Jurnal Sistem Komputer dan Kecerdasan Buatan
Volume VII Nomor 1 September 2023

Simulator Mesin *Deterministic Finite Automata* (DFA) Berdasarkan Diagram Transisi Menggunakan *Python*

Muhammad Faris Nabhan[1], Dhieka Avrilia Lantana[2], Pandu Zidni[3], Afif Arofi[4], Perwira Habibbullah[5], Albie Frazza Mukti[6], Muhammad Iqbal Khadavi[7]

[1,3,4,5,6,7]Program Studi Informatika, [2]Program Studi Bisnis Digital, Jalan Sawo Manila No. 61 Jakarta Selatan
muhammadfarisnabhan2021@student.unas.ac.id, dhiekalantana@civitas.unas.ac.id

Diterima : 01 September 2023
Disetujui : 28 September 2023

***Abstract*—Mesin *Deterministic Finite Automata* (DFA) adalah model matematika yang memiliki peran penting dalam ilmu komputer, terutama dalam bidang bahasa formal dan teori automata. DFA digunakan untuk menganalisis, memodelkan, dan memahami bahasa formal serta perilaku mesin yang menerima atau menolak string berdasarkan aturan yang ditentukan. Penelitian ini bertujuan untuk membangun simulator DFA yang dapat digunakan untuk membuat mesin DFA dan menguji apakah string masukan dapat diterima oleh mesin DFA atau tidak. Dalam proses pembuatan mesin DFA menggunakan simulator ini, pengguna diminta untuk memberikan informasi berupa jumlah state, tabel transisi, keadaan akhir dan input masukan. Simulator ini akan memeriksa apakah string masukan dapat mencapai keadaan akhir dalam mesin DFA yang telah dibuat berdasarkan informasi yang diberikan oleh pengguna. Hasil penelitian ini dapat memberikan kontribusi dalam pengembangan dan pemahaman DFA serta membantu dalam verifikasi dan validasi string masukan pada mesin DFA.**

***Keywords* — simulator, fsa, dfa, automata, string, python**

I. PENDAHULUAN

Automata, dalam konteks ilmu komputer dan teori automata, merujuk pada model matematika yang digunakan untuk menganalisis dan memodelkan perilaku mesin komputasi. Automata dapat digunakan untuk mempelajari dan memahami bahasa formal, mengenali pola dalam string atau rangkaian input, serta memahami kompleksitas komputasi [1]. Suatu automata dapat memiliki fungsi dasar seperti dapat membaca input berupa string. Input dalam mesin otomata dianggap sebagai bahasa yang harus dikenali oleh mesin. Kemudian mesin automata akan mengecek apakah inputan tersebut dikenali (diterima) atau ditolak [2]. Secara umum, automata terdiri dari sejumlah keadaan, simbol input, aturan transisi, dan keadaan awal serta keadaan akhir.

Salah satu jenis automata dalam konteks teori automata adalah *Finite State Automata* (FSA). Finite State Automata (FSA), bukan merupakan suatu mesin fisik, melainkan suatu model matematika dari suatu sistem yang menerima input dan menghasilkan output diskrit [3]. FSA merupakan mesin automata dari bahasa reguler. FSA yang memiliki tepat satu state berikutnya untuk setiap simbol masukan disebut dengan *Deterministic Finites State Automata* (DFA) [4].

FSA banyak digunakan untuk pemodelan pada mesin seperti pemodelan pada *vending machine* [5]–[8]. Selain itu, FSA juga diterapkan pada sistem seperti sistem untuk pendaftaran kelas kursus [9], sistem pencarian lokasi kost terdekat [10] serta sistem pengajuan berkas secara elektronik [11].

Penelitian sebelumnya mengenai pembuatan simulator mesin FSA telah dilakukan.

Penelitian pertama telah berhasil membuat suatu simulator untuk mengecek apakah suatu inputan diterima atau tidak oleh mesin DFA [12]. Sedangkan penelitian kedua telah berhasil membuat suatu simulator FSA yang digunakan untuk melakukan pengecekan terhadapat inputan-inputan string dalam bentuk aplikasi android [13]. Kelemahan dari kedua hasil penelitian sebelumnya adalah mesin automata masih ditentukan diawal sehingga dapat dikatakan mesin DFA masih bersifat statis.

Pada penelitian ini akan dibangun simulator yang bersifat dinamis dimana pengguna dapat menginputkan sejumlah state, transisi, keadaan akhir dan inputan yang akan di baca oleh state pada mesin DFA. Simulator DFA akan dibangun menggunakan bahasa pemrograman python.

II. TINJAUAN PUSTAKA

A. *Deterministic Finite Automata (DFA)*

DFA adalah salah satu model matematis dalam Teori Bahasa Formal dan Otomata. DFA digunakan untuk mengenali bahasa-bahasa formal yang dapat dihasilkan oleh bahasa atau tata bahasa tertentu [2]. Secara formal, DFA didefinisikan sebagai 5 tuple (Q, Σ, δ, q0, F), di mana:

- Q adalah himpunan keadaan (states) yang terbatas.
- Σ adalah alfabet, yaitu himpunan simbol-simbol masukan.
- δ adalah fungsi transisi, yaitu fungsi δ: Q × Σ → Q yang menghubungkan keadaan-keadaan dalam Q dengan simbol-simbol dalam Σ.
- q0 adalah keadaan awal (start state), yaitu q0 ∈ Q.
- F adalah himpunan keadaan akhir (final states), yaitu F ⊆ Q.

DFA membaca simbol-simbol dari alfabet Σ satu per satu, memulai dari keadaan awal q0, dan bergerak dari satu keadaan ke keadaan lain berdasarkan fungsi transisi δ. Setelah DFA selesai membaca seluruh masukan, DFA akan berada pada salah satu keadaan. Jika keadaan tersebut adalah keadaan akhir (F), maka DFA mengenali masukan sebagai bahasa formal tertentu. Jika tidak, maka DFA tidak mengenali masukan tersebut.

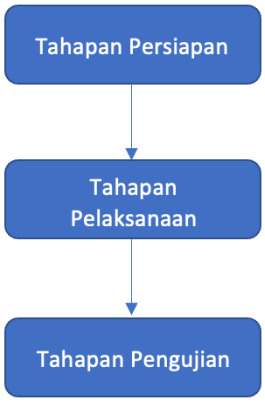
B. *Python*

Python adalah bahasa pemrograman tingkat tinggi yang umumnya digunakan dalam beragam bidang pengembangan perangkat lunak, analisis data, kecerdasan buatan, pemrosesan bahasa alami, dan banyak domain lainnya. *Python* memiliki ekosistem yang kaya dengan banyak library dan framework yang mendukung berbagai jenis pengembangan. *Python* didesain dengan filosofi kesederhanaan dan mudah dibaca, yang memudahkan pengembang untuk menulis kode yang jelas dan ekspresif.

Python bersifat *open-source* dan memiliki sintaks yang sederhana serta mudah dipahami, sehingga cocok untuk pemula maupun pengembang berpengalaman [14]. *Python* memiliki ekosistem yang kaya dengan banyak *library* dan *framework* yang mendukung berbagai jenis pengembangan.

III. METODOLOGI PENELITIAN

Langkah-langkah penelitian yang ada dalam penelitian ini terlihat pada Gambar 1. Tahap-tahap penelitian terdiri dari persiapan, pelaksanaan, dan pengujian.



Gambar 1. Tahapan Penelitian

A. Tahapan Persiapan

Tahap persiapan adalah langkah awal dalam metodologi penelitian. Pada tahap ini, peneliti mengidentifikasi dan merencanakan langkah-langkah yang akan diambil untuk menjawab pertanyaan penelitian atau mencapai tujuan

penelitian. Tahap persiapan mencakup beberapa aktivitas, antara lain:

1. Mencari referensi dan mempelajari buku terkait teori bahasa dan automata.
2. Identifikasi permasalahan dari penelitian sebelumnya.
3. Menyiapkan diagram transisi mesin DFA dari inputan pengguna

B. Tahapan Pelaksanaan

Pada tahapan pelaksanaan, akan dilakukan perancangan algoritma untuk simulator DFA. Adapun algoritma yang dibuat berupa algoritma untuk mengkonversi inputan dari pengguna berupa jumlah state, himpunan input, himpunan state, state awal dan state akhir serta transisi menjadi suatu mesin DFA. Berikut adalah tahapan algoritmanya:

Langkah 1: Menentukan Tuple dari DFA yaitu total state, transisi, state awal dan state akhir.

Langkah 2: Memasukkan string yang akan diuji oleh DFA yang di rancang pada langkah 1.

Langkah 3: Mengambil string yang akan diuji oleh DFA.

Langkah 4: Menjalankan DFA:

1. Inisialisasi status saat ini dengan state awal (S).
2. Iterasi melalui setiap simbol dalam masukan:
   - Gunakan fungsi transisi untuk memperbarui status berdasarkan simbol inputan.
   - Jika tidak ada transisi yang tersedia, berhenti dan nyatakan bahwa masukan tidak diterima (tidak valid).
3. Setelah semua simbol diinputkan, periksa apakah status saat ini adalah state akhir atau bukan. Jika status saat ini adalah status akhir, nyatakan bahwa masukan diterima (valid); jika tidak, nyatakan bahwa masukan tidak diterima.

Langkah 5: Memberikan hasil akhir apakah masukan diterima atau tidak serta menampilkan diagram state DFA yang dirancang pada langkah 1.

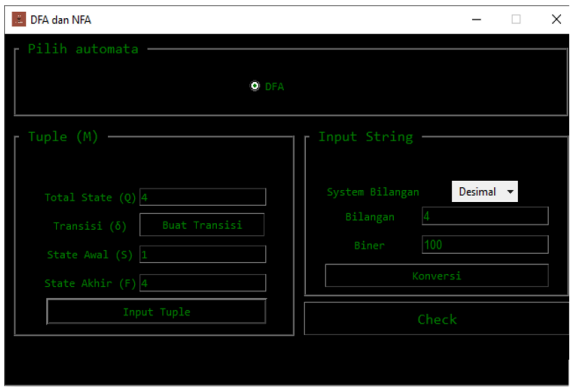
C. Tahapan Pengujian

Tahapan terakhir pada metodologi penelitian yaiut tahapan pengujian. Pada tahapan ini akan dilakukan pengujian sebuah inputan string ke dalam simulasi DFA dan memvalidasi hasil pembacaan string yang dibaca oleh DFA secara manual.

IV. HASIL DAN PEMBAHASAN

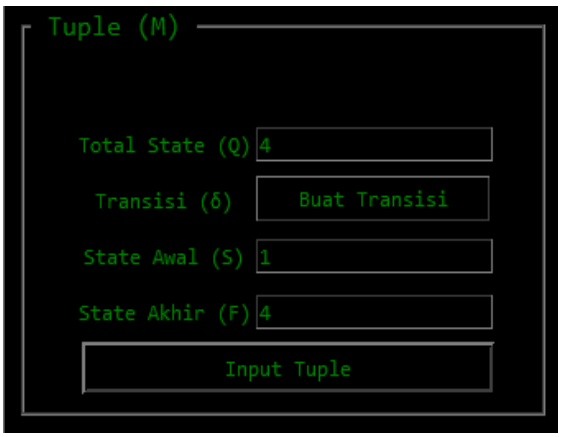
A. Tampilan Program

Tampilan simulator DFA yang dibangun menggunakan bahasa pemrograman *python* dapat dilihat pada Gambar 2.



Gambar 2. Tampilan Awal Simulator DFA

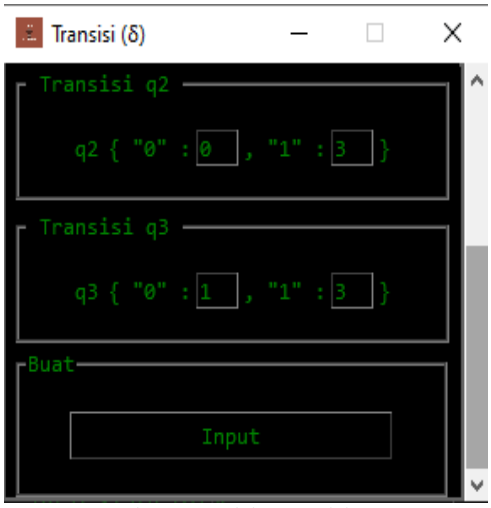
Pada bagian Tuple (M) yang ditunjukkan pada Gambar 3, masukkan “total state (Q)” untuk menentukan banyaknya state yang akan dibuat. Setelah itu tentukan “State Awal (S)” untuk menentukan awal mula state dan “State Akhir (F)” untuk menentukan akhir dari total state yang dibuat. Kemudian klik “Buat Transisi”.



Gambar 3. Inputan Tuple Untuk Mesin DFA

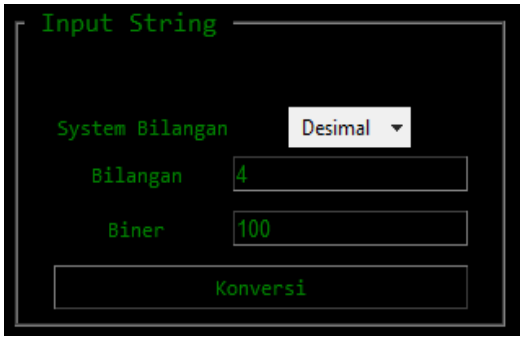
Setelah mengklik tombol “Buat Transisi” akan memunculkan jendela baru untuk memasukkan transisi (dengan angka sebagai notasi states) dari inputan “0” dan “1”. Setelah selesai memasukkan klik “Input” untuk disimpan

di dalam sistem. Adapun Gambar 4 menunjukkan tampilan jendela "Transisi (δ)" yang diinputkan oleh pengguna.



Gambar 4. Jendela Transisi DFA

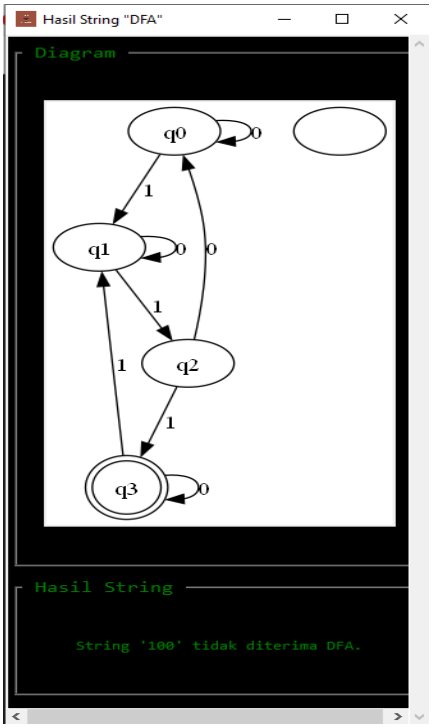
Setelah memasukkan semua inputan yang diperlukan, pengguna dapat menekan tombol "Input Tuple" untuk memproses dan menyimpan semua state. Kemudian pengguna dapat menginputkan string yang ingin diuji oleh mesin DFA yang baru saja dibuat. Gambar 5 menunjukkan tampilan untuk memasukkan string yang akan diuji.



Gambar 5. Masukan String pada Mesin DFA

Pada Bagian "Input String", pengguna memasukkan bilangan yang inin diuji kemudian klik tombol "Konversi" untuk mengkonversi inputan bilangan menjadi biner (0 dan 1).

Gambar 6. Memeriksa Hasil DFA



Gambar 7. Hasil DFA dari String yang Dimasukkan

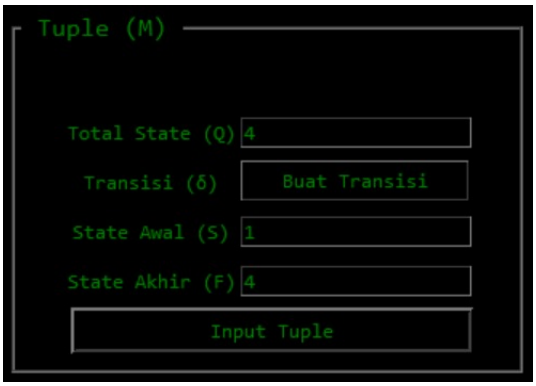
Setelah semua inputan sudah dimasukkan, tekan tombol "*Check*" yang di tunjukkan pada Gambar 6 untuk men-*generate graph* dan menentukan apakah string diterima atau tidak. Tampilan jendela "Hasil String DFA" dapat dilihat pada Gambar 7. Graph diagram berhasil di-*generate* sesuai inputan pengguna dan berhasil menguji masukan string.

B. Pengujian

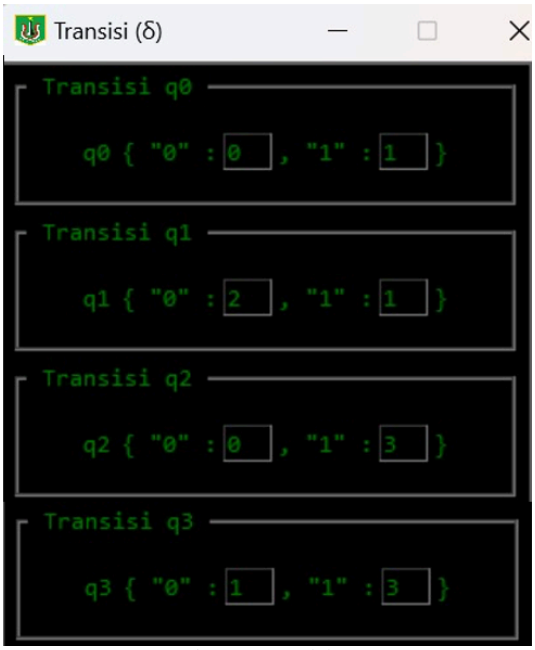
Pengujian (*testing*) DFA (*Deterministic Finite Automaton*) penting untuk memastikan bahwa DFA yang telah dirancang oleh simulator bekerja dengan benar sesuai dengan spesifikasi yang diinginkan. Berikut adalah pengujian yang dilakukan:

1. Pengujian string diterima

Gambar 8 menunjukkan inputan tuple untuk inisialisasi DFA sedangkan Gambar 9 menunjukkan inputan transisinya. Gambar 10 menunjukkan inputan string yang akan diujikan dan Gambar 11 menunjukkan hasil dari pengujian DFA menggunakan simulator.

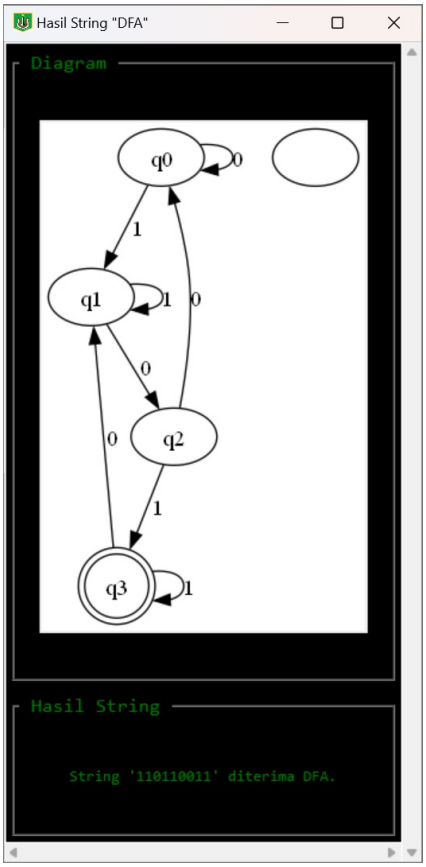


Gambar 8. Inputan Tuple DFA



Gambar 9. Transisi DFA

Gambar 10. String yang Diuji



Gambar 11. Hasil Uji Dari Tuple dan String

Untuk membuktikan kebenaran hasil tersebut maka dapat dilakukan pengujian manual terhadap tuple dan input string yang diberikan sebagai berikut:

$$
\begin{aligned}
\delta(q0, 110110011) &= \delta(q1, 10110011) \\
&= \delta(q1, 0110011) \\
&= \delta(q2, 110011) \\
&= \delta(q3, 10011) \\
&= \delta(q3, 0011) \\
&= \delta(q1, 011) \\
&= \delta(q2, 11) \\
&= \delta(q3, 1) \\
&= q3
\end{aligned}
$$

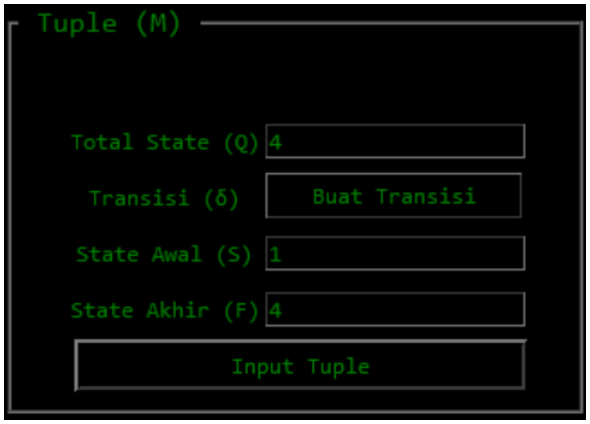
Karena q3 merupakan anggota himpunan state akhir maka string ‘110110011’ diterima oleh DFA. Adapun beberapa pengujian string yang dilakukan dapat dilihat pada Tabel 1.

Tabel 1. Pengujian Input yang Diterima

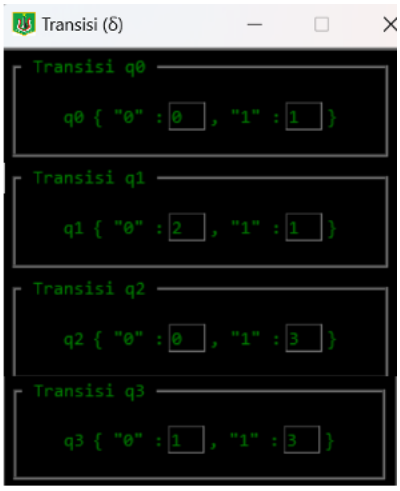
String	Hasil Mesin DFA	Kesimpulan
101	[x] Diterima [ ] Ditolak	Benar
01101	[x] Diterima [ ] Ditolak	Benar
00101	[x] Diterima [ ] Ditolak	Benar
1100101	[x] Diterima [ ] Ditolak	Benar
1011	[x] Diterima [ ] Ditolak	Benar

2. Pengujian string di tolak

Gambar 12 menunjukkan inputan tuple untuk inisialisasi DFA sedangkan Gambar 13 menunjukkan inputan transisinya. Gambar 14 menunjukkan inputan string yang akan diujikan dan Gamba 15 menunjukkan hasil dari pengujian DFA menggunakan simulator.

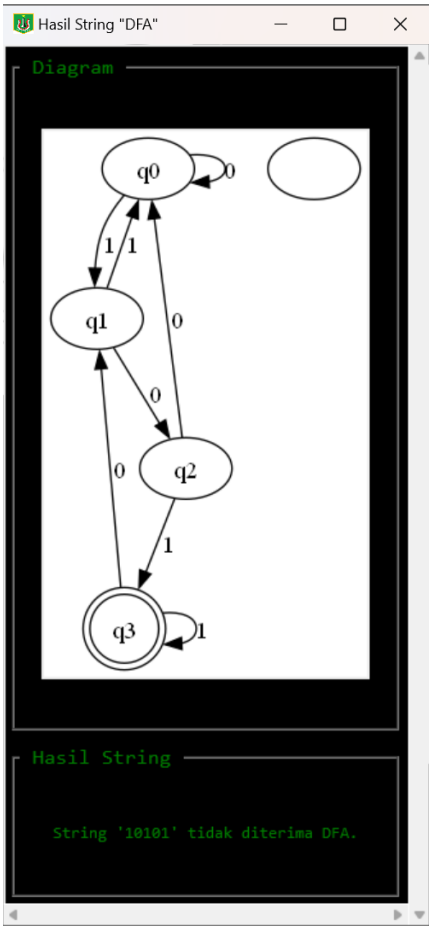


Gambar 12. Inputan Tuple DFA



Gambar 13. Inputan Transisi

Gambar 14. Inputan String yang Diuji



Gambar 15. Hasil Uji Berdasarkan Tuple dan String

Untuk membuktikan kebenaran hasil tersebut maka dapat dilakukan pengujian manual terhadap tuple dan input string yang diberikan sebagai berikut:

$$\begin{aligned}\delta(q0, 10101) &= \delta(q1, 0101)\\ &= \delta(q2, 101)\\ &= \delta(q3, 01)\\ &= \delta(q1, 1)\\ &= q0\end{aligned}$$

Karena q0 bukan merupakan anggota himpunan state akhir maka string ‘10101’ tidak diterima atau ditolak oleh DFA. Adapun beberapa

pengujian string yang dilakukan dapat dilihat pada Tabel 2.

Tabel 2. Pengujian Input yang Ditolak

String	Hasil Mesin DFA	Kesimpulan
10101	[ ] Diterima [x] Ditolak	Benar
111	[ ] Diterima [x] Ditolak	Benar
110000	[ ] Diterima [x] Ditolak	Benar
0100	[ ] Diterima [x] Ditolak	Benar
11000010	[ ] Diterima [x] Ditolak	Benar

Dari hasil pengujian diatas dapat disimpulkan bahwa model DFA yang dibangun sudah dapat dengan baik mengenali apakah string diterima atau di tolak. Hasil pengujian menunjukkan 100% kesimpulan sesuai dengan yang diharapkan.

C. Pembahasan

Dalam perbandingan dengan penelitian sebelumnya, yang dirujuk pada [12] dan [13], penelitian ini memperkenalkan suatu pendekatan baru dalam pembangunan simulator DFA. Pada penelitian sebelumnya, desain DFA telah ditentukan terlebih dahulu dan bersifat statis, yang berarti DFA tidak dapat berubah selama proses simulasi. Namun, penelitian ini telah berhasil membangun simulator DFA yang bersifat dinamis.

Perbedaan utama antara penelitian ini dan penelitian sebelumnya adalah dalam pendekatan desainnya. Penelitian sebelumnya, yang dirujuk pada [12] dan [13], mendasarkan simulasi pada DFA yang telah ditentukan sebelumnya, sedangkan dalam penelitian ini, simulator DFA mampu menerima masukan tuple dari pengguna. Hal ini berarti pengguna dapat mengubah DFA yang disimulasikan sesuai dengan masukan yang diberikan, sehingga menjadikannya bersifat dinamis.

Pendekatan dinamis dalam penelitian ini membuka peluang baru dalam pemodelan dan simulasi DFA, memungkinkan pengguna untuk menjalankan berbagai eksperimen dengan mudah. Hal ini dapat berguna dalam pengujian, analisis, dan eksplorasi berbagai jenis DFA yang mungkin diperlukan dalam konteks penelitian atau pengembangan perangkat lunak tertentu.

V. SIMPULAN

Simulator mesin DFA dengan menggunakan bahasa pemrograman *python* telah berhasil dibangun. Mesin simulator DFA dapat meng-*generate* DFA sesuai dengan masukan pengguna berupa jumlah state, keadawal awal state, keadaan akhir state, transisi dan inputan string. Berdasarkan pengujian sejumlah string input pada DFA yang telah di *generate* oleh simulator, dapat disimpulkan bahwa algoritma yang telah dirancang berfungsi dengan baik, atau dengan kata lain, algoritma tersebut dapat dianggap valid.

DAFTAR PUSTAKA

[1] A. Salomaa, *Theory of Automata*. Elsevier, 2014.
[2] A. Adil, *Pengantar Teori Bahasa Formal, Otomata, Dan Komputasi*. Yogyakarta: CV. Budi Utama, 2018.
[3] D. Puspita and D. Gusmaliza, *Teori Bahasa dan Otomata*. Yogyakarta: CV. Bintang Semesta Media, 2022.
[4] A. Aranski, *Teori Bahasa dan Otomata*. Padang: Pustaka Galeri Mandiri, 2018.
[5] T. Hari Wicaksono, F. Dwiki Amrizal, H. Atun Mumtahana, and J. Setia Budi No, "Pemodelan Vending Machine dengan Metode FSA (Finite State Automata)," *DoubleClick J. Comput. Inf. Technol.*, vol. 2, no. 2, pp. 66–69, 2019, [Online]. Available: http://e-journal.unipma.ac.id/index.php/doubleclick/article/view/3901
[6] S. Hidayat, F. Said, F. Titiani, and W. Gata, "Desain Konsep Finite State Automata (Fsa) Pada Simulasi Vending Machine (Vm) Masakan Padang," *J. Inf. Syst. Informatics Comput.*, vol. 5, no. 1, p. 134, 2021, doi: 10.52362/jisicom.v5i1.442.
[7] R. Suharsih and F. Atqiya, "Penerapan Konsep Finite State Automata (FSA) pada Aplikasi Simulasi Vending Machine Yoghurt Walagri," *Edsence J. Pendidik. Multimed.*, vol. 1, no. 2, pp. 71–78, 2019, doi: 10.17509/edsence.v1i2.21778.
[8] N. A. Maori and O. Setiono, "Penerapan Finite State Automata Pada Simulasi Vending Machine Water and Ice Cube," vol. 2, no. 4, pp. 1032–1039, 2023.
[9] F. Aziz, "Penerapan Konsep Finite State Automata Dalam Proses Pendaftaran Kelas Kursus Bahasa Inggris Pada Tempat Kursus," *Matics*, vol. 12, no. 2, pp. 93–98, 2021, doi: 10.18860/mat.v12i2.9330.
[10] E. Sofyan and Gusrianty, "Penerapan Metode MultiFactor Evaluation Process (MFEP) dan Algoritma Finite State Automata (FSA) untuk

Pencarian Lokasi Kost Terdekat," *J. Mhs. Apl. Teknol. Komput. dan Inf.*, vol. 2, no. 1, 2020.
[11] V. Yulianty *et al.*, "Penerapan Finite State Automata Pada Pengajuan Berkas," pp. 282–289, 2021.
[12] Suparyanto, "Simulator Pengenal String Yang Diterima Sebuah Deterministic Finite Automata (DFA)," pp. 377–381, 2017.
[13] J. Sistem Komputer dan Kecerdasan Buatan and J. Selatan, "Simulator String Yang Diterima Finite State Automata (FSA) Berbasis Android," vol. III, no. 1, pp. 1–6, 2019.
[14] M. Asif, "Python for Geeks: Build Production-ready Applications Using Advanced Python Concepts and Industry Best Practices," Britania Raya: Packt Publishing, 2021, p. 546.