Informatika : Fakultas Sains dan Teknologi
Universitas Labuhanbatu
Vol. 12 No. 2 / Mei/2024
2615-1855 (E-ISSN)
2303-2863 (P-ISSN)

Penerapan Algoritma *Dynamic Programming* Dalam Penentuan Prioritas Pengerjaan Tugas Kuliah Mahasiswa

Febrian Nur Hidayat[1], Bryan Desmonda Ferdinan[2], Rendy Saputra[3], Efrans Christian[4], Viktor Handrianus Pranatawijaya[5]

Teknik Informatika, Fakultas Teknik, Universitas Palangka Raya[1,2,3,4,5]

E-mail: febriannurhidayat@mhs.eng.upr.ac.id[1], bryandesmondaferdinan@mhs.eng.upr.ac.id[2], rendysaputra@mhs.eng.upr.ac.id[3], efranschristian2@gmail.com[4], vhpranatawijaya@gmail.com[5]

Corresponding Author: febriannurhidayat@mhs.eng.upr.ac.id

Abstract

College assignments are an essential part of every student's learning process. However, the multitude of tasks from various subjects often leaves students confusedabout prioritizing their work. This research aims to assist students in effectively determining the priority of completing college assignments using a dynamic programming algorithm. This research uses a modeling method of various subjects as variables with different weights, such as deadlines and task difficulty levels. The dynamic programming algorithm is then used to find the optimal solution in completing all tasks, considering various possibilities, both deadlines and difficulty levels. This research has found that the application of the dynamic programming algorithm can create the best solution that takes into account all aspects and produces optimal schedulling in completimg tasks. Althought it requires more time in the calculation process, this algorithm can also provide an optimal solution to help students complete their tasks efficiently and effectively. The application of this dynamic programming algorithm is expected to be a solution for students in completing their tasks better and avoiding stress due to task accumulation. Besides, this research also has the potential to benefit lecturers in assigning tasks to students by considering optimal capabilities and working time. Thus, this research can help improve the quality of learning in higher education.

Keywords: *Academic Tasks, Prioritization, Dynamic Programing Algorithm.*

I. Pendahuluan

Mahasiswa kerap kali diberikan berbagai tugas dari berbagai mata kuliah yang harus diselesaikan dalam waktu yang terbatas. Dalam situasi seperti ini, menentukan skala prioritas penyelesaian tugas menjadi hal yang sangat penting untuk dilakukan. Prioritas yang tepat dapat membantu mahasiswa dalam mengelola waktu dan energi dengan lebih efektif sehingga dapat mencapai hasil yang lebih baik.

Namun, penentuan prioritas ini bukanlah suatu hal yang mudah dilakukan. Mahasiswa perlu melakukan pertimbangan berbagai bobot penugasan seperti tingkat kesulitan tugas, tenggat waktu (deadline), dan yang lainnya. Sehingga, metode yang dapat mendukung mahasiswa dalam hal ini sangat dibutuhkan untuk menentukan prioritas dalam menyelesaikan tugas.

Metode yang bisa diterapkan adalah algoritma *dynamic programming*. Algoritma ini merupakan strategi yang memecahkan suatu masalah dengan mendekomposisikannya menjadi sub masalah yang lebih sederhana, lalu menggunakan solusi dari submasalah tersebut untuk merumuskan solusi untuk masalah yang lebih kompleks. Dalam konteks penentuan urutan tugas, algoritma *dynamic programming* bisa diaplikasikan untuk menetapkan urutan penyelesaian tugas yang optimal dengan memperhitungkan semua bobot relevan.

Harapan dari penelitian ini yaitu untuk memberikan jawaban atas pertanyaan-pertanyaan berikut:

1. Bagaimana solusi optimal pada algoritma *dynamic programming* dipengaruhi oleh variasi bobot (tingkat kesulitan dan tenggat waktu)?
2. Bagaimana cara menggunakan algoritma *dynamic programming* untuk menetapkan prioritas tugas kuliah secara efisien?
3. Bagaimana penerapan algoritma *dynamic programming* dalam menentukan prioritas tugas kuliah?

Penelitian ini bertujuan untuk merancang model penjadwalan tugas mahasiswa yang optimal, memberikan bantuan kepada mahasiswa dalam menetapkan urutan penyelesaian tugas kuliah secara efisien dengan menggunakan algoritma *dynamic programming*, dan juga untuk menerapkan algoritma *dynamic programming* dalam konteks penjadwalan tugas.

II. Landasan Teori

Program dinamis, juga dikenal sebagai *Dynamic Programming*, merupakan teknik pemecahan masalah yang memecah solusi menjadi serangkaian tahapan atau *stage*. Setiap tahapan menghasilkan solusi yang optimal dan hasil dari keputusan yang terkait satu sama lain. Tahapan ini mewakili sub-masalah yang dipecahkan secara berurutan (M.N. Zein *et.al.*, 2021).

Teknik yang digunakan dalam algoritma *dynamic programming* yaitu membagi suatu persoalan menjadi beberapa bagian (tahap), yang kemudian tiap-tiap tahap tersebut dipecah dengan pengoptimalan keputusan hingga seluruh persoalan yang ada dapat dipecahkan (Insidini Fawwaz, 2019).

Pendekatan pada algoritma *dynamic programming* berdasarkan pada prinsip optimalitas yang dinyatakan oleh Richard Bellman, yaitu "Suatu kebijakan optimal mempunyai sifat bahwa apapun keadaan dan keputusan awal, keputusan berikutnya harus membentuk suatu kebijakan optimal dengan memperhatikan keadaan dari hasil keputusan pertama" (Nurma Indah Sari, 2020).

Program dinamis dan algoritma *greedy* sama-sama mencari solusi yang maksimal dan optimal. Namun, algoritma *greedy* hanya fokus pada optimasi lokal, yang bisa berarti hasilnya tidak selalu optimal. Sebaliknya, program dinamis menggunakan hasil dari setiap sub-masalah untuk mempertimbangkan dan menyelesaikan sub-masalah yang lebih kompleks (Ainun Fitryh Vianiryzki, 2019).

Ada dua jenis metode penyelesaian dalam algoritma *dynamic programming*, yaitu perhitungan rekursif maju dan perhitungan rekursif mundur. Perhitungan rekursif maju

Informatika : Fakultas Sains dan Teknologi
Universitas Labuhanbatu
Vol. 12 No. 2 / Mei/2024
2615-1855 (E-ISSN)
2303-2863 (P-ISSN)

berlangsung dari iterasi pertama hingga iterasi terakhir, sementara perhitungan rekursif mundur berlangsung sebaliknya, dimulai dari iterasi terakhir dan bergerak mundur ke iterasi pertama (Devita, R. N., & Wibawa, A. P., 2020).

Prinsip kunci dalam penyelesaian masalah dengan algoritma *dynamic programming* adalah prinsip optimalitas. Prinsip ini mendefinisikan solusi masalah tidak sebagai solusi tunggal, tetapi sebagai serangkaian tahapan (Vicky Lises Pasepti, 2021).

Dynamic programming, berdasarkan pendekatannya, terbagi menjadi dua jenis, yaitu *top-down* dan *bottom-up* (Rifqi Naufal Abdjul, 2021). Berikut ini adalah penjelasan tentang pembagiannya.

1. ***Top-down***
   Pendekatan *top-down* berarti melihat masalah dari perspektif solusi (dari atas) dan bergerak menuju tahapan-tahapan (ke bawah) untuk menyelesaikan masalah tersebut. Misalnya, jika kita berambisi untuk meraih kesuksesan dan kekayaan, maka kita harus mendapatkan pekerjaan yang baik. Untuk itu, kita perlu menyelesaikan studi dengan hasil yang memuaskan, yang berarti kita harus belajar dengan sungguh-sungguh. Teknik yang biasanya diterapkan dalam pendekatan top-down sering disebut sebagai memoization, yang berarti proses penyimpanan data dari langkah-langkah sebelumnya yang berpotensi digunakan dalam langkah-langkah selanjutnya.
2. ***Bottom-up***
   Pendekatan *bottom-up* berarti melihat masalah dari aspek-aspek kecilnya (bawah) dan bergerak menuju aspek yang lebih besar (atas). Sebagai contoh, misalkan kita mulai dengan belajar, yang akan membantu kita mendapatkan nilai baik, yang kemudian memungkinkan kita lulus kuliah dengan prestasi baik, yang kemudian akan membuka jalan untuk mendapatkan pekerjaan yang baik, dan pada akhirnya, kita akan meraih kesuksesan dan kekayaan. Teknik yang biasa digunakan dalam pendekatan *bottom-up* sering disebut sebagai tabulasi, yang berarti proses penyimpanan setiap kondisi dalam sebuah tabel untuk memudahkan akses di masa yang akan datang.

Dalam konteks program dinamis, pendekatan ini didasarkan pada prinsip optimalitas. Artinya, sebuah kebijakan optimal memiliki karakteristik di mana, tidak masalah apa kondisi dan keputusan awalnya, keputusan berikutnya harus membentuk strategi optimal dengan mempertimbangkan kondisi yang muncul dari keputusan pertama (Taufan Adhi Putra *et.al., 2021*).

Metode *dynamic programming* adalah pendekatan yang sering digunakan dan telah terbukti berguna di berbagai area dalam Operasi Riset (RO) (Muhammad Abdurrahman Rois, 2019).

Mirip dengan algoritma *divide-and-conquer*, algoritma *dynamic programming* memecahkan masalah dengan menggabungkan solusi dari submasalah-submasalah yang ada. (Naufal Dean Anugrah, 2019).

Informatika : Fakultas Sains dan Teknologi
Universitas Labuhanbatu
Vol. 12 No. 2 / Mei/2024
2615-1855 (E-ISSN)
2303-2863 (P-ISSN)

III. Metode Penelitian

Metode penelitian yang diterapkan kali ini masuk dalam kategori penelitian kuantitatif, yaitu pengembangan dengan berbagai model sistematis, berbagai teori, yang berkaitan dengan kasus yang sedang terjadi. Solusi terhadap kasus tersebut diperoleh dengan melalui suatu proses pengukuran. Pengumpulan data biasanya dilakukan dengan menggunakan suatu instrumen penelitian maupun analisis data bersifat statistik. Pada penelitian ini, metode penelitian kuantitatif yang dilakukan bersifat eksperimen. Penyajian data dilakukan melalui percobaan terencana (memberikan permisalan nilai variabel/parameter). Data yang diharapkan akan diproses dengan metode eksperimental untuk mengevaluasi pengaruh variabel independen, yang berfungsi sebagai perlakuan atau intervensi, terhadap hasil penelitian. Berdasarkan penelitian ini dilakukan model algoritma *Dynamic Programming*.

Dynamic programming merupakan algoritma pendekatan untuk memecahkan suatu permasalahan menjadi masalah-masalah yang lebih sederhana dan menjadikannya sebagai pondasi untuk membangun solusi. Dalam permasalahan untuk menentukan prioritas tugas mata kuliah mahasiswa, algoritma *dynamic programming* dapat diterapkan untuk menentukan nilai optimal dari seluruh parameter/variabel yang ditentukan sebagai bentuk perlakuan untuk menciptakan urutan yang optimal dalam penyelesaian tugas dengan mempertimbangkan semua bobot yang relevan.

Untuk menentukan solusi dengan algoritma *dynamic programming* pada permasalahan prioritas tugas mata kuliah, dapat dilakukan langkah-langkah berikut:

1. Menentukan Data Tugas Mata Kuliah.

   Setiap tugas mata kuliah diberikan permisalan nilai bobot dan tenggat waktu.
2. Membuat Tabel Penelitian.

   Mencatat setiap tugas mata kuliah beserta bobot dan tenggat waktu yang diperkirakan dalam tabel penelitian.
3. Menentukan Prioritas Tugas.

   Memasukan variabel/parameter tugas mata kuliah ke dalam pendekatan algoritma *dynamic programming*, melalui rumus yang telah ditentukan.

Tahapan penelitian dimulai dengan analisa data tugas mata kuliah secara keseluruhan yaitu bobot dan tenggat waktu. Hasil penelitian ditentukan menggunakan rumus nilai optimal dengan pendekatan algoritma *dynamic programming*, dan disajikan dalam *output* program berbahasa Python yang terdapat pada bagian 4 Hasil dan Pembahasan.

Secara garis besar rumus nilai optimal dengan pendekatan algoritma *dynamic programming* untuk mendapatkan solusi permasalahan pada bagian ini yaitu sebagai berikut:

$$Nilai\ Optimal\ (Nilai\ Per\ Hari) = \frac{Bobot}{Tenggat\ Waktu}$$

Untuk mempertimbangkan kedua aspek (bobot dan tenggat waktu) tugas mata kuliah, membagi bobot dengan tenggat waktu, merupakan bentuk pendekatan untuk mendapatkan "nilai per hari" terhadap setiap tugas. Artinya, nilai optimal yang ditentukan merupakan solusi terhadap pengerjaan tugas per hari. Dengan rumus ini, tugas dengan nilai per hari tertinggi menjadi prioritas.

Informatika : Fakultas Sains dan Teknologi
Universitas Labuhanbatu
Vol. 12 No. 2 / Mei/2024
2615-1855 (E-ISSN)
2303-2863 (P-ISSN)

Rumus tersebut merupakan salah satu pendekatan yang sering digunakan dalam penjadwalan dan teori antrian, yang dikenal sebagai heuristik nilai per unit atau heuristik rasio. Dasar rumus ini berusaha untuk mendapatkan "solusi terbaik" dari setiap unit waktu yang digunakan untuk mengerjakan tugas. Dengan membagi bobot dengan tenggat waktu, maka pengerjaan tugas dilakukan dengan ukuran seberapa banyak "nilai" yang didapatkan per unit waktu. Rumus dirancang untuk memaksimalkan "nilai" yang diperoleh per unit waktu. Artinya, "nilai terbaik" berfokus pada kesulitan atau bobot tugas.

IV. Hasil Dan Pembahasan

Framework Penelitian

Untuk *framework* penelitian ini yaitu seperti pada gambar 1 berikut.

Gambar 1. *Framework* Penelitian

Data Penelitian

Untuk mencari prioritas pengerjaan tugas kuliah mahasiswa, maka diperlukan data penelitian berupa data mana kuliah beserta bobot kesulitan tugas dan tenggat waktu tugas seperti pada tabel 1 berikut.

Tabel 1. Data penelitian

Tugas ke-	Mata kuliah	Bobot kesulitan tugas (skala 1-5)	Tenggat waktu (dalam hari)
1	Algoritma dan Pemrograman II	5	3
2	Arsitektur dan Organisasi Komputer	4	4
3	Kalkulus II	4	7
4	Metode Numerik	1	3
5	Sistem Operasi	3	6
6	Struktur Data	4	3

Analisis Penentuan Prioritas Pengerjaan Tugas Kuliah Mahasiswa Dengan Algoritma *Dynamic Programming*

Setelah menghimpun data tugas kuliah mahasiswa, kemudian akan dilakukan proses penentuan prioritas pengerjaan tugas kuliah mahasiswa dengan menggunakan algoritma *dynamic programming* berdasarkan data yang telah diperoleh. Dalam hal ini, algoritma *dynamic programming* akan diimplementasikan ke dalam *pseudocode*. Isi dari *pseudocode* tersebut yaitu seperti berikut.

```
Data Structure CourseAssignment:
    FUNCTION __init__(self, name, deadline, difficulty)
    # Inisiasi objek CourseAssignment dengan atribut nama, tenggat waktu, dan kesulitan

DEKLARASI
name, deadline, difficulty: string or integer

ALGORITMA
SET self.name to name
SET self.deadline to deadline
SET self.difficulty to difficulty
-----------------------------------------------
# Fungsi untuk meminta pengguna memasukkan tugas-tugas baru
FUNCTION get_assignments() --> assignments: list of CourseAssignment objects

DEKLARASI
(tidak ada)
```

Informatika : Fakultas Sains dan Teknologi
Universitas Labuhanbatu
Vol. 12 No. 2 / Mei/2024
2615-1855 (E-ISSN)
2303-2863 (P-ISSN)

```
ALGORITMA
assignments ← empty list
REPEAT
    name ← INPUT("Masukkan Mata Kuliah: ")
    IF name is empty THEN
        PRINT "Mata Kuliah tidak boleh kosong. Silakan coba lagi!"
        CONTINUE
    END IF

deadline←CONVERT_TO_INTEGER(INPUT("Masukkan Tingkat Tenggat Waktu (1-10): "))
    difficulty←CONVERT_TO_INTEGER(INPUT("Masukkan Tingkat Kesulitan (1-10): "))
    APPEND CourseAssignment(name, deadline, difficulty) to assignments
    PRINT "Data Berhasil Dimasukkan!"

    REPEAT
        repeat ← INPUT("Ulangi Menu (y/t): ").strip().lower()
        IF repeat is 'y' THEN
            EXIT REPEAT
        ELSE IF repeat is 't' THEN
            RETURN assignments
        ELSE
            PRINT "Pilihan tidak valid. Silakan coba lagi."
        END IF
    UNTIL FALSE
UNTIL FALSE
```

```
UNTIL FALSE
-----------------------------------------
# Fungsi untuk mencetak daftar tugas-tugas
FUNCTION print_assignments(assignments: list of CourseAssignment objects)

DECLARE
(tidak ada)
ALGORITMA
FOR i ← 1c to LENGTH(assignments) DO
    PRINT i + ". " + assignments[i].name + " (Deadline: " + assignments[i].deadline + ", Difficulty: " + assignments[i].difficulty + ")"
END FOR
-----------------------------------------
# Fungsi untuk menghitung prioritas tugas berdasarkan kesulitan dan tenggat waktu
FUNCTION
calculate_priority(assignment: CourseAssignment) --> tuple (string, integer)

DECLARE
(tidak ada)

ALGORITMA
SET bobot_tugas to assignment.difficulty
    SET deadline to assignment.deadline
    SET nilai_optimal to bobot_tugas - (bobot_tugas / deadline)
    RETURN (format(nilai_optimal, ".1f"), bobot_tugas)
-----------------------------------------
# Fungsi untuk mengurutkan dan mencetak daftar tugas-tugas berdasarkan prioritas nilai optimal
FUNCTION prioritize_assignments(assignments: list of CourseAssignment)
```

```
ALGORITMA
//SORT assignments using calculate_priority in reverse order
PRINT "PRIORITAS NILAI OPTIMAL:"
FOR i ← 0c to LENGTH(assignments) - 1 DO
    SET nilai_optimal, bobot_tugas to calculate_priority(assignments[i])
    PRINT i + 1 + ". " + assignments[i].name + " (Nilai Optimal: " + nilai_optimal + ")"
END FOR
-----------------------------------------
# Program utama untuk mengatur interaksi dengan pengguna
PROGRAM main()

DECLARASI
(tidak ada)

ALGORITMA
REPEAT

    opsion ← INPUT("Masukkan pilihanmu (1-4): ")
    SWITCH option
    CASE '1':
    assignments ← get_assignments()
    CASE '2':
    IF LENGTH(assignments) == 0 THEN
    PRINT "Belum ada data yang dimasukkan. Silakan masukkan data terlebih dahulu."
    ELSE
    print_assignments(assignments)
    END IF
    CASE '3':
    IF LENGTH(assignments) == 0 THEN
    PRINT "Belum ada data yang dimasukkan. Silakan masukkan data terlebih dahulu."
    ELSE
        prioritize_assignments(assignments)
    END IF
    CASE '4':
    PRINT "Program Selesai!"
    EXIT REPEAT
    OTHERWISE
    PRINT "Pilihan tidak valid. Silakan coba lagi."
    INPUT("Tekan enter untuk melanjutkan...")
        END SWITCH
    UNTIL FALSE

IF __name__ == '__main__':
    CALL main()
```

Peran utama *pseudocode* tersebut terletak pada fungsi "*calculate_priority*", yang digunakan untuk memperhitungkan prioritas tugas berdasarkan bobot tingkat kesulitan dan tenggat waktu, sehingga diperoleh nilai optimal. Nilai optimal diperoleh dengan rumus "bobot_tugas - (bobot_tugas / deadline)", mengembalikan nilai optimal dan bobot tugas. Fungsi "*__init_*" bertujuan untuk menginisiasi objek "*CourseAssignment*" yang berguna sebagai struktur data yang menyimpan informasi terkait mata kuliah dan tugasnya, berupa atribut 'nama', 'tenggat waktu', 'tingkat kesulitan'. Fungsi "*__It__*" berguna untuk membandingkan dua objek/bobot pada "*CourseAssignment*", yaitu tenggat waktu dan tingkat kesulitan.

Hasil Penentuan Prioritas Pengerjaan Tugas Kuliah Mahasiswa Dengan Algoritma *Dynamic Programming*

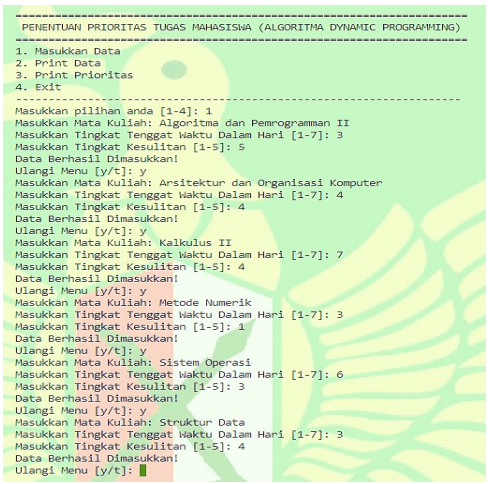
Penerapan algoritma *dynamic programming* untuk menentukan prioritas pengerjaan tugas kuliah mahasiswa diterapkan dengan menggunakan bahasa pemrograman Python yang bersumber dari *pseudocode* yang telah dibuat sebelumnya. Program akan menampilkan hasil perhitungan nilai optimal per tugas mata kuliah, sehingga solusi untuk prioritas pengerjaan tugas kuliah tersebut. Berikut adalah penerapan algoritma *dynamic programming* pada pemrograman.

```
=================================================================
 PENENTUAN PRIORITAS TUGAS MAHASISWA (ALGORITMA DYNAMIC PROGRAMMING)
=================================================================
1. Masukkan Data
2. Print Data
3. Print Prioritas
4. Exit
-----------------------------------------------------------------
Masukkan pilihan anda [1-4]: 1
Masukkan Mata Kuliah: Algoritma dan Pemrogramman II
Masukkan Tingkat Tenggat Waktu Dalam Hari [1-7]: 3
Masukkan Tingkat Kesulitan [1-5]: 5
Data Berhasil Dimasukkan!
Ulangi Menu [y/t]: y
Masukkan Mata Kuliah: Arsitektur dan Organisasi Komputer
Masukkan Tingkat Tenggat Waktu Dalam Hari [1-7]: 4
Masukkan Tingkat Kesulitan [1-5]: 4
Data Berhasil Dimasukkan!
Ulangi Menu [y/t]: y
Masukkan Mata Kuliah: Kalkulus II
Masukkan Tingkat Tenggat Waktu Dalam Hari [1-7]: 7
Masukkan Tingkat Kesulitan [1-5]: 4
Data Berhasil Dimasukkan!
Ulangi Menu [y/t]: y
Masukkan Mata Kuliah: Metode Numerik
Masukkan Tingkat Tenggat Waktu Dalam Hari [1-7]: 3
Masukkan Tingkat Kesulitan [1-5]: 1
Data Berhasil Dimasukkan!
Ulangi Menu [y/t]: y
Masukkan Mata Kuliah: Sistem Operasi
Masukkan Tingkat Tenggat Waktu Dalam Hari [1-7]: 6
Masukkan Tingkat Kesulitan [1-5]: 3
Data Berhasil Dimasukkan!
Ulangi Menu [y/t]: y
Masukkan Mata Kuliah: Struktur Data
Masukkan Tingkat Tenggat Waktu Dalam Hari [1-7]: 3
Masukkan Tingkat Kesulitan [1-5]: 4
Data Berhasil Dimasukkan!
Ulangi Menu [y/t]:
```

Gambar 2. Halaman Menu Utama

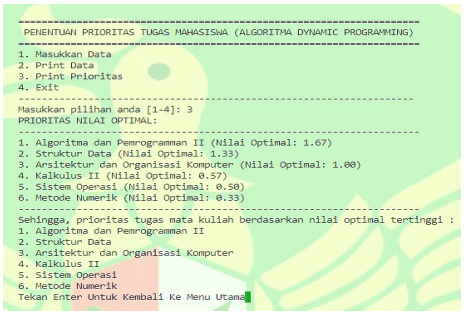
Gambar diatas merupakan halaman awal atau menu utama dari program yang menerapkan algoritma *dynamic programming*, dimana program meminta pengguna untuk memasukkan menu yang tersedia.

Informatika : Fakultas Sains dan Teknologi
Universitas Labuhanbatu
Vol. 12 No. 2 / Mei/2024
2615-1855 (E-ISSN)
2303-2863 (P-ISSN)



Gambar 3. Menu *Input* Data

Jika pengguna memilih menu pertama, maka program akan meminta pengguna untuk memasukkan data yang terdiri dari nama mata kuliah, tenggat waktu, dan tingkat kesulitan tugas. Selanjutnya, program akan meminta pengguna memasukkan pilihan. Jika ingin memasukkan lebih dari satu data, maka pengguna bisa mengulangi menu tersebut, namun jika data dirasa sudah, maka pengguna bisa keluar dari menu tersebut.



Gambar 4. Menu *Print* Data

Selanjutnya, pengguna bisa menampilkan data yang telah dimasukkan ke program dengan memilih menu kedua, sehingga program akan menampilkan seluruh data yang telah dimasukkan oleh pengguna.

Gambar 5. Menu *print* hasil

Terakhir, pengguna bisa memilih menu ketiga, sehingga program akan menampilkan hasil perhitungan nilai optimal berdasarkan data tugas yang dimasukkan serta menampilkan prioritas pengerjaan tugas kuliah mahasiswa berdasarkan nilai optimal tertinggi. Berdasarkan hasil penentuan prioritas pengerjaan tugas oleh algoritma *dynamic programming*, maka ditemukan urutan penyelesaian tugas dari mahasiswa, yaitu:

1. Tugas pertama yang diselesaikan yaitu tugas “Algoritma dan Pemrograman II”, karena memiliki nilai optimal tertinggi, yaitu 1,67.
2. Tugas kedua yang diselesaikan yaitu tugas “Struktur Data”, karena memiliki nilai optimal urutan kedua, yaitu 1,33.
3. Tugas ketiga yang diselesaikan yaitu tugas “Arsitektur dan Organisasi Komputer”, karena memiliki nilai optimal urutan ketiga, yaitu 1,00.
4. Tugas keempat yang diselesaikan yaitu tugas “Kalkulus II”, karena memiliki nilai optimal urutan keempat, yaitu 0,57.
5. Tugas kelima yang diselesaikan yaitu tugas “Sistem Operasi”,

karena memiliki nilai optimal urutan kelima, yaitu 0,50.

6. Tugas terakhir yang diselesaikan yaitu tugas "Metode Numerik", karena memiliki nilai optimal urutan terendah, yaitu 0,33.

V. Kesimpulan Dan Saran

Kesimpulan

Algoritma *dynamic programming* efektif dalam menentukan prioritas tugas mata kuliah berdasarkan bobot dan tenggat waktu. Tabel penelitian memungkinkan pengorganisasian data secara sistematis, dan rumus nilai optimal memfasilitasi penentuan prioritas yang efisien. Hasil dari penelitian ini menunjukkan bahwa pendekatan yang digunakan mampu menghasilkan solusi yang optimal untuk mengatur tugas-tugas mata kuliah.

Saran

Program yang telah dibuat masih memiliki banyak ruang untuk peningkatan karena terbatasnya waktu, sehingga perlu ada pengembangan lebih lanjut di masa mendatang, seperti:

1. Perlu ditambahkan fitur menu untuk mengedit dan menghapus data tugas mata kuliah yang diinginkan. Sehingga tidak harus mengulangi program dari awal.
2. Perlu ditambahkan limitasi dalam program, seperti batas waktu pengerjaan tugas dalam sehari. Agar solusi yang diberikan merupakan pertimbangan faktor-faktor lainnya (selain bobot tugas dan tenggat waktu).
3. Untuk penggunaan yang lebih efisien, perlu dikembangkan secara luas seperti dalam *website*.

VI. Daftar Pustaka

Ainun Fitryh Vianiryzki, 2019. *Implementasi Algoritma Dynamic Programming Dalam Menyelesaikan Airport Gate Assignment Problem.* Bandung, Indonesia

Devita, R. N., & Wibawa, A. P. 2020. *Teknik-Teknik Optimasi Knapsack Problem. Sains, Aplikasi, Komputasi dan Teknologi Informasi Vol 2, No 1, 35-40.*

Insidini Fawwaz *et al.,*2019. Penerapan Algoritma Dynamic Programming Pada Pergerakan Lawan Dalam Permainan Police & Thief. *Journal Of Informatics And Telecommunication Engineering.*

Muhammad Abdurrahman Rois, 2019. *Penyelesaian Integer Knapsack Problem Menggunakan Eksplorasi Algoritma Greedy, Dynamic Programming, Brute Force Dan Genetic*. Skripsi, Fakultas Sains Dan Teknologi, Univeristas Islam Negeri Walisongo Semarang.

M.N. Zein *et al.*, 2022. Penerapan Program Dinamis Untuk Menentukan Jalur Yang Optimum Dalam Pengiriman Benih Ikan Ceps Aquarium. *Bulletin of Applied Industrial Engineering Theory* Vol. 3 No. 1 Maret 2022

Naufal Dean Anugrah, 2019. *Implementasi Multiple Constraint Knapsack Problem dengan Algoritma Dynamic Programming dalam Optimasi Nilai Transfer Pemain*

Olahraga. Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.

Nurma Indah Sari, 2020. *Penentuan Rute Terpendek Pendistribusian Produk Kue Dengan Menggunakan Algoritma Dynamic Programming Pada Pabrik Kue Ima Brownies*. Skripsi, Program Studi Matematika, Fakultas Sains Dan Teknologi, Universitas Islam Negeri Sumatera Utara Medan.

Rifqi Naufal Abdjul, 2021. *Penerapan Dynamic Programming Dalam Optimisasi Pesanan Pada Platform Pemesanan Makanan Online*. Program Studi Teknik Informatika Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.

Taufan Adhi Putra *et al*., 2020. Optimasi Jumlah Operator Alat Berat Pada Pekerjaan Struktur Proyek Apartemen X Menggunakan Dynamic Programming Method. *JOS-MRK Jurnal Online Skripsi Menajemen Rekayasa Kontruksi Polinema Volume 1, Nomor 1, Juni 2020*

Vicky Lises Pasepti, 2021. *Implementasi Algoritma Dynamic Programming Dalam Menyelesaikan Permasalahan Integer Knapsack.* Tugas Akhir, Fakultas Sains Dan Teknologi, Universitas Islam Negeri Sultan Syarif Kasim Riau Pekan Baru.