. Setiap elemen-elemen dalam daftar disimpan secara berurutan dan diberi nomor posisi atau indeks seperti halnya array. Indeks pertama dalam list adalah nol, indeks kedua adalah satu dan seterusnya.

```
[ ]  # deklarasi list pada bahasa pemrograman python
     list1 = ['kimia', 'fisika', 1993, 2017]
     list2 = [1, 2, 3, 4, 5 ]
     list3 = ["a", "b", "c", "d"]

     print(type(list1))
     print(type(list2))
     print(type(list3))
```

```
<class 'list'>
<class 'list'>
<class 'list'>
```

```
[ ]  #Cara mengakses nilai di dalam list Python
     print ("list1[0]: ", list1[2])
     print ("list2[1:5]: ", list2[1:5])
```

```
list1[0]:  1993
list2[1:5]:  [2, 3, 4, 5]
```

```
[ ]  list1[2] = "biologi"  #mengganti nilai pada indeks ke 2
     print ("Nilai baru ada pada index 2 : ", list1[2])
```

```
Nilai baru ada pada index 2 :  biologi
```

Selain cara diatas, terdapat banyak cara untuk memodifikasi nilai atau isi dari suatu list, salah satunya yakni dengan metode append(). Untuk mempelajari method-method list lainnya, silahkan mengunjungi tautan berikut.

3.3.2 tuple

Sebuah tupel adalah urutan objek Python yang tidak berubah (***immutable***). Tupel adalah urutan, seperti daftar/list. Perbedaan utama antara tupel dan list adalah bahwa tupel tidak dapat diubah isi elemennya.

Tupel menggunakan tanda kurung, sedangkan List Python menggunakan tanda kurung siku. Membuat tuple semudah memasukkan nilai-nilai yang dipisahkan koma. Secara opsional, Anda dapat memasukkan nilai-nilai yang dipisahkan koma ini di antara tanda kurung juga. Sebagai contoh :

```
[ ]  #Contoh sederhana pembuatan tuple pada bahasa pemrograman python

     tup1 = ('fisika', 'kimia', 1993, 2017)
     tup2 = (1, 2, 3, 4, 5 )
     tup3 = "a", "b", "c", "d"

     print(type(tup1))
     print(type(tup2))
     print(type(tup3))
```

```
<class 'tuple'>
<class 'tuple'>
<class 'tuple'>
```

```
[ ]  #Cara mengakses nilai di dalam tuple
     print ("tup1[0]: ", tup1[0])
     print ("tup3[1:4]: ", tup3[1:4])
```

```
tup1[0]:  fisika
tup3[1:4]:  ('b', 'c', 'd')
```

Karena tuple bersifat immutable, maka isi dari suatu tuple tidak dapat dimodifikasi seperti pada list. Untuk mempelajari tuple lebih lanjut, anda dapat mengunjungi tautan berikut.

3.3.3 range

range adalah suatu fungsi pada python yang mereturn bilangan sequence dimulai dari 0 hingga mencapai batas tertentu. Biasanya range digunakan pada perulangan for dan while.

```
[ ]  a = range(5) # menghasilkan angka mulai dari 0 sampai 4

     print(a)

     # untuk menampilkan semua nilai dari range, perlu casting ke list dahulu
     print(list(a))
     print(type(a))
```

```
range(0, 5)
[0, 1, 2, 3, 4]
<class 'range'>
```

```
[ ]  b = range(2,10) # menghasilkan angka mulai dari 2 sampai 9

     print(b)

     #untuk menampilkan semua nilai dari range, perlu casting ke list dahulu
     print(list(b))
     print(type(b))
```

```
range(2, 10)
[2, 3, 4, 5, 6, 7, 8, 9]
<class 'range'>
```

3.4 Mapping Type

3.4.1 Dictionary

Dictionary seperti buku alamat, dengan buku alamat kita bisa mencari alamat atau detail kontak hanya menggunakan nama orang yang kita cari. Kita mengasosiasikan key (nama) dengan value (detail). Catatan: key harus bersifat unik, kita tidak bisa menemukan informasi yang tepat jika ada dua orang yang mempunyai nama yang sama dalam buku alamat kita bukan. Untuk key/kunci, kita hanya bisa menggunakan objek immutable (seperti string). Sedangkan untuk value dalam dictionary, kita bisa menggunakan objek mutable maupun immutable seperti list, angka, tuple, maupun string.

```
[ ]  #membuat dictionary
     dict1 = {'Nama': 'Seneca', 'Umur': 15, 'Kelas': '10'}
     print(type(dict1))

     #mengkases value dictionary berdasarkan key-nya.
     print("dict1['Nama']: ", dict1['Nama'])
     print("dict1['Umur']: ", dict1['Umur'])
```

```
<class 'dict'>
dict1['Nama']:  Seneca
dict1['Umur']:  15
```

```
[ ]  dict1['Hobby'] = "Merenung" # Menambah entri baru
     dict1['Umur'] = 20; # Mengubah value yang sudah ada berdasarkan keynya


     print(dict1)
```

```
{'Nama': 'Seneca', 'Umur': 20, 'Kelas': '10', 'Hobby': 'Merenung'}
```

Untuk mempelajari dictionary lebih lanjut, anda dapat mengunjungi tautan berikut.

3.5 Boolean Type

Menyatakan benar ***True*** yang bernilai 1, atau salah ***False*** yang bernilai 0

```
[ ]  print(type(True))
     print(type(False))
     #penulisan true dan false wajib diawali huruf kapital
```

```
<class 'bool'>
<class 'bool'>
```

4 Variabel pada Python

Pada python, untuk menuliskan sebuah variabel tidak perlu mendefinisikan tipe data yang ingin disimpan di sebuah variabel. Sebuah variabel dapat berganti tipe data sedinamis mungkin. Jalankan baris kode di bawah agar kalian lebih memahami penggunaan variabel pada python.

```
[ ]  a = 5
     print(a) #5
     print(type(a)) #mencetak tipe data dari variabel a yaitu int
     a = "Ganti Variable" #pergantian nilai dari variable a beserta tipe datanya
     print(a) #"Ganti Variable"
     print(type(a)) #mencetak tipe data dari variabel a yang baru yaitu str (string)
```

```
5
<class 'int'>
Ganti Variable
<class 'str'>
```

Di dalam python, untuk menggabungkan antara variabel satu dengan variabel lainnya, ada sebuah aturan, yaitu variabel dengan tipe data yang berbeda tidak dapat digabungkan kecuali melalui casting. Agar lebih memahami mengenai penggabungan variabel dan casting kalian bisa menjalankan baris kode di bawah ini

```
[ ]  # akan menampilkan 12 karena 5 dan 7 merupakan tipe data int
     5 + 7
```

```
12
```

Selain bisa menggabungkan tipe data numerik, anda juga bisa menggabungkan tipe data teks dan sekuens.

```
[ ]  #Menggabungkan tipe data String
     "Hallo" + " dunia"
```

```
'Hallo dunia'
```

```
[ ]  #menggabungkan tipe data list
     [1,2,3,4] + [5,6,7,8]
```

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

```
[ ]  a = 5
     b = 10
     print(a+b)
```

```
15
```

```
[ ]  # akan menampilkan error karena dua tipe data berbeda tidak dapat digabung
     5 + "Halo"
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-18-0dd7ff18140d> in <module>()
      1 # akan menampilkan error karena dua tipe data berbeda tidak dapat digabung
----> 2 5 + "Halo"

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

```
[ ]  c = 5
     d = "Halo"
     print(c+d)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-19-519beec7e877> in <module>()
      1 c = 5
      2 d = "Halo"
----> 3 print(c+d)

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

4.1 Casting

Casting berguna untuk mengubah tipe data yang berbeda agar dapat digabungkan/dioperasikan secara bersamaan.

```
[ ]  #merubah tipe data int menjadi str
     print("Contoh casting int ke string")
     x = 1 #pada awalnya tipe data dari x adalah int,
           #karena x memiliki nilai sebuah bilangan integer
     print(type(x))
     x = str(x) #fungsi str() digunakan untuk merubah variabel didalamnya
                #(yaitu variabel x) untuk diubah menjadi string
     print(type(x)) #setelah diubah dengan method str(),
                    #variabel x sekarang bertipe data string / str
```

```
Contoh casting int ke string
<class 'int'>
<class 'str'>
```

```
[ ]  #merubah tipe data float menjadi int
     print("Contoh casting int ke string")
     y = 20.3 #pada awalnya tipe data dari y adalah float, karena y memiliki nilai
              #sebuah float (bilangan yang memiliki koma / floating point)
     print(type(y))
     y = int(y) #fungsi int() digunakan untuk merubah variabel didalamnya
                #(yaitu variabel y) untuk diubah menjadi integer
     print(type(y)) #setelah diubah dengan method int(), variabel y
                    #sekarang bertipe data integer / int
```

```
Contoh casting int ke string
<class 'float'>
<class 'int'>
```

```
[ ]  #setelah mempelajari beberapa metode untuk melakukan casting tipe data,
     #kalian bisa mencoba program dibawah ini untuk menerapkan casting tipe data
     #dengan penggabungan variabel
     a = "hello"
     b = 10

     print(a + " Colab, kamu berumur " + str(b) + " tahun")
```

```
hello Colab, kamu berumur 10 tahun
```

Selain bisa mengubah tipe data primitive, casting juga bisa mengubah tipe data sequence.

```
[ ]  print("contoh casting list menjadi tuple")
     ini_list = [1,1,2,3,4,5]
     print(type(ini_list))
     ini_tuple = tuple(ini_list) #proses casting
     print(type(ini_tuple))
```

```
contoh casting list menjadi tuple
<class 'list'>
<class 'tuple'>
```

5 Percabangan dalam Python

```
[ ]  # if
     a = 33
     b = 200
     if b > a:
       print("b is greater than a")
```

```
b is greater than a
```

```
[ ]  # elif
     a = 33
     b = 33
     if b > a:
       print("b is greater than a")
     elif a == b:
       print("a and b are equal")
```

```
a and b are equal
```

```
[ ]  # if, elif, else
     a = 200
     b = 33
     if b > a:
       print("b is greater than a")
     elif a == b:
       print("a and b are equal")
     else:
       print("a is greater than b")
```

```
a is greater than b
```

```
intermediate
import os

# Inisialisasi data peserta
data_peserta = []

# Fungsi untuk menambah data peserta oleh admin
def tambah_peserta():
    os.system('cls' if os.name == 'nt' else 'clear')  # Clear terminal
    id_peserta = len(data_peserta)
    print("\nID Peserta baru:", id_peserta)  # Menampilkan ID peserta yang baru ditambahkan
    nama = input("Masukkan nama peserta: ")
    nilai_matematika = int(input("Masukkan nilai Matematika peserta: "))
    nilai_bahasa_indonesia = int(input("Masukkan nilai Bahasa Indonesia peserta: "))
    nilai_ipa = int(input("Masukkan nilai IPA peserta: "))
    nilai_ips = int(input("Masukkan nilai IPS peserta: "))

    rata_rata_nilai = (nilai_matematika + nilai_bahasa_indonesia + nilai_ipa + nilai_ips) / 4
    hasil_akhir = "Lolos" if rata_rata_nilai >= 75 else "Tidak Lolos"

    data_peserta.append([id_peserta, nama, nilai_matematika, nilai_bahasa_indonesia, nilai_ipa, nilai_ips, rata_rata_nilai, hasil_akhir])

    print("\nData peserta berhasil ditambahkan.")
    input("\nTekan Enter untuk kembali ke menu...")

# Fungsi untuk mengedit nilai peserta oleh admin
def edit_nilai():
    id_peserta = int(input("Masukkan ID peserta yang akan diedit: "))
    if id_peserta < len(data_peserta):
        nilai_matematika = int(input("Masukkan nilai Matematika baru untuk peserta {}: ".format(id_peserta)))
        nilai_bahasa_indonesia = int(input("Masukkan nilai Bahasa Indonesia baru untuk peserta {}: ".format(id_peserta)))
        nilai_ipa = int(input("Masukkan nilai IPA baru untuk peserta {}: ".format(id_peserta)))
        nilai_ips = int(input("Masukkan nilai IPS baru untuk peserta {}: ".format(id_peserta)))

        rata_rata_nilai = (nilai_matematika + nilai_bahasa_indonesia + nilai_ipa + nilai_ips) / 4

        data_peserta[id_peserta][2] = nilai_matematika
        data_peserta[id_peserta][3] = nilai_bahasa_indonesia
        data_peserta[id_peserta][4] = nilai_ipa
        data_peserta[id_peserta][5] = nilai_ips
        data_peserta[id_peserta][6] = rata_rata_nilai
        data_peserta[id_peserta][7] = "Lolos" if rata_rata_nilai >= 75 else "Tidak Lolos"

        print("\nNilai peserta berhasil diubah.")
    else:
        print("\nID peserta tidak valid.")
    input("Tekan Enter untuk kembali ke menu...")

# Fungsi untuk menampilkan nilai dan hasil akhir peserta dalam bentuk tabel
def tampilkan_hasil_peserta(id_peserta):
    os.system('cls' if os.name == 'nt' else 'clear')  # Clear terminal
    if id_peserta < len(data_peserta):
        print("\nHasil Peserta:")
        print("{:<12} {:<15} {:<10} {:<20} {:<10} {:<10}".format("ID", "Nama", "Matematika", "Bahasa Indonesia", "IPA", "IPS"))
        peserta = data_peserta[id_peserta]
        print("{:<12} {:<15} {:<10} {:<20} {:<10} {:<10}".format(peserta[0], peserta[1], peserta[2], peserta[3], peserta[4], peserta[5]))
        print("\nRata-rata Nilai: {:.2f}".format(peserta[6]))
        print("Hasil Akhir: {}".format(peserta[7]))
    else:
        print("ID peserta tidak valid.")
    input("Tekan Enter untuk kembali ke menu...")

# Fungsi untuk menampilkan data peserta dalam bentuk tabel
def tampilkan_data_peserta_tabel():
    os.system('cls' if os.name == 'nt' else 'clear')  # Clear terminal
    print("\nData Peserta:")
    print("{:<5} {:<15}".format("ID", "Nama"))
    for peserta in data_peserta:
        print("{:<5} {:<15}".format(peserta[0], peserta[1]))

# Main program
while True:
    os.system('cls' if os.name == 'nt' else 'clear')  # Clear terminal
    print("\nMenu:")
    print("1. Admin")
    print("2. Peserta")
    print("3. Keluar")

    peran = input("Pilih peran Anda (1/2/3): ")

    if peran == "1":  # Menu Admin
        os.system('cls' if os.name == 'nt' else 'clear')  # Clear terminal
        print("\nMenu Admin:")
        print("1. Tambah Peserta")
        print("2. Edit Nilai")
        print("3. Kembali ke Menu Utama")

        pilihan_admin = input("Pilih menu Admin (1/2/3): ")

        if pilihan_admin == "1":
            tambah_peserta()
        elif pilihan_admin == "2":
            tampilkan_data_peserta_tabel()
            edit_nilai()
        elif pilihan_admin == "3":
            continue
        else:
            print("Pilihan Admin tidak valid.")

    elif peran == "2":  # Menu Peserta
        os.system('cls' if os.name == 'nt' else 'clear')  # Clear terminal
        tampilkan_data_peserta_tabel()
        id_peserta = int(input("Masukkan ID peserta Anda: "))
        tampilkan_hasil_peserta(id_peserta)

    elif peran == "3":  # Keluar
        break

    else:
        print("Pilihan tidak valid. Silakan pilih peran yang benar.")
```

terhadap sequence (list, tuple, dictionary, set atau string)

```
[ ]  #for loop pada list
     fruits = ["apple", "banana", "cherry"]
     for x in fruits:
       print(x)
```

```
apple
banana
cherry
```

```
[ ]  #for loop pada string
     for x in "banana":
       print(x)
```

```
b
a
n
a
n
a
```

```
[ ]  #menggunakan keyword break untuk keluar dari for loop
     #program akan berhenti ketika nilai x adalah "banana"
     fruits = ["apple", "banana", "cherry"]
     for x in fruits:
       print(x)
       if x == "banana":
         break
```

```
apple
banana
```

```
[ ]  #menggunakan keyword continue
     #ketika x bernilai "banana",
     #maka akan di skip dan dilanjut iterasi selanjutnya
     fruits = ["apple", "banana", "cherry"]
     for x in fruits:
       if x == "banana":
         continue
       print(x)
```

```
apple
cherry
```

```
[ ]  #menggunakan range pada for loop
     for x in range(6):
       print(x)
```

```
0
1
2
3
4
5
```

untuk lebih lengkapnya mengenai for loop, bisa mengunjungi link berikut

6.2 While Loop

Pada python, while loop mirip dengan while loop yang ada pada bahasa pemrograman lainnya. Tidak seperti for loop yang memiliki aturan khusus. While loop pada python cukup mudah dipahami. Kalian bisa mencoba langsung while loop pada python dengan menjalankan program dibawah ini

```
[ ]  i = 1
     while i < 6:
       print(i)
       i += 1
```

```
1
2
3
4
5
```

untuk lebih lengkapnya mengenai while loop, bisa mengunjungi link berikut.

7 Function pada python

Jika pada paradigma pemrograman sebelumnya (OOP-Java) kalian lebih familiar dengan istilah method, maka pada pemrograman fungsional dengan python kali ini kita akan banyak menggunakan function (di python juga ada method kok, cek disini untuk tau perbedaan keduanya).

Python sendiri memiliki banyak built-in function, diantaranya yang sering kita gunakan, antara lain print() berfungsi untuk mencetak objek, len() berfungsi mengembalikan (return) panjang suatu objek. Sedangkan untuk membuat suatu function, perlu didefinisikan dengan keyword “def”. Untuk lebih memahami mengenai function pada python, anda dapat menjalankan program dibawah ini.

```
[ ]  # deklarasi fungsi tanpa parameter
     def my_function():
       print("Hello from a function")

     my_function()
```

```
Hello from a function
```

```
[ ]  # deklarasi fungsi dengan sebuah parameter
     def my_function(fname):
       print(fname + " Refsnes")

     my_function("Emil")
     my_function("Tobias")
     my_function("Linus")
```

```
Emil Refsnes
Tobias Refsnes
Linus Refsnes
```

```
[ ]  # deklarasi fungsi dengan beberapa parameter
     def my_function(fname, lname):
       print(fname + " " + lname)

     my_function("Emil", "Refsnes")
```

```
Emil Refsnes
```

```
import os

# Inisialisasi database buku yang tersedia
books_available = {
    "11": "Belajar Pemrograman Fungsional",
    "22": "Belajar Pemrograman Web",
    "33": "Belajar Pemrograman Mobile",
}

# Inisialisasi database peminjaman buku
book_borrowed = {}

# Fungsi untuk membersihkan layar
def clear_screen():
    os.system('cls' if os.name == 'nt' else 'clear')

# Fungsi untuk menambahkan buku baru oleh admin
def add_book(book_id, book_title):
    books_available[book_id] = book_title
    print(f"Buku '{book_title}' dengan (ID: {book_id}) telah ditambahkan.")

# Fungsi untuk melakukan peminjaman buku oleh user
def borrow_book(book_id):
    global book_borrowed
    if book_id in books_available:
        if book_id not in book_borrowed:
            book_title = books_available[book_id]
            book_borrowed[book_id] = book_title
            print(f"Anda telah meminjam buku '{book_title}' (ID: {book_id}).")
        else:
            print("Buku ini sudah dipinjam oleh pengguna lain.")
    else:
        print("Buku dengan ID ini tidak tersedia.")

# Fungsi untuk mengembalikan buku oleh user
def return_book(book_id):
    global book_borrowed
    if book_id in book_borrowed:
        book_title = book_borrowed.pop(book_id)
        print(f"Anda telah mengembalikan buku '{book_title}' (ID: {book_id}).")
    else:
        print("Anda tidak memiliki buku ini untuk dikembalikan.")

# Fungsi untuk menampilkan buku yang tersedia
def show_available_books():
    print("\nBuku Tersedia:")
    for book_id, book_title in books_available.items():
        print(f"ID: {book_id} - Judul: {book_title}")


#Menu utama
def main():
    while True:
        clear_screen()
        print("\nSelamat datang di sistem manajemen perpustakaan!")
        print("Pilih jenis akun:")
        print("1. Admin")
        print("2. User")
        print("3. Keluar")
        choice = input("Masukkan pilihan Anda: ")

        if choice == "1":
            clear_screen()
            print("Selamat datang, admin!")
            while True:
                print("\nPilih aksi:")
                print("1. Tambahkan Buku")
                print("2. Keluar sebagai admin")
                sub_choice = input("Masukkan pilihan Anda: ")

                if sub_choice == "1":
                    book_id = input("Masukkan ID buku baru: ")
                    book_title = input("Masukkan judul buku baru: ")
                    add_book(book_id, book_title)
                elif sub_choice == "2":
                    break
                else:
                    print("Pilihan tidak valid.")

        elif choice == "2":
            clear_screen()
            print("Selamat datang, user!")
            while True:
                print("\nPilih aksi:")
                print("1. Pinjam Buku")
                print("2. Kembalikan Buku")
                print("3. Tampilkan Buku Tersedia")
                print("4. Keluar sebagai user")
                sub_choice = input("Masukkan pilihan Anda: ")

                if sub_choice == "1":
                    book_id = input("Masukkan ID buku yang ingin Anda pinjam: ")
                    borrow_book(book_id)
                elif sub_choice == "2":
                    book_id = input("Masukkan ID buku yang ingin Anda kembalikan: ")
                    return_book(book_id)
                elif sub_choice == "3":
                    clear_screen()
                    show_available_books()  # Memanggil clear_screen() di sini akan membuat layar kosong.
                elif sub_choice == "4":
                    break
                else:
                    print("Pilihan tidak valid.")

        elif choice == "3":
            break

        else:
            print("Pilihan tidak valid.")
if __name__ == "__main__":
    main()
```

Mari kita perbaiki dan susun ulang kode fungsi sebelumnya dengan pendekatan fungsional sbb:

```
[ ]  # deklarasi fungsi dengan beberapa parameter
     def my_function(fname, lname):
       # kita cukup menambahkan return value sebagai output dari fungsi
       return (fname + " " + lname)

     full_name = my_function("Emil", "Refsnes")
     print(full_name)
```

```
Emil Refsnes
```

Kalian bisa melakukan hal yang sama untuk fungsi lainnya. Contoh lain dari pemanfaatan fungsi sebagaimana sebuah ekspresi matematika adalah sbb:

```
[ ]  # deklarasi fungsi sebagai sebuah expresi yang mengembalikan sebuah nilai
     def add(x, y):
         return x + y

     result = add(3, 5)
     print(result)
```

```
8
```

Berikut contoh Implementasi pohon ekspresi sebagai fungsi dalam paradigma fungsional:

```
[ ]  # Definisikan pohon ekspresi sebagai fungsi
     def tree(node):
         if type(node) in (int, float):
             return node
         elif type(node) is tuple and len(node) == 3:
             operator, left_operand, right_operand = node
             if operator == '+':
                 return tree(left_operand) + tree(right_operand)
             elif operator == '-':
                 return tree(left_operand) - tree(right_operand)
             elif operator == '*':
                 return tree(left_operand) * tree(right_operand)
             elif operator == '/':
                 return tree(left_operand) / tree(right_operand)

     # Contoh pohon ekspresi: (2 + 3) * (5 - 1)
     expression_tree = ('*', ('+', 2, 3), ('-', 5, 1))

     # Evaluasi pohon ekspresi dengan fungsi pada paradigma fungsional
     result = tree(expression_tree)

     print("Hasil evaluasi pohon ekspresi:", result)
```

```
Hasil evaluasi pohon ekspresi: 20
```

lalu tunjukkan kepada asisten saat pekan materi.

KEGIATAN 1

Melanjutkan materi fungsi tadi, sekarang coba kalian buat dan lengkapi kode berikut untuk fungsi aritmatik lainnya!

```
[ ]  # fungsi pengurangan
     def minus ():

     # fungsi perkalian
     def mult ():

     # fungsi pembagian
     def div ():
```

Sekarang, bisakah kalian kombinasikan fungsi tree dengan fungsi-fungsi aritmatik yang telah kalian buat tadi (add, minus, mult, & div)!!!

Next challenge nya adalah coba modifikasi kode fungsi tree kalian agar dapat mengolah input dengan struktur data yang kita modif juga menjadi seperti ini:

```
[ ]  # Contoh pohon ekspresi: (2 + 3) * (5 - 1)
     expression_tree = ((2, '+', 3), '*', (5, '-', 1))

     # Evaluasi pohon ekspresi dengan fungsi pada paradigma fungsional
     result = tree(expression_tree)

     print("Hasil evaluasi pohon ekspresi:", result)
```

Kalau hasil outputnya sama dengan contoh yang di materi, berarti kalian berhasil. Selamat...

KEGIATAN 2

Diberikan data list campuran sebagai berikut:

```
[ ]  random_list = [105, 3.1, "Hello", 737, "Python", 2.7, "World", 412, 5.5, "AI"]
```

pisahkan antara nilai int, float, dan string dengan ketentuan :

- Data float disimpan dalam bentuk tuple
- Data string disimpan ke dalam list
- Dan data int disimpan dalam dictionary dengan memisahkan angka satuan, puluhan, dan ratusan

KEGIATAN 3

Tambahkan beberapa fungsi dan lengkapi kode di bawah ini sehingga menghasilkan pemrograman yang fungsional lalu tunjukkan kepada asisten saat pekan materi.

```
[ ]  # Sistem Penilaian Akhir Mahasiswa

     # Tambahkan fungsi untuk menghitung nilai akhir

     # Tambahkan fungsi untuk menghitung nilai akhir semua mahasiswa

     def tampilkan_nilai_akhir(data_nilai_akhir):
         print("Hasil Nilai Akhir Mahasiswa:")
         for nama, nilai_akhir in data_nilai_akhir.items():
             print("Nama: {}\tNilai Akhir: {:.2f}".format(nama, nilai_akhir))

     def main():
         data_mahasiswa = {
             # Data mahasiswa (nama sebagai key dan nilai UTS serta UAS sebagai value dalam bentuk dictionary)
         }

         data_nilai_akhir = # Menghitung nilai akhir semua mahasiswa

         tampilkan_nilai_akhir(data_nilai_akhir)

     if __name__ == "__main__":
         main()
```

TUGAS PRAKTIKUM

Buatlah sebuah program tentang sistem informasi, bebas menggunakan google collab ataupun Pycharm atau IDE python lain. Ketentuan program tersebut adalah:

SOAL A (INTERMEDIATE)

1. Terdapat dua akun, admin dan peserta.
2. Akun admin dapat melakukan input data peserta dan edit nilai
3. Terdapat 4 kolom untuk data peserta, yakni ID (untuk mempermudah, id sebaiknya berurut dari 0 dst.), Nama, nilai, hasil akhir
4. Akun peserta bisa menampilkan nilai dan hasil akhir dari peserta itu sendiri
5. Hasil akhir ditentukan jika nilai > =75 maka lolos, nilai < 75 maka tidak lolos
6. Demokan skenario kepada asisten

SOAL B (ADVANCE)

1. Ada dua jenis akun, akun user dan akun admin
2. Akun admin dapat melakukan input buku yang akan dipinjam

3. Akun user dapat melakukan peminjaman buku yang tersedia
4. Buku yang sudah dipinjam tidak dapat dipinjam oleh user lain.
5. Akun user harus dapat mengembalikan buku.
6. Demokan skenario kepada asisten

Pilihlah **salah satu** diantara dua jenis soal lalu buatlah program sekreatif kalian dengan syarat harus sesuai dengan ketentuan diatas dan gunakan semaksimal mungkin materi yang sudah kalian pelajari di modul 1 ini. **Dilarang menggunakan library tambahan (seperti pandas dll.)**.

KRITERIA & DETAIL PENILAIAN TUGAS PRAKTIKUM

Kriteria	Soal A	Soal B	Program identik*
1. Program dapat berfungsi	10	20	0
2. Program menggunakan materi yang sudah dipelajari pada modul serta **mengimplementasikan paradigma pemrograman fungsional**	10	20	0
3. Menjelaskan program yang dibuat dengan **baik dan lancar**	20	20	20*
4. Menjelaskan implementasi materi modul dalam program yang dibuat **beserta alasan penggunaannya**	20	20	20*
5. Menjawab pertanyaan dari asisten dengan **baik dan lancar**	20	20	20*
Total Nilai (maksimal)	80	100	60*

*) Jika program identik dengan praktikan lain. Maka akan ada pengurangan nilai pada kedua praktikan.