

LAPORAN TUGAS BESAR
IF2124 TEORI BAHASA FORMAL DAN AUTOMATA
HTML CHECKER



DISUSUN OLEH :
Kelompok UrbanGym
13522001 Mohammad Nugraha Eka Prawira
13522008 Ahmad Farid Mudrika
13522009 Muhammad Yusuf Rafi

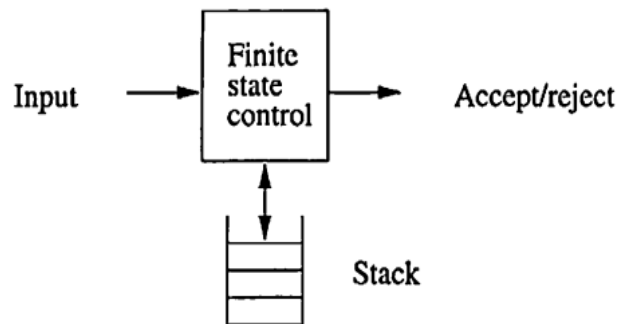
PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

BAB I

TEORI DASAR

2.1 Push Down Automata (PDA)

Pushdown automata pada dasarnya adalah sebuah automata yang bersifat nondeterministik dengan ϵ -transisi yang diperbolehkan dan satu kemampuan tambahan: **sebuah tumpukan (stack)** di mana ia dapat menyimpan sebuah rangkaian "*stack symbols*." Keberadaan tumpukan ini berarti, berbeda dengan *finite automaton*, pushdown automata dapat mengingat informasi dalam jumlah tak terbatas. Namun, berbeda dengan komputer serbaguna yang juga memiliki kemampuan untuk mengingat informasi dalam jumlah besar, pushdown automata hanya dapat mengakses informasi pada tumpukannya secara *last-in-first-out*. Akibatnya, ada bahasa-bahasa yang dapat dikenali oleh beberapa program komputer, tetapi tidak dapat dikenali oleh pushdown automata mana pun. Faktanya, automaton pushdown mengenali semua dan hanya bahasa-bahasa konteks bebas.



Gambar 3.1.1 pushdown automata pushdown pada dasarnya adalah sebuah automaton hingga dengan struktur data tumpukan.

Pushdown Automata membaca masukan, satu simbol pada satu waktu. Automaton pushdown diizinkan untuk mengamati simbol di bagian atas tumpukan dan untuk melakukan transisi berdasarkan pada keadaan saat ini, simbol masukan, dan simbol di bagian atas tumpukan. Atau dengan alternatif, ia dapat melakukan transisi "spontan," menggunakan ϵ sebagai masukannya alih-alih simbol masukan. Sebuah tumpukan (stack) melakukan dua operasi dasar. Operasi pertama adalah "Push," di mana sebuah simbol baru ditambahkan di bagian paling atas tumpukan. Operasi kedua adalah "Pop," di mana simbol yang berada di paling atas tumpukan dibaca dan kemudian dihapus dari tumpukan tersebut. Operasi Push menambahkan elemen baru ke dalam tumpukan, sedangkan operasi Pop membaca dan menghapus elemen teratas dari tumpukan.

Persamaan umum untuk menggambarkan sebuah PDA (Pushdown Automaton) adalah sebagai berikut:

$$P = (Q, \Sigma, S, \delta, q_0, I, F)$$

Di mana:

- Q adalah himpunan terbatas dari keadaan (states).
- Σ mewakili alfabet masukan (input alphabet).
- S adalah simbol-simbol pada tumpukan (stack symbols).
- δ adalah fungsi transisi: $\delta: Q \times (\Sigma \cup \{\epsilon\}) \times S \rightarrow 2^{(Q \times S^*)}$, yang menggambarkan transisi antar keadaan berdasarkan pada keadaan saat ini, simbol masukan (termasuk ϵ untuk string kosong), simbol pada tumpukan, keadaan berikutnya yang dihasilkan, dan operasi pada tumpukan (simbol yang ditambahkan ke tumpukan).
- q_0 adalah keadaan awal, di mana q_0 adalah elemen dari Q .
- I mewakili simbol pada bagian atas tumpukan yang awal, di mana I adalah elemen dari S .
- F adalah himpunan keadaan penerimaan, di mana F adalah subhimpunan dari Q .

Dalam notasi ini, P adalah representasi formal dari sebuah Pushdown Automaton yang terdiri dari informasi mengenai keadaan, alfabet masukan, simbol tumpukan, fungsi transisi, keadaan awal, simbol tumpukan awal, dan himpunan keadaan penerimaan.

Kita telah berasumsi bahwa PDA menerima inputnya, memprosesnya, dan memasuki accepting state. Pendekatan ini dapat disebut “acceptance by final state” (penerimaan berdasar final state). Terdapat pendekatan lain untuk mendefinisikan bahasa yang diterima PDA yaitu “accepted by empty stack”, artinya suatu set input string yang menyebabkan PDA untuk mengosongkan stack-nya, mulai dari awal. Kedua pendekatan ini ekuivalen, yaitu, untuk tiap bahasa yang diterima PDA berdasar final state, terdapat PDA lain yang menerima bahasa tersebut sebagai empty stack. Dalam implementasi PDA empty stack, karena kita tidak peduli dengan accepting state, kita dapat mengabaikan element ke-7 dari 7-tuple definisi PDA. Jadi, kita dapat menulis definisi PDA P sebagai 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z_0)$.

2.2 Sintaks HTML dalam pembuatan PDA

Diperlukan implementasi Pushdown Automata (PDA) untuk memeriksa kebenaran struktur dasar HTML, termasuk pengecekan nama tag dan atributnya. Ini penting karena HTML yang valid memengaruhi aspek SEO, aksesibilitas, maintenance, kecepatan render, dan kesan profesional dalam pembuatan situs web. Maka dari itu, dibuatkan program yang dapat memeriksa kesesuaian tag pembuka dan penutup, serta atribut yang tepat, memastikan HTML terbentuk dengan benar.

Tag	Attribute	Tag	Attribute
html		b	
head		abbr	
body		strong	
title		small	
link (<i>singleton tag</i>)	rel, href	hr (<i>singleton tag</i>)	
script	src	div	
h1,h2,h3,h4,h5, h6		a	href
p		img (<i>singleton tag</i>)	src, alt
br (<i>singleton tag</i>)		button	type
em		form	action, method
tr		input (<i>singleton tag</i>)	type
td		table	
th			

Pada sintaks HTML, html selalu menjadi elemen terluar. Head selalu mendahului body. Title selalu di dalam head. Elemen-elemen lain berada di dalam body. *Singleton tag* menandakan elemen yang tidak memerlukan terdapat tag penutup untuk menjadi valid. Dalam pembuatan PDA, diimplementasikan *empty stack*, jadi *accepting state* tidak diperlukan atau dapat diabaikan. Untuk pengecekan kebenaran HTML, akan diperiksa pada keadaan stack, jika fungsi transisi menghasilkan *empty stack*, maka syntax tersebut akan diterima, namun jika tidak menghasilkan *empty stack*, maka sintaks tersebut akan ditolak.

BAB II

IMPLEMENTASI PDA

Untuk program kami, kami mendefinisikan PDA di file terpisah, “pda_definition.txt”.

```
src > pda_definition.txt
1 Q headstate bodystate tablestate stringstate trstate
2 <html> </html> <head> </head> <body> </body> <link> </link> <title> </title> <script> </script>
3 ht he body ttl link script p br em b abbr strong small hr div a img button
4 Q
5 Z
6 E
7 Q <html> Z Q ht
8 Q <head> e headstate he
9 headstate <title> e stringstate ttl
10 stringstate </title> ttl headstate e
11 stringstate <link> e stringstate e
12 headstate <link> e headstate e
13 headstate <script> e stringstate script
14 stringstate </script> script headstate e
15 headstate </head> he bodystate e
```

- Baris pertama merupakan himpunan state PDA.
- Baris kedua merupakan himpunan simbol input.
- Baris ketiga merupakan himpunan alfabet yang digunakan dalam stack.
- Baris keempat merupakan start state dari PDA.
- Baris kelima merupakan start symbol dari stack PDA.
- Baris keenam merupakan pilihan PDA final state atau empty stack (F atau E). Dalam hal ini, PDA kami menggunakan penerimaan berdasar empty stack.
- Baris ketujuh dan seterusnya merupakan 5-tuple yang menyatakan aksi PDA, yaitu state awal, input symbol, take from stack, state berikutnya, push to stack. Kami juga menyatakan atribut dari tiap tag di sini. Dalam implementasi kami, e merupakan epsilon.

Contoh:

Q <html> Z Q ht

berarti pada state Q, jika menerima input <html> dan top dari stack saat ini adalah Z, program akan me-pop Z dari stack, tetap di state Q, dan memasukkan ‘ht’ ke top stack PDA.

stringstate <link> e stringstate e

Berarti pada state stringstate, jika menerima input <link>, tidak peduli apa yang ada di top stack, program akan tetap di stringstate dan tidak akan memasukkan apapun ke top stack PDA.

 src w e e

Berarti pada tag , terdapat attribute src yang bersifat wajib (w). Kami menggabungkannya dengan tuple aksi PDA untuk efisiensi program.

 alt h e e

Berarti pada tag , terdapat atribut alt yang bersifat opsional.

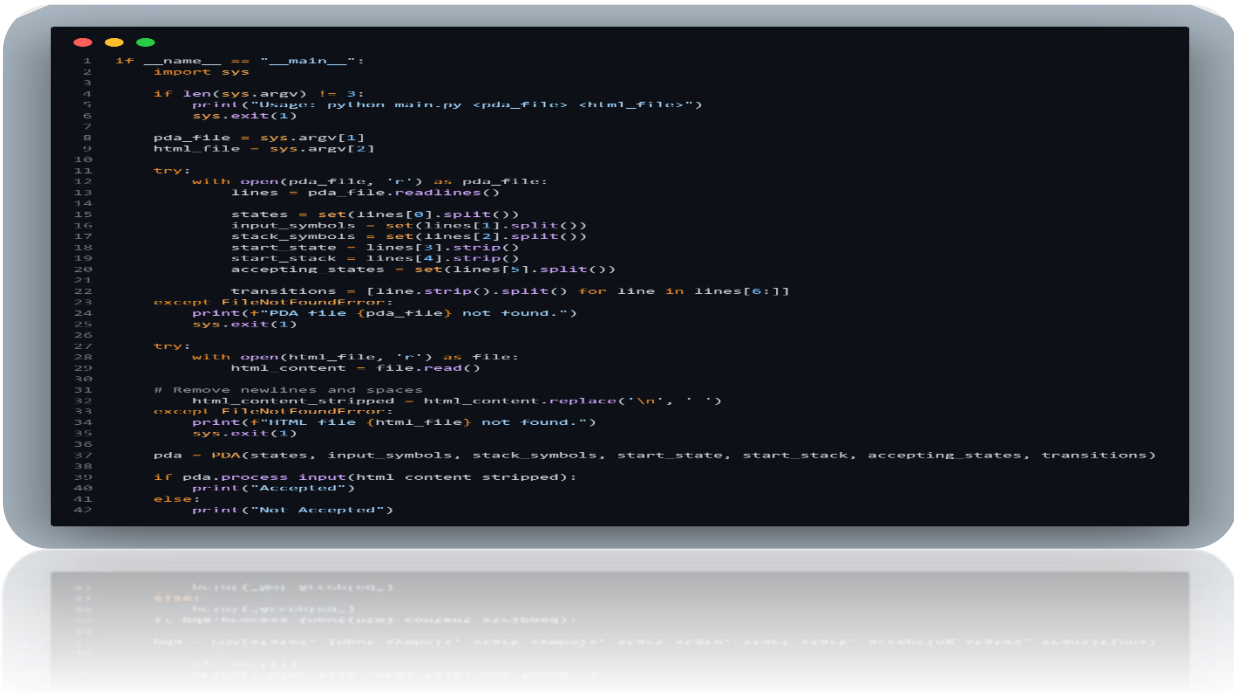
STATE-STATE PDA

PDA kami memiliki 6 state, yaitu Q, headstate, bodystate, tablestate, stringstate, trstate. Q adalah state awal, headstate menyatakan bahwa PDA saat ini sedang di dalam head, bodystate menyatakan bahwa PDA saat ini sedang di dalam body, tablestate menyatakan bahwa PDA saat ini sedang di dalam tag table, stringstate menyatakan tag yang dapat menerima string di antara pembuka dan penutupnya, dan trstate menyatakan bahwa PDA saat ini sedang di dalam tag tr, di dalam tag table.

BAB III

IMPLEMENTASI

3.1 Procedur main Program Utama



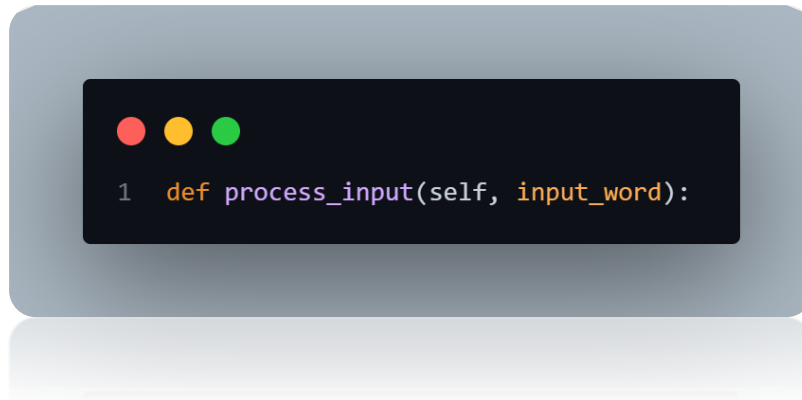
Gambar Program main

Program utama berbasis Bahasa *Python* dengan teknis program membaca file dua file masukkan yaitu file txt yang berisi definisi PDA yang digunakan dan file html yang akan pengecekan. Program main akan membaca text PDA dan menambahkan tiap definisi PDA ke dalam variable.

- Variabel *states* berisi state state yang terdefiisi dalam PDA.
- Variabel *input_symbols* berisi symbol symbol yang terdefinisi sebagai inputan.
- Variables *stack_symbols* berisi symbol dari stack yang terdefinisi.
- Variabel *start_state* berisi kondisi state awal.
- Variable *start_stack* berisi kondisi stack awal
- Variable *accepting_states* berisi kondisi state final yang valid
- Variabel *transitions* berisi transisi transisi yang terdefinisi.


Text html akan dibaca dan diubah juga menjadi sebuah variable *html_content_stripped* yang kemudian akan dicek. Proses selanjutnya memanggil class method *PDA* ke dalam variable *pda* lalu mengecek hasil proses dengan fungsi *process_input* yang akan menentukan html tersebut valid atau tidak.

3.2 Fungsi *process_input*



Fungsi ini berguna untuk menentukan apakah text html tersebut valid dan sesuai dengan PDA yang telah kita definisikan. Proses awalnya adalah memecah hasil bacaan text html dan membaginya agar membaca bagian tag dan atributnya. Kemudian akan dicek setiap text htmlnya secara berurutan mengikuti konsep PDA dan menentukan tiap transisinya berdasarkan tag yang dibaca. Fungsi ini akan memanggil fungsi *find_transition* untuk mengecek transisi tiap tag yang diberikan. Apabila pada pembacaan tag tersebut tidak terdapat ataupun tidak mengikuti definisi PDA nya, maka text html tersebut akan di reject dan fungsi ini akan mengembalikan nilai False. Sebaliknya, jika text html tersebut valid dan mengikuti tiap transisi yang ada dan berhenti pada kondisi stack kosong, maka text html tersebut akan di accept dan fungsi akan mengembalikan nilai True.

3.3 Fungsi find_transition






```
1 def find_transition(self, current_state, input_symbol, stack_top):
2     for transition in self.transitions:
3         if (
4             transition[0] == current_state
5             and (transition[1] == input_symbol )
6             and (transition[2] == stack_top or (transition[2]=='e'))
7         ):
8             return transition[3], transition[4], transition[2]
9     return None
```

Fungsi ini akan mengembalikan tuple yang berisi state yang dituju, symbol stack yang ingin di push, dan stack yang ingin di pop berdasarkan parameter yang diterima yaitu state saat ini, inputan symbol yang dibaca dan stack teratasnya.

3.4 Pda_definition.txt

Teks definisi PDA berisi total states yang digunakan, input word yang diberikan, stack symbol, state awal, stack awal, stack kosong sebagai akhir kondisi yang diterima, dan transisi yang digunakan. State yang digunakan adalah Q sebagai state awal, headstate sebagai penanda state di bagian head html, bodystate sebagai penanda state di bagian body html, stringstate sebagai penanda ketika sedang berada pada tag yang mengurus bagain text pada html dan tablestate sebagai penanda ketika sedang berurusan dengan tabke pada html. Stack awal berisi Z dan menerima stack kosong untuk menandakan valid tidaknya text html tersebut. Lalu transisi yang berupa state awal, inputan yang dibaca, stacksymbol yang di push, state selanjutnya dan stacksymbol yang di pop.

BAB IV Eksperimen

Keterangan	Text html input
<p>File rejected1.html berisi text html yang salah karena bagian head muncul setelah body</p>	 <pre> 1 <html> 2 <body> 3 <h1>Hello, World!</h1> 4 <p>This is a simple webpage.</p> 5 </body> 6 <head> 7 <title>Simple Webpage</title> 8 </head> 9 </html> 10 </pre>
<p>File rejected2.html berisi text html dengan bagian tag html yang typo menjadi 'hmif'.</p>	 <pre> 1 <hmif> 2 <head> 3 <title>Simple Webpage</title> 4 </head> 5 <body> 6 <h1>Hello, World!</h1> 7 <p>This is a simple webpage.</p> 8 </body> 9 </hmif> 10 </pre>
<p>File rejected3.html berisi text html yang salah karena tidak memiliki bagian head.</p>	 <pre> 1 <html> 2 <body> 3 <h1>Hello, World!</h1> 4 <p>This is a simple webpage.</p> 5 </body> 6 </html> 7 </pre>

File rejected4.html berisi tag img yang tidak mengandung atribut wajib yaitu 'src'.

```
1 <html>
2 <head>
3   <title>Simple Webpage</title>
4 </head>
5 <body>
6   <!-- Bagian utama web -->
7   <h1>Hello, World!</h1>
8   <h2>Welcome to my page</h2>
9   <img alt="welcome banner">
10  <p>This is a <em>simple</em> webpage.</p>
11
12
13   <!-- Custom element -->
14   <div id="footer" class="footer"> This is the end of the page </div>
15 </body>
16 </html>
17
18
```

File rejected5.html berisi text html yang salah karena value method tidak termasuk value yang diperbolehkan(POST,GET).

```
1 <html>
2 <head>
3   <title>Simple Webpage</title>
4 </head>
5 <body>
6 <h2>HTML Forms</h2>
7 <form action="/action_page.php" method="IEHTTP">
8   <div id="label">first name:</div><br>
9   <input type="text" id="fname"><br>
10  <div id="label">last name:</div><br>
11  <input type="text" id="lname"><br>
12  <button type="submit">Submit</button>
13 </form>
14 <p>If you click the "Submit" button, the form data will be sent to a page called "/action_page.php".</p>
15 </body>
16 </html>
17
18
19
20
21
```

File rejected6.html berisi text html yang salah karena tag p bukanlah void element.

```
1 <html>
2 <head>
3   <title>Simple Webpage</title>
4   <script>
5     document.getElementById("demo").innerHTML = "Hello JavaScript!";
6   </script>
7 </head>
8 <body>
9
10  <h1>The script element</h1>
11
12  <p id="demo">
13
14  </body>
15 </html>
16
```

<p>File accepted1.html berisi text html yang benar mengikuti peraturan html.</p>	 <pre> 1 <html> 2 <head> 3 <title>Simple Webpage</title> 4 </head> 5 <body> 6 <h1>Hello, World!</h1> 7 <h2>Welcome to my page</h2> 8 </pre>
<p>File accepted2.html berisi text html yang benar mengikuti peraturan html.</p>	 <pre> 1 <html> 2 <head> 3 <title>Simple Webpage</title> 4 </head> 5 <body> 6 <!-- Region utama web --> 7 <h1>Hello, World!</h1> 8 <h2>Welcome to my page</h2> 9 <hr> 10 </pre>
<p>File accepted3.html berisi text html yang benar mengikuti peraturan html.</p>	 <pre> 1 <html> 2 <head> 3 <title>Simple Webpage</title> 4 </head> 5 <body> 6 <h2>HTML Forms</h2> 7 8 <form action="/> </pre>
<p>File accepted4.html berisi text html yang benar mengikuti peraturan html.</p>	 <pre> 1 <html> 2 <head> 3 <title>Simple Webpage</title> 4 <script> 5 document.getElementById("demo").innerHTML = "Hello JavaScript!"; 6 </script> 7 </head> 8 <body> 9 10 <h1>The script element</h1> 11 12 <p id="demo"></p> 13 14 </body> 15 </html> 16 </pre>

Keterangan	Hasil Program
<p>Hasil penjalanan program pada text- text html yang salah.</p>	 <pre> 49 transition = self.find_transition(current_state, tag_name, current PROBLEMS TERMINAL DEBUG CONSOLE PORTS COMMENTS OUTPUT PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt rejected1.html Not Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt rejected2.html Not Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt rejected3.html Not Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt rejected4.html Salah di : ['src'] Not Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python main2.py pda_definition.txt rejected5.html Salah di : TEMBAK Not Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt rejected6.html Not Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> </pre>
<p>Hasil Penjalanan program pada text-text html yang benar.</p>	 <pre> PROBLEMS TERMINAL DEBUG CONSOLE PORTS COMMENTS OUTPUT PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt accepted1.html Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt accepted2.html Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python main2.py pda_definition.txt accepted3.html Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt accepted4.html Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> python ma.py pda_definition.txt accepted5.html Accepted PS C:\ITB\SEM 3\TBFO\TBFO-01> </pre>

BAB V

Github Repository dan Diagram

GITHUB :

<https://github.com/EkaaPrawiraa/TBFO-01>

DIAGRAM :

<https://drive.google.com/file/d/1nEeAZMmT-7fc3dFexWd6jH1RRfA571q/view?usp=sharing>

BAB VI

Pembagian Tugas

PIC (Person in Charge)	Job Description
13522001 Mohammad Nugraha Eka Prawira	PDA_DEFINITION.txt dan Main.py
13522008 Ahmad Farid Mudrika	PDA_DEFINITION.txt dan Main.py
13522009 Muhammad Yusuf Rafi	PDA_DEFINITION.txt dan diagram state

BAB VII

Referensi

Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2006). Automata Theory, Languages, and Computation (edisi ke-3). Cornell University, Stanford University: Addison Wesley.