PERTEMUAN 2

NOTASI GRAMMAR DAN BAHASA, HIMPUNAN, RELASI

A. TUJUAN PEMBELAJARAN

Pada pertemuan ini akan dijelaskan mengenai notasi grammar, konsep dan notasi bahasa, serta himpunan relasi. Setelah menyelesaikan materi pada pertemuan ini, mahasiswa mampu

1. Mendefinisikan pengertian notasi grammar.
2. Menjelaskan konsep dan notasi bahasa.
3. Menjelaskan himpunan dan relasi.

B. URAIAN MATERI

1. Pengertian Notasi Grammar

Grammar G didefinisikan sebagai pasangan 4 tuple : VT , VN , S, dan Q, dan dituliskan sebagai G(VT , VN , S, Q), dimana :

a. VT : himpunan simbol-simbol terminal (atau himpunan token - token, atau alfabet)

b. VN : himpunan simbol-simbol non terminal

c. S $\in$ VN : simbol awal (atau simbol start)

d. Q : himpunan produksi

Berdasarkan komposisi bentuk ruas kiri dan ruas kanan produksinya ($\propto \rightarrow \beta$), Grammar diklasifikasikan menjadi 4, yaitu:

a. Grammar tipe ke-0: Unrestricted Grammar (UG)

Ciri: $\propto, \beta \epsilon$ (V_T | V_N)*, | $\propto$ | > 0

b. Grammar tipe ke-1: Context Sensitive Grammar (CSG)

Ciri: $\propto, \beta \epsilon$ (V_T | V_N)*, | $\propto$ | $\leq$ | β |

c. Grammar tipe ke-2: Context Free Grammar (CFG)

Ciri: $\propto, \epsilon$ V_N, $\beta \epsilon$ (V_T | V_N)*

d. Grammar tipe ke-3: Regular Grammar (RG)

Ciri: $\propto, \epsilon$ V_N, $\beta\epsilon$ {V_T, V_T V_N} atau $\propto, \epsilon$ V_N, $\beta\epsilon$ {a, Bc}

Tipe sebuah grammar ditentukan dengan aturan sebagai berikut:

Suatu bahasa dikatakan sebagai bahasa tipe-i (i=0, 1, 2, 3) jika dapat ditentukan oleh tata bahasa tipe-i tetapi tidak dapat ditentukan tata bahasa tipe (i+1).

2. Konsep Dan Notasi Bahasa

a. Teori Bahasa

Dalam semantik formal, ilmu komputer dan linguistik, tata bahasa formal (juga disebut aaturan formasi) adalah deskripsi yang tepat dari bahasa formal, yaitu sekumpulan string diatas beberapa alfabet, dengan kata lain, tata bahasa mendeskripsikan urutan simbol mana (string) dalam suatu bahasa merupakan kata atau pernyataan yang valid dalam bahasa tersebut, tetapi tidak mendeskripsikan semantiknya (yaitu arti dari kata tersebut).

Tata bahasa biasanya dianggap sebagai sarana untuk menghasilkan semua string yang valid, itu juga dapat digunakan sebagai dasar untuk pengenal yang menentukan untuk setiap string yang diberikan apakah itu gramatikal (tata bahasa) untuk menggambarkan pengenakan semacam itu, teori bahasa formal menggunakan formalisme terpisah, yang dikenal seagai automata.

Tata bahasa juga dapat digunakan untuk menganalisis string bahasa, yaitu untuk mendeskripsikan struktur internalnya. Dalam ilmu komputer, proses ini dikenal dengan istilah parsing. Kebanyakan bahasa memiliki semantik yang sangat komposisional, yaitu arti ucapannya terstruktur menurut sintaksnya; oleh karena itu, langkah pertama untuk mendeskripsikan makna suatu ucapan bahasa adalah dengan menganalisisnya dan melihat bentuk analisisnya (dikenal dengan pohon parse dalam ilmu komputer dan sebagai struktur dalam tata bahasa generatif).

Aturan produksi menspesifikasikan bagaimana suatu tata bahasa mentransformasikan suatu string ke bentuk lainnya. Biasanya aturan produksi diberi simbol:

α→β

- Dimana, α adalah aturan produksi sebelah kiri

- β adalah aturan produksi sebelah kanan

- aturan produksi sebelah kiri menghasilkan aturan produksi sebelah kanan.

Simbol-simbol dalam aturan produksi dapat berupa simbol terminal maupun non terminal. Simbol terminal sudah tidak dapat diturunkan lagi. Simbol non-terminal dapat diturunkan lagi sampai menjadi simbol terminal. Simbol terminal biasanya ditulis dalam huruf kecil, sedangkan simbol non-terminal biasanya ditulis dalam huruf besar.

Ada beberapa kata dalam teori dan bahasa grammar :

1) Sintaks: setiap bahasa memiliki seperangkat aturan umum tentang bagaimana sebuah kata dan kalimat harus disusun. Aturan-aturan ini secara kolektif dikenal sebagai sintaks bahasa. Dalam pemrograman komputer, sintaks memiliki tujuan untuk menentukan bagaimana deklarasi, fungsi dan penulisan kode program harus diatur.

2) Semantic : semantik adalah penggambaran proses yang dilakukan komputer saat menjalankan program bahasa tertentu. Hal tersebut dapat ditunjukkan dengan mendeskripsikan hubungan antara input dan output suatu program, atau penjelasan bagaimana program akan dijalankan pada platform tertentu, sehingga tercipta model kompilasinya.

3) Sebuah simbol : simbol dalam pemrograman komputer adalah sebuah tipe data abtrak yang memiliki bentuk unik yang dapat dibaca manusia. Simbol dapat digunakan sebagai pengenal, dalam bahasa pemrograman, simbol dapat berupa huruf dan angka.

4) String (kata/untai): string adalah tipe data yang digunakan dalam pemrograman, seperti integer dan floating, tetapi digunakan untuk mewakili teks dapida angka. String terdiri dari sekumpulan karakter yang dapat berisi spasi dan angka. Misalnya kata “makan” dan frase kata “saya sedang makan” keduanya adalah string. Bahkan 1, 2, 3, 4, 5 bisa

dianggap sebagai string jika dituliskan dengan benar. Biasanya, programmer harus menyertakan string dala tanda kutip agar data dapat dikenali sebagai string dan bukan nama angka atau variabel.

5) String kosong adalah turunan string dengan panjang nol, sedangkan string nol tidak memiliki nilai sama sekali. String kosong direpresentasikan dengan " ". Ini adalah urutan karakter dari karakter nol. Sebuah string kosong diwakili oleh null. Ini dapat digambarkan sebagai tidak adanya instance string.

Dalam bahasa pemrograman, kalimat dikenal sebagai ekspresi, dan kata sebagai token. Kata terdiri atas beberapa karakter. Kelompok karakter yang membentuk sebuah token dinamakan lexeme untuk token tersebut. Setiap token yang dihasilkan, disimpan dalam tabel simbol.

Derivasi adalah sebuah proses dimana suatu himpunan produksi akan diturunkan / dipilah-pilah dengan melakukan sederetan produksi sehingga membentuk untai terminal.

1) Simbol-simbol berikut adalah simbol terminal
   a) huruf kecil awal alfabet, misalnya : a,b,c
   b) simbol operator, misalnya : +,-, dan x
   c) simbol tanda baca, misalnya (), , dan
   d) string yang tercetak tebal, misalnya if, then, dan else
2) Simbol-Simbol Non Terminal:
   a) huruf besar awal alfabet, misalnya : A, B, C
   b) huruf S sebagai simbol awal
   c) String yang tercetak miring, misalnya : expr dan stmt

b. Bahasa Pemrograman

Bahasa pemrograman adalah bahasa yang menjadi sarana manusia untuk berkomunikasi dengan komputer. Pikiran manusia yang tidak terstruktur atau tertata harus dibuat terstruktur dan tertata agar bisa berkomunikasi dengan sebuah komputer. Komputer itu sendiri memerlukan

kepastian dan logika yang benar untuk dapat melakukan suatu intstruksi atau perintah yang diinput oleh manusia atau user agar mendapatkan output yang diinginkan. Untuk itu dibutuhkan sebuah algoritma yang baik dan benar. Ada beberapa bahasa penggolongan bahasa pemrograman berdasarkan tingkat ketergantungannya dengan sebuah mesin :

1) Bahasa Mesin

Bahasa mesin adalah bahasa yang berisi kode-kode mesin yang hanya dapat diinterprestasikan langsung oleh mesin komputer. Bahasa mesin sering juga disebut native code (sangat tergantung pada mesin tertentu). Bahasa ini merupakan bahasa level terendah dan hanya berupa kode biner yaitu 0 dan 1. Sekumpulan instruksi dalam bahasa mesin dapat membentuk *microcode* (semacam prosedur dalam bahasa mesin).

Contoh :

Untuk mesin IBM/370

0001100000110101 = 1835

Yang berarti mengkopikan isi register 5 ke register 3

Keuntungan dari sebuah bahasa mesin yaitu eksekusi yang cepat, dan adapun kerugiannya adalah sangat sulit untuk dipelajari oleh manusia.

2) Bahasa *Assembly* (Memonic Code)

Merupakan bentuk simbolik dari sebuah bahasa mesin, dianggap sebagai bahasa pemrograman yang pertama kali berbentuk string dan lebih mudah dimengerti manusia. Setiap kode bahasa mesin memiliki simbol sendiri dalam bahasa *assembly*.

Misalnya ADD untuk penjumlahan, MUL untuk perkalian, SUB untuk pengurangan, dan lain-lain.

Sekumpulan kode-kode bahasa *assembly* dapat membentuk *mikroinstruksi*. Bahasa *assembly* juga memiliki program pendebug-nya, tidak seperti bahasa mesin. Misalnya : Turbo *Assembler* dan debug pada DOS. *Assembler* akan mencocokan token simbol dari awal hingga akhir, kemudian dikodekan menjadi sebuah bahasa mesin. Adapun kelebihan dari sebuah bahasa *assembly* adalah eksekusi yang cepat,

masih bisa dan dapat dipelajari dibandingkan dengan bahasa mesin, dan file yang dihasilkan sangat kecil. Sedangkan kekurangan nya adalah tetep sulit dipelajari oleh kebanyakan manusia, dan program yang sangat panjang.

3) Bahasa tingkat tinggi (*High Level Language*) / *user oriented*

Bahasa ini lebih dekat dekat manusia. Bahasa ini juga dapat memberikan banyak sekali fasilitas dan kemudahan dalam pembuatan program, misalnya : variabel, tipe data, konstanta, struktur kontrol, loop, fungsi, prosedur dan lain-lain. Contoh bahasa tingkat tinggi adalah : Pascal, Basic, C++, dan Java. mendukung information hiding, enkapsulasi, dan abstract data type.

Bahasa tingat tinggi memiliki generasi, misalnya generasi ke-3 (Pascal, C/C++) dan generasi ke-4 (Delphi, VB, VB.NET,Visual Foxpro)

Adapun beberapa keuntungan penggunaan bahasa tingkat tinggi ialah mudah dipelajari, mendekati permasalahan yang akan dipecahkan, dan kode program yang pendek. Salah satu kerugian dalam penggunaan bahasa ini yaitu eksekusi yang lambat.

4) Bahasa yang berorientasi pada masalah spesifik (*spesific problem oriented*).

Bahasa ini adalah bahasa yang digunakan langsung untuk memecahkan suatu masalah tertentu. Contoh : SQL untuk aplikasi database, COGO untuk aplikasi teknik sipil, Regex untuk memcocokan pola pada string tertentu, dan MatLab untuk matematika, dll. Bahasa problem oriented terkadang digolongkan kedalam bahasa tingkat tinggi.

Mesin pengenal bahasa

Tabel 2.1 Mesin Pengenal Bahasa

Kelas Bahasa	Mesin Pengenal Bahasa
Unrestricted Grammar (UG)	Mesin Turing (*Turing Machine*), TM
Context Sensitive Grammar (CSG)	*Linear Bounded Automaton*, LBA
Context Free Gammar (CFG)	Automata Pushdown (*Pushdown Automaton*), PDA
Regular Grammar, RG	Automata Hingga (*Finite Automaton*)

3. Himpunan Dan Relasi

a. Himpunan

Himpunan adalah kumpulan objek yang memiliki properti yang didefinisikan dengan jelas, atau lebih tepatnya, semua koleksi atau kumpulan objek tertentu yang memiliki kesamaan dan tergabung dalam sebuah unit.

contoh: kumpulan dari 4 huruf a, b, c, dan d termasuk sebuah himpunan, yang ditulis sebagai berikut:

H = {a,b,c,d}

untuk menyatakan bahwa a adalah anggota himpunan H, kita tulis a $\subset$ H, sedangkan y bukan anggota himpunan H, kita tulis y $\subseteq$ H.

Metode Penulisan Asosiasi $\in$ Mendaftarkan semua anggota

Contoh:

A = {a,e,i,o,u}

B = {2,3,5,7,11,13,17,19}

$\in$ = Menyatakan sifat yang dimiliki anggotanya.

Contoh:

A = Himpunan vokal dalam abjad latin

B = Himpunan bilangan prima yang kurang dari 20 Menggunakan notasi pembentuk himpunan

Contoh:

P = {x | x himpunan bilangan asli antara 7 dan 15}

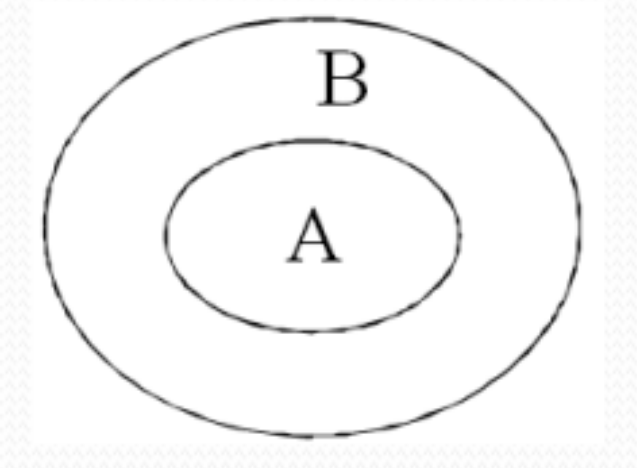
(Maksudnya P = {8,9,10,11,12,13,14})

Q = { t | t biangan asli}

(Maksudnya Q = {1,2,3,4,5,6,7,8,9,10,…}

Himpunan bagian (subset) A adalah himpunan bagian dari B jika A terkandung dalam B. A dan B mungkin merupakan himpunan yang sama. Hubungan suatu himpunan bagian lain disebut sebagai “termasuk dalam” atau “memuat”. Himpunan B adalah himpunan dari A karena A juga merupakan elemen B.

Contoh : A = {1,3,5} dan B = {0,1,2,3,4,5,6}. Maka A⊆B

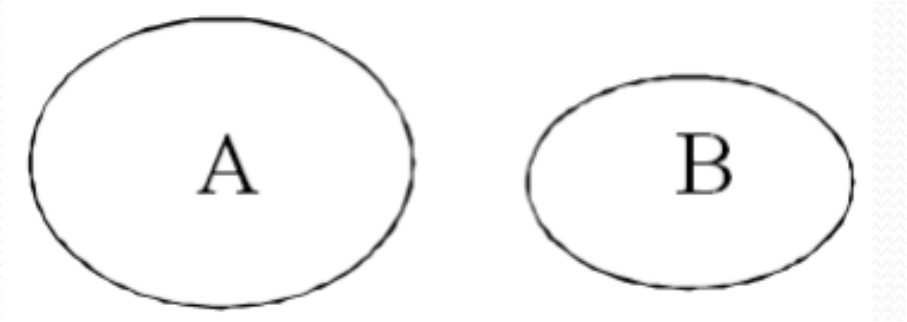


Gambar 2.1. Himpunan Bagian Subset

Himpunan disjoint adalah ketika dua atau lebih himpunan tidak ada anggota himpunan yang sma. Himpunan yang tidak memiliki anggota yang sama disebut sebagai himpunan eksklusif yang saling menguntungkan. Dua

himpunan dikatakan disjoint, jika himpunan tersebut tidak memiliki elemen yang sama.

Contoh : A = {1,2,3} dan B = {5,6}. Maka A B = ∅



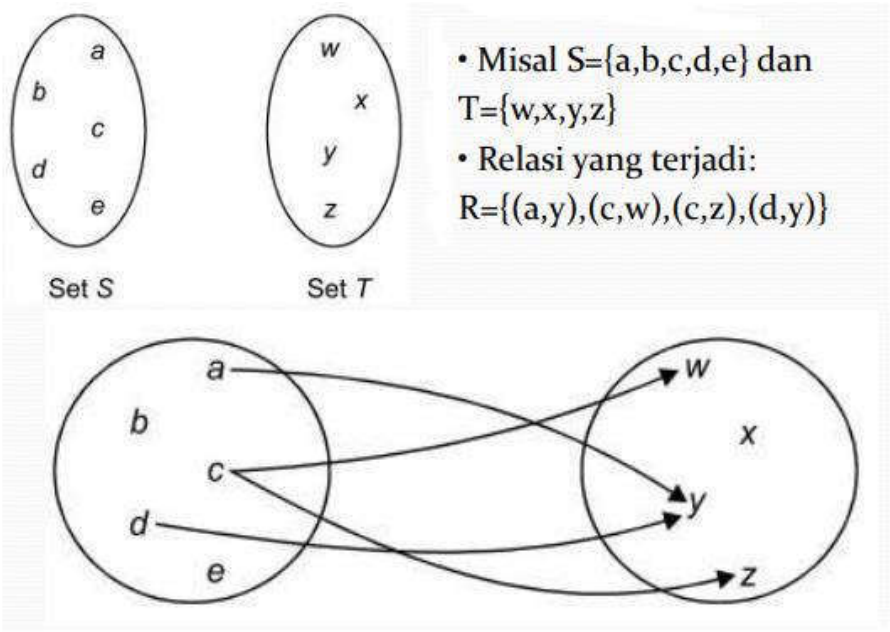
Gambar 2.2 Himpunan Disjoint

Himpunan Kosong adalah himpunan yang terdiri dari data kosong atau himpunan yang tidak memiliki elemen himpunan, himpunan kosong dilambangkan dengan "∅" atau { } ∅ = {} S ∪ ∅ = S S ∩ ∅ = ∅ ∅ = Universal set S - ∅ = S ∅ - S = ∅.

b. Relasi

Relasi adalah hubungan antar anggota himpunan dengan anggota himpunan lainnya. Cara termudah untuk menunjukkan hubungan antara anggota-anggota dalam sebuah himpunan adalah dengan himpunan pasangan yang saling urut. Himpunan pasangan tersebut diperoleh dengan perkalian kartesius.

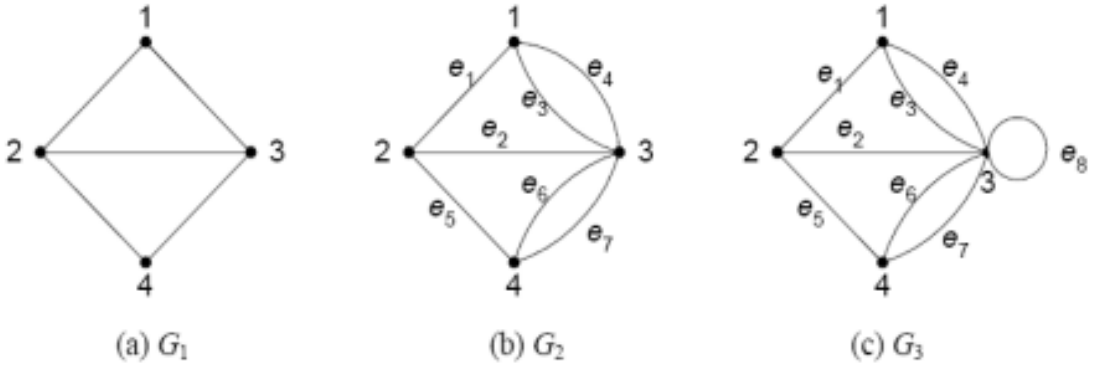
Relasi antar himpunan S dan T adalah himpunan dari pasangan berurutan (s,t) dimana: s ∈ S (s Anggota dari S) t ∈ T. Himpunan dari elemen pertama di sebut "DOMAIN" dari relasi. Himpunan dari elemen kedua disebut "RANGE" dari relasi.



Gambar 2.3 Relasi Antar Himpunan

Secara matematis, graph didefinisikan sebagai berikut: Graph G didefinisikan sebagai pasangan himpunan (V,E) yang dalam hal ini: V = himpunan tidak-kosong dari simpul-simpul (vertices atau node) = {v1,v2, ... , vn} E = himpunan sisi (edges atau arcs) yang menghubungkan sepasang simpul = {e1,e2, ..., en} atau dapat ditulis singkat notasi G = (V,E) Menyatakan bahwa V tidak boleh kosong, sedangkan E boleh kosong. Jadi, sebuah graph dimngkinkan tidak mempunyai sisi satu buah pun. Tetapi simpulnya harus ada, minimal satu. Graph yang hanya mempunyai satu buat simpul tanpa sebuah sisi pun dinamakan graf trivial.

Simpul pada graph dapat dinomori dengan huruf seperti a, b, c, ..., dengan bilangan asli 1, 2, 3, ..., atau gabungan keduanya. Sedangkan sisi yang menghubungkan simpul vi dengan simpul vj dinyatakan dengan pasangan (vi, vj) atau dengan lambang e1, e2, ... Dengan kata ain jika e adalah sisi yang menghubungkan simpul vi dengan vj, maka e dapat ditulis sebagai: e = (v1,vj)



Gambar 2.4 Tiga buah graph (a) graph sederhana, (b) graph ganda dan (c) graph semu

c. Otomata

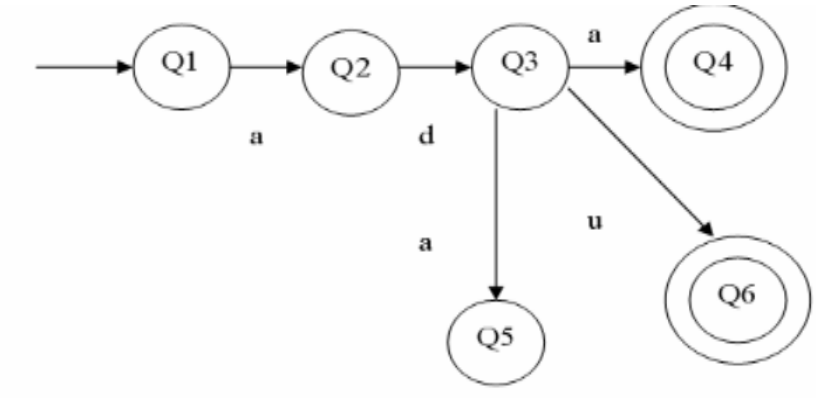
Untuk memodelkan hardware dari komputer diperkenalkanlah otomata / automata. Otomataadalah suatu sistem yang memiliki fungsi-fungsi komputer digital.

Karaketeristik Otomata :

1) Menerima input.
2) Menghasilkan output.
3) Mempunyai penyimpanan sementara (buffer)
4) Mampu membuat keputusan dalam mentransformasikan input ke output.

Otomata merupakan suatu sistem yang terdiri atas sejumlah berhingga state, dimana setiap state menyatakan informasi tentang input sebelumnya, dan dapat dianggap sebagai memori mesin.

Contoh:



Gambar 2.5 Siklus Otomata

Jika mesin mendapat string :

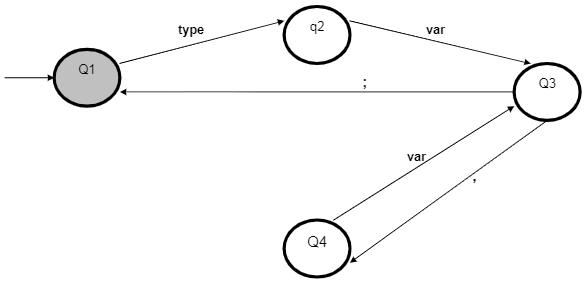
1) “ada” maka diterima
2) “adu” maka diterima
3) “add” maka ditolak

Aturan: input string diterima jika dan hanya jika mencapai state akhir yang disimbolkan dengan lingkaran ganda.

Keterangan:

1) Mesin diatas memiliki 6 state {q1,q2,q3,q4,q5,q6}.
2) Mesin memiliki state awal q1
3) Mesin memiliki state akhir {q4,q6}
4) Mesin memiliki himpunan input contohnya : {a,d,d}

Contoh pada bahasa C: int a,b;



Gambar 2.6 Otomata pada bahasa C

State awal: Q1, State akhir: Q1

C. SOAL LATIHAN/TUGAS

1. Carilah pengertian notasi grammar selain yang telah dijelaskan dalam modul. Buatlah kesimpulan berdasarkan definisi tersebut dengan bahasa Anda sendiri!
2. jelaskan konsep dan notasi bahasa menurut anda dan berikan contoh nya!
3. Jelaskan himpunan dan relasi dan berikan contoh!

D. REFERENSI

Practice And Principles Of Compiler Building With C, Henk Alblas, Albert Nymeyer, Prentice Hall, 1996

Introduction To The Theory Of Computation, Michael Sipser, PWS Publishing Company, 1997

The Essence Of Compilers, Robin Hunter, Prentice Hall Europe, 1999

Modern Compiler Design, Dick Grune, Henri E. Bal, Et All, John Wiley & Son, 2000

GLOSARIUM

microcode adalah menerjemahkan instruksi yang diterima CPU ke dalam operasi tingkat sirkuit fisik yang terjadi di dalam CPU.

makroinstruksi adalah proses penerjemahan dan eksekusi dari setiap instruksi prosesor menjadi urutan instruksi yang lebih kecil mikro

Graph adalah objek dasar pelajaran dalam teori graf. Dalam bahasa sehari-hari, sebuah graf adalah himpunan dari objek-objek yang dinamakan titik, simpul, atau sudut dihubungkan oleh penghubung yang dinamakan garis atau sisi