Penerapan High Order Function pada Permasalahan Penentuan Biaya UKT

Cyntia Kristina Sidauruk (122450023)[1], Patricia Leondrea Diajeng Putri (122450050)[2], Berliana Enda Putri (122450065)[3], Feryadi Yulius(122450087)[4], Naufal Fakhri (122450089)[5]

Program Studi Sains Data Institut Teknologi Sumatera
Jl. Terusan Ryacudu, Way Huwi, Kec. Jatiagung, Kabupaten Lampung Selatan,
Lampung 35365

Email:[1]cyntia.122450023@student.itera.ac.id, [2]patricia.122450050@student.itera.ac.id, [3]berliana.122450065@student.itera.ac.id, [4]feryadi.122450087@student.itera.ac.id, [5]naufal.122450089@student.itera.ac.id

ABSTRAK

Metode penentuan biaya UKT baru yang menerapkan konsep High order function dan fungsi dekorator,dengan menggunakan fungsi dekorator untuk menentukan bobot nilai dari setiap kriteria dalam menentukan biaya UKT. Dengan menerapkan parameter yang lebih efisien dari metode yang sudah ada,dimungkinkan untuk mencapai akurasi yang sama atau lebih baik ketika menerapkan fungsi higher order ketika menentukan total biaya UKT yang harus ditanggung mahasiswa, sehingga menghasilkan pengelolaan keuangan yang lebih efisien dan efektif ketika menentukan bobot nilai berdasarkan kriteria-kriteria yang sudah ditentukan,dan memungkinkan sistem secara efektif mengatasi permasalahan dalam pengelolaan keuangan biaya UKT. Eksperimen Ini menunjukkan Efektivitas Dalam mengatasi adaptasi biaya UKT akibat perubahan kebijakan dan kondisi yang terjadi di dalam universitas,sehingga biaya UKT dapat terus disesuaikan dengan kebutuhan mahasiswa dan perubahan kondisi di lingkungan akademik.

Kata kunci : Fungsi dekorator, Higher order function, Pembobotan skor, Penentuan biaya ukt.

PENDAHULUAN

Pendidikan tinggi merupakan investasi yang diandalkan bagi mahasiswa maupun orang tua dan biaya kuliah tunggal(UKT) merupakan faktor penting yang perlu dipertimbang oleh pihak kampus. Untuk melakukan pengelolaan keuangan biaya ukt secara lebih efisien dan efektif maka diperlukan juga pendekatan yang cermat dan terstruktur dalam menetapkan biaya ukt baru. Oleh karena itu, penelitian ini mengusulkan sebuah metode baru yang lebih efektif untuk memanfaatkan konsep fungsi high order function dan fungsi dekorator untuk menentukan biaya ukt bagi mahasiswa.

Metode ini diusulkan dengan menggunakan suatu fungsi dekorator dengan cara memberikan nilai bobot dari setiap kriteria yang telah ditentukan terkait dengan penentuan biaya UKT mahasiswa. Dengan menggunakan suatu kriteria atau parameter yang lebih efisien dan efektif, sehingga metode ini diharapkan dapat memangkas efisiensi waktu dan mencapai tingkat akurasi yang lebih baik dibandingkan menggunakan metode konvensional.Penggunaan High

order Function untuk menentukan total biaya ukt yang harus harus dibayar oleh mahasiswa diharapkan dapat membantu dalam melakukan pengelolaan keuangan yang lebih efektif dan efisien.

Penelitian ini lebih ditekankan untuk mengatasi efektivitas metodologi dalam adaptasi penentuan biaya ukt terhadap perubahan kebijakan dan kondisi yang terjadi di universitas.Sehingga, diharapkan dalam menentukan biaya ukt baru dapat terus disesuaikan dengan kondisi kebutuhan mahasiswa dan kondisi dinamika akademik.

Penelitian ini menunjukan bahwa eksperimen ini dapat memberikan solusi yang responsif terhadap perubahan sehingga universitas dapat terus melakukan pengelolaan biaya keuangan bagi semua pihak yang terlibat dalam menentukan biaya ukt.

Penelitian terbukti memiliki potensi besar dalam meningkatkan proses pada manajemen keuangan di suatu institusi pendidikan tinggi serta meningkatkan partisipasi mahasiswa dalam melakukan pengambilan keputusan mengenai biaya kuliah,oleh karena itu penelitian ini dapat membantu dalam aksesibilitas di perguruan tinggi secara keseluruhan

TUJUAN

1. Menerapkan fungsi High Order Function dan fungsi Dekoraktor
2. Meningkatkan efektifitas dalam pengelolaan keuangan
3. Meningkatkan tingkat akurasi dalam menentukan biaya ukt
4. Mengatasi permasalahan perubahan biaya ukt yang disebabkan suatu kebijakan dan lingkungan akademik

METODE DASAR

Higher Order Functions

Higher Order Functions merupakan fungsi yang beroperasi dalam fungsi lain dan menjadikannya sebagai argumen atau mengembalikan suatu nilai. Secara sederhananya, Higher Order Functions adalah fungsi yang menggunakan suatu fungsi lain sebagai argumen kemudian mengembalikan nilai fungsi tersebut sebagai output. Konsep ini merupakan suatu konsep terpenting pada paradigma pemrograman fungsional. Bahkan, dapat dikatakan bahwa konsep ini mendefinisikan apakah sebuah bahasa pemrograman dapat disebut sebagai fungsional atau tidak.

Higher Order Function memiliki beberapa contoh seperti fungsi **map( )** untuk membuat *array* baru dengan memanggil fungsi *callback* sebagai argumen elemen *array*. Selain itu, ada fungsi **filter ( )** untuk membuat *array* baru dengan mengoperasikan fungsi *callback* untuk setiap anggota dengan memfilter nilai-nilai pada *array* yang telah ada. Serta, fungsi **reduce ( )** untuk menjalankan fungsi *callback* untuk setiap anggota *array* yang dipanggil dengan menghasilkan *single output.*

Fungsi Dekorasi

Dekorator dapat dikatakan sebagai alat yang kuat serta bermanfaat untuk memungkinkan pemrograman mengubah perilaku suatu fungsi atau kelas, seperti menambahkan fungsi logging atau pengaturan waktu, menambah caching, menerapkan pemeriksaan keamanan atau otoritas,

serta memodifikasi nilai kembalian suatu fungsi. Fungsi ini juga dapat digunakan untuk menambahkan ataupun memperluas fungsionalitas ke program yang telah ada. Dekorator dapat digunakan dengan menggabungkan fungsi yang telah ada dengan fungsi lain yang berada di sekitarnya, atau dinamakan fungsi bersarang. Hal tersebut digunakan untuk memperluas perilaku fungsi yang dibungkus tanpa mengubahnya secara permanen.

Fungsi dekorator dapat digunakan dengan cara yang lebih elegan guna mencapai fungsionalitas daripada harus ditetapkan dengan pemanggilan fungsi ke variabel, yaitu dengan menggunakan simbol @. Simbol tersebut kemudian diikut dengan nama dari fungsi dekorator, yang di mana berada tepat sebelum definisi fungsi.

PEMBAHASAN

Dalam mengimplementasikan pola dekorator untuk menghitung skor dan biaya UKT. Perhitungan include up to biaya menghitung skor uang kulia tunggal (UKT) ke dalam fungsi terpisah yang didekorasi oleh calculate_total_cost. Penggunaan parameter score_weight dalam calculate_total_cost memungkinkan penyesuaian biaya menghitung skor uang kulia tunggal (UKT) sesuai kebutuhan. Berikut adalah kode untuk menghitung skor uang kulia tunggal (UKT)

Code

Program ditulis pada bahasa pemrograman Python sebagai berikut:

```python
# Fungsi dekorator untuk menghitung skor
def calculate_score(func):
    def wrapper(*args, **kwargs):
        score = func(*args, **kwargs)
        return score
    return wrapper

# FUngsi dekorator untuk menghitung total keseluruhan
def calculate_total_cost(score_weight):
    def decorator(func):
        def wrapper(*args, **kwargs):
            score = func(*args, **kwargs)
            total_score = sum(score_weight[i] * score[i] for i in
range(len(score)))
            base_cost = 750000
            total_cost = base_cost + total_score * 500000
            return total_cost
        return wrapper
    return decorator

# Fungsi menghitung jumlah tanggung
@calculate_score
```

```
def hitung_skor_tanggungan(jumlah_tanggungan): #menghitung skor tanggungan yang diambil dari jumlah tanggungan
    if jumlah_tanggungan >= 5: #apabila jumlah tanggungan lebih atau sama dengan 5
        return 1 #maka akan diberi skor 1
    else:
        return 5 - jumlah_tanggungan #jika tidak maka akan diberi sekor 5 dan selanjutnya akan dikurangkan dengan jumlah tanggungan

# Fungsi menghitung ketentuan token listrik
@calculate_score
def hitung_skor_listrik(rata_rata_listrik): #menghitung skor listrik yang diambil dari rata-rata listrik
    if rata_rata_listrik > 100000: #apabila rata-rata listik lebih dari 100000
        return 3 #akan diberi skor 3
    elif 50000 <= rata_rata_listrik <= 100000: #apabila rata-rata listik lebih dari sama 500000 dengan atau kurang dari sama dengan 100000
        return 2  #maka akan diberi skor 2
    else:
        return 1 #jika bukan maka akan diberi skor 1s

# Fungsi menghitung ketentuan gaji
@calculate_score
def hitung_skor_gaji(gaji_penanggung_jawab): #menghitung skor gaji yang diambil dari gaji penanggung jawab
    if gaji_penanggung_jawab > 10:   #apabila gaji penanggung jawab lebih dari 10
        return 7 #maka akan diberi skor 7
    elif 8 < gaji_penanggung_jawab <= 10:
        return 6
    elif 6 < gaji_penanggung_jawab <= 8:
        return 5
    elif 4 < gaji_penanggung_jawab <= 6:
        return 4
    elif 3 < gaji_penanggung_jawab <= 4:
        return 3
    else:
        return 2

# Fungsi menghitung status kipk
@calculate_score
def hitung_skor_kip_k(status_kip_k):
    if status_kip_k == "Ya":
        return 1
```

```
    else:
        return 5

# Fungsi menghitung total biaya ukt yang akan didapat
@calculate_total_cost([0.2, 0.3, 0.2, 0.3])
def hitung_biaya_ukt(jumlah_tanggungan, listrik, gaji_penanggung_jawab,
status_kip_k):
    skor_1 = hitung_skor_tanggungan(jumlah_tanggungan)
    skor_2 = hitung_skor_listrik(sum(listrik) / len(listrik))
    skor_3 = hitung_skor_gaji(gaji_penanggung_jawab)
    skor_4 = hitung_skor_kip_k(status_kip_k)
```

Penerapan Pada Contoh Data

```
Python
jumlah_tanggungan = 3
listrik = [120000, 75000, 50000]
gaji_penanggung_jawab = 5.5 # Dalam Juta
status_kip_k = "Tidak"

biaya_ukt = hitung_biaya_ukt(jumlah_tanggungan, listrik, gaji_penanggung_jawab,
status_kip_k)
print("Biaya UKT yang harus dibayarkan: Rp", biaya_ukt)
```

Pada kode penerapan di atas digunakan untuk mengimplementasikan pola dekorator untuk menghitung skor dan biaya UKT. untuk meningkatkan fleksibilitas dan modularitas nya dengan memisahkan perhitungan skor ke dalam fungsi-fungsi terpisah yang didekorasi oleh calculate_score, dan perhitungan incorporate up to biaya UKT ke dalam fungsi terpisah yang didekorasi oleh calculate_total_cost. Penggunaan parameter score_weight dalam calculate_total_cost memungkinkan penyesuaian biaya UKT sesuai kebutuhan. Didapatkan hasil bahwah jumlah tanggungan ada 3, listrik selama tiga bulan berturut-turut 120 ribu 75 ribu, dan 50 ribu, gaji penanggungjawab di angka 5,5 juta rupiah, dan tidak menerima kipk. Kemudian data yang ada dimasukan sebagai argumen pada fungsi menghitung biaya ukt, sehingga didapat kalau hasil biaya ukt yang harus dibayarkan adalah Rp.2.400.000.

KESIMPULAN

Dengan menerapkan pemrograman python dan paradigma fungsional dengan Konsep High Order Function bisa digunakan untuk permasalahan penghitungan biaya UKT dengan sederhana. Program tersebut masih bisa untuk dikembangkan lagi untuk meningkatkan fleksibilitas dan modularitasnya. Implementasi dengan menggunakan fungsi dekorator dengan menerapkan sebuah fungsi untuk menghitung skor berdasarkan kriteria yang ditentukan, menghitung biaya UKT berdasarkan skor yang telah dihitung dan menyesuaikan biaya UKT sesuai kebutuhan. Dengan demikian, program ini menunjukkan efektivitas dalam mengatasi adaptasi biaya UKT yang terjadi akibat perubahan kebijakan dan kondisi yang terjadi, sehingga biaya UKT dapat terus disesuaikan dengan kebutuhan mahasiswa dan perubahan kondisi lingkungan akademik.

REFERENSI

Priambodo, B. 2017. *Higher-Order Function-Paradigma Fungsional Praktis*. *Part 4.*

Romadhona, N.F. 2020. *Python Decorator.* Telematika.ORG

Setiawan, A. 2016. *Higher Order Function di PHP dan Contoh Penerapannya pada array_map*. Codepolitan, Jawa Barat.