**LAPORAN**

**TUGAS BESAR**

**PEMROGRAMAN FUNGSIONAL**



**Data Nilai Matakuliah Bahasa Indonesia**

**Mahasiswa Kelas SIIF-07-O**

**ANGGOTA KELOMPOK:**

Adelia Sannomiya (19102064)

M. Fikri Aqsha Zulfa Ismail (19102136)  
Risang Abdurrahman Gustiwa (19102178)

**KELAS S1 IF-07-O**

**PROGRAM STUDI S1 INFORMATIKA**

**FAKULTAS INFORMATIKA**

**INSTITUT TEKNOLOGI TELKOM PURWOKERTO**

**2021**

**KATA PENGANTAR**

Pertama – tama kami panjatkan puji syukur kepada Allah Subhanawata’ala yang telah melimpahkan rahmat, hidayah serta karunia-Nya kepada penulis sehingga dapat menyelesaikan Tugas Besar Mata Kuliah Pemrograman Fungsional dengan judul Data Nilai Matakuliah Mahasiswa Kelas SIIF-07-O yang merupakan implementasi dari dua studi kasus yaitu studi kasus pertama Higher Order Functions dan kedua The Itertools Module.

Tujuan dari pembuatan laporan Tugas Besar ini adalah sebagai penjelasan serta dokumentasi dari praktikum yang sudah kami lakukan dalam pembuatan program untuk menghitung nilai akhir suatu matakuliah mahasiswa kelas SIIF-07-O.

Terima kasih kami ucapkan kepada Bapak Agus Priyanto selaku dosen Mata Kuliah Pemrograman Fungsional yang selama satu semester ini telah membimbing kami dan memberikan ilmunya, semoga ilmu tersebut dapat menjadi amal jariyah dan dapat bermanfaat bagi pembaca. Yang tidak kalah pentingnya terima kasih kepada rekan – rekan satu kelompok yang sudah berpikir, berdiskusi dan bekerja sama untuk menyelesaikan Tugas Besar ini.

**DAFTAR ISI**

[BAB 1. PENDAHULUAN](#_3znysh7) **4**

[Latar Belakang](#_2et92p0) 4

[Tujuan](#_3dy6vkm) 6

[BAB 2. TINJAUAN PUSTAKA](#_2s8eyo1) **7**

[Higher Order Functions](#_17dp8vu) 7

[Lambda](#_3rdcrjn) 7

[Map](#_26in1rg) 7

[Filter](#_lnxbz9) 8

[Sorted](#_35nkun2) 8

[Functool Module](#_1ksv4uv) 9

[@lru\_cache()](#_hknfi28q4x30) 9

[Partial()](#_34wk8wodupn0) 9

[Reduce()](#_2o9lks1l3pmk) 9

[BAB 3. PEMBAHASAN](#_44sinio) **10**

[Program](#_9du8b1kjnz13) 10

[Cara Kerja](#_2jxsxqh) 12

[BAB 4. PENUTUP](#_1y810tw) **13**

[Kesimpulan](#_4i7ojhp) 13

# BAB 1. PENDAHULUAN

## Latar Belakang

Perkembangan tekonolgi yang sangat pesat mendorong kita untuk selalu kreatif, inovatif dan cepat tanggap akan perubahan teknologi. Gordon E. Moore *co-founder* Intel pada tahun 1965 memprediksi bahwa jumlah transistor pada IC(*Integrated Circuit*) akan berkembang dua kali lipat setiap dua tahun. Observasi dia yang kemudian dikenal dengan nama Moore’s law ini sangat relevan tidak hanya pada bidang elektronika, tetapi juga teknologi secara umum. Salah astu contoh perkembangan teknologi yang sangat pesat adalah komputer, komputer dulunya sebuah mainframe yang bentuknya besar dan butuh ruangan untuk menyimpannya dan tidak semua orang dapat mengakses atau bahkan memilikinya, berbeda dengan komputer sekarang. Sekarang komputer ada dimana-mana ada IoT, Ponsel Pintar, atau rasbery pi yang bisa dibilang chip tetapi memiliki fungsional seperti komputer.

Bahasa pemrograman merupakan komponen penting dalam berkembangan teknologi saat ini. Perkembangan teknologi juga berpengaruh terhadap perkembangan bahasa pemrograman sebagai alat untuk menciptakan sebuah teknologi, berbagai macam bahasa pemrograman juga bermunculan untuk memenuhi kebutuhan industri teknologi bahkan di Indonesia juga memiliki bahasa pemrograman seperti bahasa Nusa. Paradigma dalam bahasa pemrograman juga berkembang, kebanyakan bahasa pemrograman menggunakan beberapa paradigma pemrograman untuk menyediakan pemrogram untuk menggunakan gaya pemrograman yang paling cocok bagi mereka.

Pemrograman fungsional merupakan salah satu paradigma pemrograman yang masih digunakan saat ini. Meskipun pada industri teknologi paradigma pemrograman ini kurang populer dibandingkan paradigma misalnya imperative ataupun object-oriented tetapi beberapa sektor seperti keuangan, perdagangan, komunikasi(*concurrency*), data science, machine learning, dan juga bidang akademik pemrogaman fungsional sangat populer, beberapa projek skala besar juga menggunakan paradigma pemrograman fungsiona.

Python merupakan salah satu bahasa pemrograman yang mendukung paradigma pemrograman fungsional dan cukup banyak digunakan saat ini. Sebuah permasalahan yang mudah diselesaikan dengan mengekspresikan sebuah fungsi merupakan salah satu keunggulan dari paradigma pemrograman fungsional

Maka dari itu pada laporan ini kita akan mengimplementasikan fitur-fitur yang ada di paradigma pemrograman fungsional terutama Python ke permasalahan dengan data *real.* Hal ini bertujuan agar kita mengetahui bagaimana modularitas pada paradigma pemrograman fungsional sangat berguna dan kita mengetahui gambaran bagaimana konsep paradigma pemrograman fungsional berbeda dengan paradigma pemrograman lainnya.

## Tujuan

Tujuan dari dibuatnya program perhitungan nilai mahasiswa

1. Mengetahui nilai rata-rata keseluruhan mahasiswa kelas O.
2. Mengetahui nilai tertinggi dan terendah mahasiswa kelas O.
3. Mencari data nilai mahasiswa dengan cara mengetik nama.
4. Mengetahui nilai setiap mahasiswa kelas O.

# BAB 2. TINJAUAN PUSTAKA

## Higher Order Functions

Higher-order Functions merupakan fungsi yang menggunakan fungsi lain sebagai parameter atau sebagai hasil return. Ada tiga jenis Higher-order Functions, yaitu sebagai berikut :

* Fungsi yang menerima fungsi sebagai salah satu argumennya.
* Fungsi yang mengembalikan fungsi.
* Fungsi yang menerima fungsi dan mengembalikan fungsi.

### Lambda

Salah satu fitur python yang cukup berguna adalah fungsi lambda (anonim). Dikatakan anonim karena fungsi ini tidak diberikan nama pada saat didefinisikan. Pada fungsi biasa kita menggunakan kata kunci *def* untuk membuatnya, sedangkan fungsi anonim ini kita menggunakan kata kunci lambda. Fungsi lambda bisa mempunyai banyak argumen, tapi hanya boleh memiliki satu ekspresi.

Sintaks fungsi lambda :

lambda arguments: expression : lambda x: x\*\*2

### Map

Fungsi map () berfungsi untuk mengaplikasikan satu fungsi ke semua anggota dari iterable (list, tuple dan lain-lain) dan mengembalikan hasilnya berupa objek map. Objek map ini bisa dengan mudah diubah menjadi list baru yang anggotanya berupa hasil pemrosesan dari fungsi dengan menggunakan fungsi list (), tuple () maupun set ().

Fungsi map () mengambil dua buah parameter, yaitu:

* Function – fungsi yang akan memproses setiap item dari iterable.
* Iterable – iterable yang akan diproses (dimapping) menggunakan function.

Sintaks :

map(function, iterable, …)

### Filter

Fungsi filter () berfungsi menyaring anggota – anggota dari iterable menggunakan sebuah fungsi untuk menguji tiap anggota dari iterable tersebut. Hasilnya adalah iterable baru yang anggotanya berasal dari iterable yang lama yang bernilai true saat pengujian.

Fungsi filter () mengambil dua buah parameter, yaitu:

* Function – fungsi untuk menguji elemen-elemen dari iterable apakah True atau False.
* Iterable – iterable yang akan difilter, bisa string, list, tuple dan lain sebagainya.

Sintaks :

filter(function, iterable)

### Sorted

Fungsi sorted () berfungsi untuk mengurutkan suatu iterable baik secara naik maupun turun.

Fungsi sorted () memiliki tiga parameter, yaitu:

* Iterable – sequence (string, tuple, list), koleksi (set, dictionary), maupun sembarang iterator.
* Reverse (opsional) – apakah list diurut terbalik atau tidak (ascending atau descending).
* Key (opsional) – fungsi kunci yang digunakan untuk pembandingan.

Sintaks :

sorted(iterable[, key][, reverse])

## Functool Module

### @lru\_cache()

lru\_cache () adalah dekorator, yang membungkus fungsi dengan [memoizing](https://www.python-course.eu/python3_memoization.php) callable digunakan untuk menyimpan hingga maxsizeyang hasil pemanggilan fungsi dan kembali nilai yang disimpan jika fungsi ini disebut dengan argumen yang sama lagi. Ini dapat menghemat waktu ketika fungsi mahal atau terikat I / O dipanggil secara berkala dengan argumen yang sama.

Contoh utilitas lru\_cache () dapat ditampilkan dalam mengoptimalkan kode yang menghasilkan faktorial angka

### Partial()

Fungsi parsial adalah fungsi turunan yang memiliki beberapa parameter masukan yang telah ditetapkan sebelumnya. Misalnya, jika sebuah fungsi mengambil dua parameter mengatakan " **a"** dan " **b"** , fungsi parsial dapat dibuat darinya yang memiliki " **a"** sebagai argumen yang telah diisi dan kemudian dapat dipanggil dengan " **b"** sebagai satu-satunya parameter . Functool's partial () digunakan untuk membuat fungsi / objek parsial dan ini adalah fitur yang berguna karena memungkinkan untuk:

1. Replikasi fungsi yang sudah ada dengan beberapa argumen sudah diteruskan.
2. Pembuatan versi terbaru dari fungsi yang ada dengan cara yang terdokumentasi dengan baik.

### Reduce()

Fungsi **reduce()** menerima dua argumen, fungsi dan iterable. Ini menerapkan fungsi argumen secara kumulatif ke semua elemen iterable dari kiri ke kanan dan kemudian mengembalikan satu nilai.

Misalnya, reduce () dapat digunakan untuk menghitung jumlah atau produk dari daftar dengan mudah.

Sintaks :

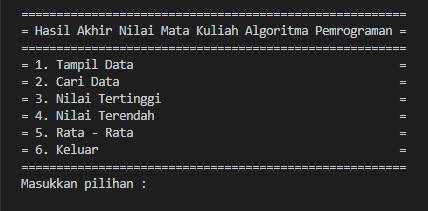
reduce(fungsi, urutan)

# BAB 3. PEMBAHASAN

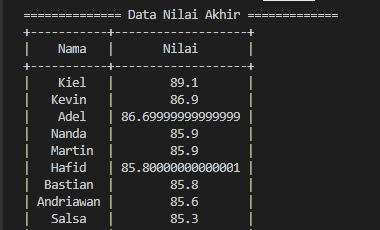
## Program

|  |
| --- |
| 1. ################################################################# 2. # # 3. # TUGAS BESAR IMPLEMENTASI STUDI KASUS : # 4. # 1. Higher Order Function # 5. # > lambda # 6. # > map # 7. # > filter # 8. # > sorted # 9. # 2. Functools Module # 10. # > @lru\_cache() # 11. # > partial() # 12. # > reduce() # 13. # IMPLEMENTASI STUDI KASUS DATA MATA KULIAH KELAS SIIF-07-O # 14. # # 15. ################################################################# 16. import functools 17. import operator 18. import os 19. from prettytable import PrettyTable 20. # element tuple : 21. # element 0 = presensi ----> bobot 10% 22. # element 1 = nilai tugas ---> bobot 30% 23. # element 2 = nilai UTS ---> bobot 25% 24. # element 3 = nilai UAS ----> bobot 35% 25. # data nilai presensi, tugas, uts, uas, nama mahasiswa 26. student\_data = { 27. (100, 70, 77, 90, "Hikmah"), (100, 85, 87, 90, "Kiel"), (100, 75, 80, 72, "Eleven"), (100, 86, 81, 75, "Shodik"), (100, 79, 75, 80, "Satrio"), 28. (90, 85, 88, 75, "Fadh"), (100, 80, 89, 83, "Nanda"), (80, 70, 70, 87, "Wildan"), (100, 80, 77, 94, "Adel"), (100, 87, 78, 87, "Andriawan"), 29. (100, 80, 80, 86, "Faishal"), (100, 71, 77, 86, "Yuri"), (100, 70, 71, 80, "Nafis"), (80, 75, 70, 89, "Lorance"), (85, 80, 72, 77, "Adi"), 30. (100, 80, 73, 80, "Kaka"), (90, 75, 70, 85, "Hersa"), (100, 78, 76, 80, "Attar"), (100, 80, 72, 84, "Vindi"), (100, 85, 73, 72, "M Arif"), 31. (100, 76, 80, 80, "Bagus"), (100, 74, 81, 87, "Fikri"), (100, 74, 83, 80, "Fahri"), (100, 79, 71, 81, "Herwin"), (100, 70, 78, 87, "Imam"), 32. (100, 75, 76, 87, "Pandu"), (100, 78, 90, 70, "Risang"), (100, 80, 80, 86, "Ulinda"), (90, 75, 91, 85, "Salsa"), (95, 75, 92, 79, "Fakhri"), 33. (100, 70, 80, 90, "Putri"), (100, 75, 81, 87, "Ma’arif"), (100, 73, 91, 85, "Martin"), (100, 76, 92, 72, "Galuh"), (100, 74, 81, 80, "Ihza"), 34. (100, 70, 78, 80, "Rangga"), (80, 72, 81, 87, "Nurdin"), (85, 70, 90, 86, "Andika"), (80, 70, 91, 87, "Syifa"), (90, 70, 92, 88, "Bastian"), 35. (100, 70, 90, 87, "Hafid"), (85, 70, 88, 82, "Maldini"), (80, 77, 81, 87, "Aldi"), (100, 76, 80, 80, "Regi"), (100, 76, 87, 80, "Annisa"), 36. (100, 75, 92, 80, "Trias"), (100, 70, 91, 89, "Kevin"), (80, 70, 91, 87, "Atha") 37. } 38. # hitung nilai keseluruhan data 39. total\_grade = sorted(list(map(lambda x: (((x[0] \* 0.10) + (x[1] \* 0.30) + (x[2] \* 0.25) + (x[3] \* 0.35)), x[4]), student\_data)), reverse=True) 40. # jumlahkan semua nilai di total grade 41. jumlah = functools.reduce(operator.add, [x[0] for x in total\_grade]) 42. # clear screen 43. def clear(): 44. os.system('cls' if os.name == 'nt' else 'clear') 45. # tampilkan data total\_grade 46. def show\_data(data\_arg) : 47. data\_table = PrettyTable() 48. data\_table.field\_names = ['Nama', 'Nilai'] 49. print('============== Data Nilai Akhir =============') 50. t\_grade = [[j, round(i, 2)] for i, j in data\_arg] 51. for i in t\_grade: 52. data\_table.add\_rows([i]) 53. print(data\_table) 54. # menu utama 55. def show\_menu(): 56. clear() 57. print("=======================================================") 58. print("= Hasil Akhir Nilai Mata Kuliah Algoritma Pemrograman =") 59. print("=======================================================") 60. print("= 1. Tampil Data =") 61. print("= 2. Cari Data =") 62. print("= 3. Nilai Tertinggi =") 63. print("= 4. Nilai Terendah =") 64. print("= 5. Rata - Rata =") 65. print("= 6. Keluar =") 66. print("=======================================================") 67. pilihan = input("Masukkan pilihan : ") 68. return pilihan 69. # kemnbali ke menu 70. def back\_menu(): 71. print("\n") 72. input("Tekan enter untuk kembali...") 73. entry\_point() 74. # entry point program 75. def entry\_point(): 76. while True: 77. pilihan = show\_menu() 78. if (pilihan == "1"): # pilihan pertama tampilkan seluruh data 79. clear() 80. show\_data(total\_grade) 81. back\_menu() 82. elif (pilihan == "2"): # pilihan kedua cari data berdasarkan nama 83. clear() 84. key = input("Masukkan nama : ") # key pencarian 85. # cari berdasarkan elemen ke-1 86. cari = list(filter(lambda x: x if x[1] == key else False, total\_grade)) 87. show\_data(cari) 88. back\_menu() 89. elif (pilihan == "3"): # cari nilai maksimum data, tampilkan nama dan nilai 90. clear() 91. # cari nilai maksimum berdasarkan elemen ke-0 92. maximum\_val = max(total\_grade, key=lambda x: x[0]) 93. maximum\_val = [maximum\_val] 94. print("=========== nilai tertinggi ============") 95. show\_data(maximum\_val) 96. back\_menu() 97. elif (pilihan == "4"): # cari nilai minimum data, tampilkan nama dan nilai 98. clear() 99. # cari nilai minimum berdasarkan elemen ke-0 100. minimum\_val = min(total\_grade, key=lambda x: x[0]) 101. minimum\_val = [minimum\_val] 102. show\_data(minimum\_val) 103. back\_menu() 104. elif (pilihan == "5"): # cari nilai rata-rata semua data 105. clear() 106. table\_rerata = PrettyTable() 107. table\_rerata.field\_names = ['Rata - rata'] 108. rerata = lambda x, y: x / y # cari rata-rata 109. rerata = round(rerata(jumlah, 48), 2) 110. table\_rerata.add\_row([rerata]) 111. print(table\_rerata) 112. back\_menu() 113. elif(pilihan == "6"): # keluar dari program 114. print("Program keluar") 115. exit() 116. else: 117. print("Pilihan tidak tersedia") # jika pilihan tidak tersedia 118. back\_menu() 119. entry\_point() |

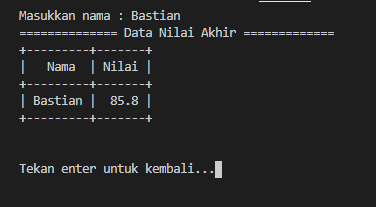
**Gambar 3.1.1** Source code progarm



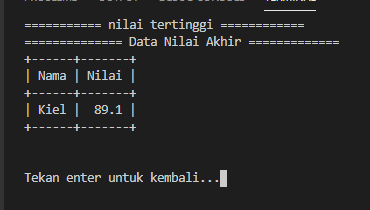
**Gambar 3.1.2** Menu utama program



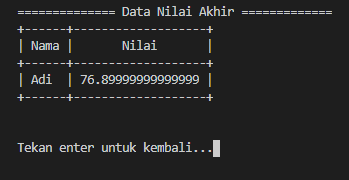
**Gambar 3.1.3** Pillihan nomor satu



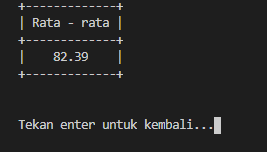
**Gambar 3.1.4** Pilihan nomor dua



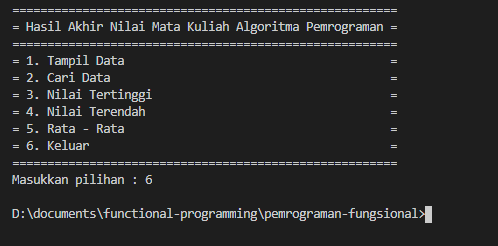
**Gambar 3.1.5** Pilihan nomor tiga



**Gambar 3.1.6** Pilihan nomor empat



**Gambar 3.1.7** Pilihan nomor lima



**Gambar 3.1.8** Pilihan nomor enam

## Cara Kerja

Pada menu program ini terdapat lima pilihan yang dapat dipilih oleh penggunan. Jika memilih nomor satu maka pengguna dapat melihat nilai akhir (nilai tugas, quiz, UTS dan UAS) serta nilai rata - rata mata kuliah algoritma dan pemrograman setiap mahasiswa kelas O. Kemudian tekan enter agar kembali ke menu utama. Jika pengguna memilih nomor kedua maka akan memasukkan kata kunci nama mahasiswa yang akan dicari data nilainya, setelah tekan enter akan muncul nilai akhir mahasiswa beserta namanya. Untuk kembali ke menu utama tekan enter. Apabila pengguna memilih nomor tiga maka akan menampilkan nilai akhir tertinggi mahasiswa beserta namanya. Tekan enter untuk kembali ke menu utama. Apabila pengguna memilih nomor empat maka akan menampilkan nilai akhir terendah mahasiswa beserta namanya. Tekan enter untuk kembali ke menu utama. Jika pengguna memilih nomor lima maka akan menampilkan nilai rata - rata keseluruhan mahasiswa kelas O pada mata kuliah bahasa Indonesia. Tekan enter untuk kembali ke menu utama. Jika pengguna memilih nomor yang tidak tersedia pada menu utama maka akan muncul kata - kata “pilihan tidak tersedia” dan disarankan untuk tekan enter agar kembali bisa memilih nomor yang tersedia.

# BAB 4. PENUTUP

## Kesimpulan

Dalam melakukan manipulasi data dalam python terdapat banyak fungsi yang digunakan untuk menghasilkan output sesuai dengan kebutuhan. Pada program yang kelompok kami buat, untuk higher-order functions menggunakan fungsi max, min, lambda, map, filter dan sorted. Sedangkan untuk the functools module, kelompok kami hanya menggunakan fungsi reduce. Dengan menggunakan fungsi - fungsi tersebut, kita dapat memanipulasi data nilai mahasiswa yang memiliki banyak subnilai seperti nilai tugas, quiz, UTS dan UAS. Untuk mencari nilai tertinggi dan terendah kami menggunakan fungsi max dan min. Untuk menampilkan data nilai mahasiswa secara keseluruhan beserta namanya, kami menggunakan list. Untuk mencari data nilai mahasiswa sesuai dengan keinginan pengguna, terdapat menu pencarian yang dapat digunakan yaitu dengan cara mengetik kata kunci nama mahasiswa yang tersedia. Pada pilihan pencarian kami menggunakan fungsi filter untuk menyaring anggota - anggota iterable dan lambda untuk menulis expressionnya. Untuk menampilkan nilai rata - rata keseluruhan mahasiswa kami menggunakan fungsi lambda untuk expressionnya serta fungsi round untuk menampilkan jumlah digit yang berada di belakang koma.

# DAFTAR PUSTAKA

[1] Priyanto, Agus. Modul 6 - Higher-order Functions (PDF)

[2] Pelleg, Dan. 2004. K-means and KD-trees resources. Dan Pelleg's Home Page. [Online] National Science Foundation, Agustus 13, 2004. [Cited: Januari 3, 2009.] http://www.cs.cmu.edu/~dpelleg/K-means.html.

[3] Priyanto, Agus. Modul 9 - The Functools Module (PDF)

[4] Wijanarko, Andarbeni 2016, PERANCANGAN JARINGAN LOCAL AREA NETWORK ( LAN ) DAN GATEWAY INTERNET PADA GEDUNG TERPADU KABUPATEN PONOROGO MENGGUNAKAN CISCO PACKET TRACER STUDI KASUS DINAS PENDIDIKAN KABUPATEN PONOROGO. Dikutip dari URL <http://eprints.umpo.ac.id/2314/> Akses : 5 Januari 2021, Pukul 20:00