FILOSOFI STRUKTUR DATA

1. Pengantar

Sebelum kita masuk ke materi utama struktur data, maka alangkah baiknya jika kita menerangkan filosofi dari struktur data itu sendiri. Filosofi yang dimaksud di sini adalah yang mencakup:

- Pengertian berdasarkan nama mata kuliah
- Pengertian mengenai tujuan dari mata kuliah ini
- Pengertian hal-hal yang dibutuhkan untuk mempelajari materi ini
- Pengertian mengenai hubungannya dengan materi-materi lain.

Struktur dapat diartikan dengan ”susunan”, ”bangunan”, ”komposisi”, dan sebagainya. Kata struktur juga mengartikan bahwa elemen-elemen pembentuk ”susunan”, ”bangunan” dan ”komposisi” di atas saling terkait sebagaimana jika kita mengartikan kata ”sistem.”

Kata “*data*” dalam bahasa Inggris berasal dari kata “*datum*” dari bahasa Latin yang berarti fakta. Kata tersebut bersifat **plural**, sebagaimana kata air, udara, dan semacamnya. Karenanya, kata “data” akan salah jika disebut atau ditulis dengan “data-data,” “banyak data,” dan semacamnya.

Bagi manusia, data dapat merupakan segala sesuatu (stimulus) yang dapat ditangkap oleh indera manusia. Berbeda dengan manusia, data bagi komputer adalah segala sesuatu yang dapat dilambangkan, dikodekan, atau didigitalisasikan ke dalam lambang-lambang atau kode-kode yang dimengerti oleh komputer.

> *Data is information that has been translated into a form that is more convenient to move or process. Relative to today's computers and transmission media, data is information converted into binary digital form.*
> (http://searchstorage.techtarget.com/sDefinition/0,,sid5_gci211894,00.html)

Di komputer, secara kasar dapat dikatakan bahwa data dapat berupa angka-angka, huruf-huruf, gambar-gambar, atau simbol-simbol apapun yang dapat diberikan (input) ke komputer, dan dikeluarkan (*output*) dari komputer. Karena komputer itu benda mati yang tidak memiliki kemampuan apapun, termasuk kemampuan untuk mengenali mana huruf, mana angka, mana data, mana informasi, dan sebagainya, maka berikut ini penjelasan mendasar tentang bagaimana data bagi komputer itu “diciptakan” oleh daya nalar manusia.

2. *Bit* (*Binary Digit*)

Bit merupakan unit terkecil informasi di komputer, atau dapat disebut *bit* adalah satuan data terkecil di komputer digital. Istilah ini digunakan oleh John W. Tukey (1946), dan Claude E. Shannon (1948). Setiap *bit* hanya dapat bernilai sebuah dari dua buah nilai, 0 atau 1, tidak ada bilangan lain.

Bit adalah sebuah *digit* dari sistem bilangan binar (*binary numeral system*), yaitu sistem bilangan yang berbasis 2. *Binary digits* ini hampir selalu digunakan sebagai dasar perhitungan kemampuan menampung pada media penyimpan data (*storage*),

perhitungan secara *digital* dan pembelajaran teori informasi secara *digital*. Sebagai contoh, kemampuan transfer data dari sebuah jaringan komputer dihitung berdasarkan *bit per second* (bps), atau prosesor komputer yang digunakan oleh komputer X adalah prosesor 32 bit. Pada kemampuan grafis di monitor, setiap titik (*dot*) akan direpresentasikan oleh banyaknya *bit* yang digunakan. Monitor *monochrome* menggunakan 1 *bit*, sedangkan yang menggunakan 8 *bit* bisa menghasilkan 256 warna atau disebut dengan *grayscales*, dan yang menggunakan 24 atau 32 *bit*, dapat menghasilkan grafis yang sempurna (*true color*).

Sebuah *bit* dari *storage* adalah laksana sebuah saklar lampu (*light switch*) yang bisa dihidupkan dan dimatikan. Bila saklar dihidupkan (*on*) dilambangkan dengan 1, dan bila saklar dimatikan (*off*) dilambangkan dengan 0. Dua perbedaan yang jelas, hitam atau putih, benar atau salah, yang membuat Gregory Bateson mendefinisikan sebuah *bit* adalah "*a difference that makes a difference*."

Tentu saja, dunia ini yang bersifat analog (sebagai contoh, di antara 0 dan 1 saja terdapat sejumlah bilangan yang tidak terhingga) menjadi begitu menarik dipelajari (di ilmu komputer), agar, bagaimana (analog tadi) dijadikan digital (hanya ada 0 dan 1) saja.

> *A bit refers to a digit in the binary numeral system (base 2). For example, the number 10010111 is 8 bits long. Binary digits are almost always used as the basic unit of information storage and communication in digital computing and digital information theory.* (http://en.wikipedia.org/wiki/Bit)

Jadi, karena komputer tidak memiliki kemampuan apapun, maka untuk "memperkenalkan" data kepada komputer adalah dengan membuat rangkaian digital (elektronis) yang, bila dialiri arus listrik (sebesar +3,3 Volt atau +5,0 Volt) akan dilambangkan dengan 1 (*on*), dan bila tidak dialiri listrik (0 Volt) akan dilambangkan dengan 0 (*off*).

3. *Byte*

Untuk lebih memberi arti, bit di atas selanjutnya digabung (dikombinasikan nilai-nilainya) dan saling bertalian (*correspondence*) yang disebut dengan *byte*. Sederhananya, kumpulan *bit* yang membentuk sebuah informasi disebut dengan *byte*. Istilah *byte* juga digunakan sebagai satuan terkecil alamat (*address*) di mikroprosesor.

Sebelum istilah *byte* tersebut muncul, dulu dinamakan dengan "*bite,*" karena hampir mirip dengan kata "*bit*", maka oleh Werner Buchholz (1957) mulai digunakan istilah *byte* pada fase awal mendesain komputer IBM Stretch. *Byte* merupakan kependekan dari **B**inar**y** **T**upl**e**, namun beberapa sumber mengatakan bahwa *byte* merupakan kependekan dari **B**inar**y** **T**abl**e**. Tidak perlu diperdebatkan.

Masalahnya, berapa banyak *bit* penggabungan itu dilakukan. Pada umumnya, sebuah *byte* terdiri atas 8 *bit* yang disebut dengan *octet* yang dapat merepresentasikan 256 nilai (dari perhitungan : 2^8 dengan nilai 0 sampai 255). Seperti halnya standar yang digunakan untuk komputer IBM System/360. Ada juga yang menggunakan 4 *bit* (disebut *nibble*, *nybble, semioctet,* atau *hex digit*), ada pula yang menggunakan 2 *bit* (disebut *crumb*).

Istilah lain selain *byte* yang digunakan dari sekumpulan *bit* adalah kata (*word*). Hanya, pada *word* tidak ada standar besaran banyaknya *bit*. Besaran itu tergantung dari ukuran sebuah *register* di dalam CPU (*Central Processing Unit*) komputer. Sebagai contoh, di dalam arsitektur komputer IA-32 (prosesor Intel 8086) digunakan 16 *bit* untuk

sebuah *word*, sehingga 32 *bit* disebut dengan *double word* atau *dword*. Ada juga arsitektur komputer lain yang menyatakan sebuah *word* terdiri atas *bit* sebanyak 4, 8, 32, 64, dan sebagainya.

Dalam hal lain, ada standar penyebutan untuk ukuran *bit* yang besar, misalkan *kilobit* (Kbit), *megabit* (Mbit), *gigabit* (Gbit), dan sebagainya. Di ilmu komputer, *byte* juga digunakan sebagai ukuran dari *storage* (tempat menyimpan data), dan dijadikan dasar dari penetapan tipe data di berbagai bahasa pemrograman. Tipe data itu antara lain, *numeric* (dan lebih spesifik lagi *integer* atau *real*), *character* atau *string*, *boolean*, dan sebagainya.

> *A byte is a collection of bits, originally variable in size but now almost always eight bits. Eight-bit bytes, also known as octets, can represent 256 values (2^8 values, 0 - 255). A four-bit quantity is known as a nibble, and can represent 16 values (2^4 values, 0 - 15)* (http://en.wikipedia.org/wiki/Bit).

Pada Tabel 1. diperlihatkan penamaan dari ukuran *byte* :

Tabel 1.
Ukuran *byte*

Besaran dari *byte*				
SI prefixes			Binary prefixes	
Nama (Simbol)	Pemakaian Umum	Standar SI	Nama (Simbol)	Nilai
kilobyte **(K**B)	2^{10}	10^3	*kibibyte* **(KiB)**	2^{10}
megabyte **(MB)**	2^{20}	10^6	*mebibyte* **(MiB)**	2^{20}
gigabyte (GB)	2^{30}	10^9	*gibibyte* **(GiB)**	2^{30}
terabyte **(TB)**	2^{40}	10^{12}	*tebibyte* **(TiB)**	2^{40}
petabyte **(PB)**	2^{50}	10^{15}	*pebibyte* **(PiB)**	2^{50}
exabyte **(EB)**	2^{60}	10^{18}	*exbibyte* **(EiB)**	2^{60}
zettabyte **(ZB)**	2^{70}	10^{21}	*zebibyte* (ZiB)	2^{70}
yottabyte **(YB)**	2^{80}	10^{24}	*yobibyte* **(YiB)**	2^{80}
SI-prefix (juga dikenal sebagai metrix prefix adalah suatu asosiasi yang menentukan ukuran suatu simbol, dengan nama asli Système International d'Unités (dari Perancis). Binary prefix adalah simbol yang ditetapkan oleh International Electrotechnical Commision.				

Dalam pemberian nama singkatan dari besaran data antara "*bit*" dan "*byte*" juga kadang membingungkan. Oleh beberapa badan standardisasi, misalkan IEEE 1541 dan Metric-Interchange-Format, sepakat untuk "*byte*" digunakan huruf "B" besar, seperti MB untuk *megabyte*. Sedangkan untuk "*bit,*" IEEE 1541 menggunakan "b" huruf kecil, tetapi Metric-Interchange-Format dan IEC 60027 menggunakan kata yang lengkap "bit" jadi IEEE 1541 menuliskan Mb untuk megabit, sedangkan Metric-Interchange-Format dan IEC 60027 menuliskan dengan Mbit.

Pada Tabel 2. berikut adalah duapuluh ukuran dari SI prefixes.

Pengkodean *Byte*

Satu *byte* dapat dikatakan sebagai sebuah karakter (seperti sebuah huruf, sebuah angka, atau sebuah tanda baca). Tetapi, karena satu *byte* merupakan sekumpulan dari

bit, maka, kombinasi yang seperti apa dari *bit* yang akan membentuk sebuah *byte* ?. Tentu saja, hal ini memerlukan kesepakatan dari beberapa pengguna (pihak yang terkait).

Salah satunya, menghasilkan kesepakatan yang memunculkan kode ASCII (American Standard Code for Information Interchange) yaitu sistem pengkodean yang berbasis pada alpabet Inggris. ASCII disepakati pada tahun 1964 oleh American Standard Assosiation, dan dipublikasi sebagai standar pada tahun 1967, dan terakhir kali dimodifikasi pada 1986.

Tabel 2.
Duapuluh ukuran dari SI prefixes.

SI prefixes						
1000^n	10^n	Prefix	Symbol	Short scale	Long scale	Decimal equivalent in SI writing style
1000^8	10^{24}	yotta	Y	Septillion	Quadrillion	1 000 000 000 000 000 000 000 000
1000^7	10^{21}	zetta	Z	Sextillion	Trilliard	1 000 000 000 000 000 000 000
1000^6	10^{18}	exa	E	Quintillion	Trillion	1 000 000 000 000 000 000
1000^5	10^{15}	peta	P	Quadrillion	Billiard	1 000 000 000 000 000
1000^4	10^{12}	tera	T	Trillion	Billion	1 000 000 000 000
1000^3	10^9	giga	G	Billion	Milliard	1 000 000 000
1000^2	10^6	mega	M	Million		1 000 000
1000^1	10^3	kilo	k	Thousand		1 000
$1000^{2/3}$	10^2	hecto	h	Hundred		100
$1000^{1/3}$	10^1	deca	da	Ten		10
1000^0	10^0	*(none)*	*(none)*	One		1
$1000^{-1/3}$	10^{-1}	deci	d	Tenth		0.1
$1000^{-2/3}$	10^{-2}	centi	c	Hundredth		0.01
1000^{-1}	10^{-3}	milli	m	Thousandth		0.001
1000^{-2}	10^{-6}	micro	µ (u)	Millionth		0.000 001
1000^{-3}	10^{-9}	nano	n	Billionth	Milliardth	0.000 000 001
1000^{-4}	10^{-12}	pico	p	Trillionth	Billionth	0.000 000 000 001
1000^{-5}	10^{-15}	femto	f	Quadrillionth	Billiardth	0.000 000 000 000 001
1000^{-6}	10^{-18}	atto	a	Quintillionth	Trillionth	0.000 000 000 000 000 001
1000^{-7}	10^{-21}	zepto	z	Sextillionth	Trilliardth	0.000 000 000 000 000 000 001
1000^{-8}	10^{-24}	yocto	y	Septillionth	Quadrillionth	0.000 000 000 000 000 000 000 001

ASCII mengkodekan sebuah *byte* atas 7 *bit (seven-bit code)*, sehingga banyak karakter yang dikodekan adalah 2^7 = 128 buah (nilainya dari 0 sampai 127 desimal), 33 buah merupakan karakter kontrol (*non printable*), dan 95 buah karakter umum (*printable characters*). Contoh karakter kontrol misalkan penekanan pada tombol Enter (Carriage Return), Esc, Delete, dan sebagainya (yang tidak tercetak). Karakter kontrol di ASCII bernilai dari 0 hingga 31 desimal dan 177 desimal. Lihat di Tabel 3.

Pada Tabel 4, dapat dilihat *printable characters* dari kode ASCII. Dapat dilihat bahwa pada nilai desimal (Des) 32 adalah spasi (meski tidak tampak ketika dicetak, tetapi ini termasuk karakter yang dapat dilihat ketika dicetak, yaitu karakter kosong).

Penggunaan kode ASCII ini banyak dijumpai di komputer-komputer *desktop* di Indonesia ini (yang menggunakan sistem operasi DOS atau Windows). Kita bisa menampilkan kode ASCII dengan menekan tombol Alt dan angka yang ada di *keypad-number*, misalkan kita tekan tombol Alt dan angka 65, maka akan tampil huruf A.

Kini, standar kode ASCII yang menggunakan 7-*bit* telah diperbaharui dengan menggunakan 8-*bit* sehingga banyaknya kombinasi karakter yang bisa dikodekan adalah 2^8 = 256 (0 sampai 255). Perluasan itu disebut dengan *Extended* ASCII atau *High* ASCII, karakter yang ditambahkan berupa karakter matematis, grafik, dan tambahan spesial karakter lainnya. Agar terdapat kesinambungan dengan tabel ASCII sebelumnya, maka nilai standar ASCII tidak diubah, penambahan dilakukan pada nilai 128 sampai 255.

.

Tabel 3.
Tombol kontrol di ASCII

Binary	Oct	Dec	Hex	Abbr	PR	CS	CEC	Description
0000 0000	000	0	00	NUL	NUL	^@	\0	*Null character*
0000 0001	001	1	01	SOH	SOH	^A		*Start of header*
0000 0010	002	2	02	STX	STX	^B		*Start of text*
0000 0011	003	3	03	ETX	ETX	^C		*End of text*
0000 0100	004	4	04	EOT	EOT	^D		*End of transmission*
0000 0101	005	5	05	ENQ	ENQ	^E		*Enquiry*
0000 0110	006	6	06	ACK	ACK	^F		*Acknowledgment*
0000 0111	007	7	07	BEL	BEL	^G	\a	*Bell*
0000 1000	010	8	08	BS	BS	^H	\b	*Backspace*
0000 1001	011	9	09	HT	HT	^I	\t	*Horizontal Tab*
0000 1010	012	10	0A	LF	LF	^J	\n	*Line feed*
0000 1011	013	11	0B	VT	VT	^K	\v	*Vertical Tab*
0000 1100	014	12	0C	FF	FF	^L	\f	*Form feed*
0000 1101	015	13	0D	CR	CR	^M	\r	*Carriage return*
0000 1110	016	14	0E	SO	SO	^N		*Shift Out*
0000 1111	017	15	0F	SI	SI	^O		*Shift In*
0001 0000	020	16	10	DLE	DLE	^P		*Data Link Escape*
0001 0001	021	17	11	DC1	DC1	^Q		*Device Control 1 (oft. XON)*
0001 0010	022	18	12	DC2	DC2	^R		*Device Control 2*
0001 0011	023	19	13	DC3	DC3	^S		*Device Control 3 (oft. XOFF)*
0001 0100	024	20	14	DC4	DC4	^T		*Device Control 4*
0001 0101	025	21	15	NAK	NAK	^U		*Negative*

Binary	Oct	Dec	Hex	Abbr	PR	CS	CEC	Description
								Acknowledgement
0001 0110	026	22	16	SYN	SYN	^V		*Synchronous Idle*
0001 0111	027	23	17	ETB	ETB	^W		*End of Trans. Block*
0001 1000	030	24	18	CAN	CAN	^X		*Cancel*
0001 1001	031	25	19	EM	EM	^Y		*End of Medium*
0001 1010	032	26	1A	SUB	SUB	^Z		*Substitute*
0001 1011	033	27	1B	ESC	ESC	^[	\e	*Escape*
0001 1100	034	28	1C	FS	FS	^\		*File Separator*
0001 1101	035	29	1D	GS	GS	^]		*Group Separator*
0001 1110	036	30	1E	RS	RS	^^		*Record Separator*
0001 1111	037	31	1F	US	US	^_		*Unit Separator*
0111 1111	177	127	7F	DEL	DEL	^?		*Delete*

Perluasan itu dibedakan antara untuk sistem operasi DOS dan sistem operasi Windows. Kode 8-*bit* ASCII ini digunakan di komputer IBM-PC dan yang kompatibel dengannya. Yang dikatakan kompatibel adalah komputer yang dirancang berdasarkan desain IBM-PC asli dari perusahaan International Business Machines (IBM). Mereka menggunakan produk Intel dengan arsitektur x86. Kini, yang dikatakan kompatibel bisa berarti "dapat dioperasikan dengan system operasi Windows yang terbaru."

Di Tabel 5. ditampilkan tabel perluasan tersebut (untuk yang menggunakan sistem operasi DOS). Selain perluasan, ada pula beberapa varian dari kode ASCII (dengan perubahan beberapa kode *printable*-nya), antara lain : ATASCII (Atari Standard Code for Information Interchange), dan YUSCII (Yugoslav Standard Code for Information Interchange).

Selain ASCII, ada banyak standar lain yang ditetapkan. Mengapa tidak disepakati saja untuk semua pelaku pembuat teknologi informasi dan komunikasi?, tentu ada banyak alasan, satu yang mungkin adalah alasan persaingan merebut pasar. Dua standar kode lain di antaranya adalah BCD (*Binary-Coded Decimal*), EBCDIC (*Extended Binary-Coded Decimal Interchange Code*).

BCD mengkodekan sebuah *byte* dengan 6 karakter, dan EBCDIC menggunakan 8-*bit* untuk mengkodekan satu buah karakternya. Standar EBCDIC ini biasa digunakan di mesin IBM-*mainframe operating system*, seperti z/OS, OS/390, VM, dan VSE. Mini komputer IBM *operating system* seperti OS/400 dan i5/OS menggunakan kode ini juga.

Tabel 4.
Printable ASCII *Characters*

Binary	Dec	Hex	Glyph	Binary	Dec	Hex	Glyph	Binary	Dec	Hex	Glyph
0010 0000	32	20	SP	0100 0000	64	40	@	0110 0000	96	60	`
0010 0001	33	21	!	0100 0001	65	41	A	0110 0001	97	61	a
0010 0010	34	22	"	0100 0010	66	42	B	0110 0010	98	62	b
0010 0011	35	23	#	0100 0011	67	43	C	0110 0011	99	63	c
0010 0100	36	24	$	0100 0100	68	44	D	0110 0100	100	64	d
0010 0101	37	25	%	0100 0101	69	45	E	0110 0101	101	65	e
0010 0110	38	26	&	0100 0110	70	46	F	0110 0110	102	66	f
0010 0111	39	27	'	0100 0111	71	47	G	0110 0111	103	67	g
0010 1000	40	28	(	0100 1000	72	48	H	0110 1000	104	68	h
0010 1001	41	29	)	0100 1001	73	49	I	0110 1001	105	69	i
0010 1010	42	2A	*	0100 1010	74	4A	J	0110 1010	106	6A	j
0010 1011	43	2B	+	0100 1011	75	4B	K	0110 1011	107	6B	k
0010 1100	44	2C	,	0100 1100	76	4C	L	0110 1100	108	6C	l
0010 1101	45	2D	-	0100 1101	77	4D	M	0110 1101	109	6D	m
0010 1110	46	2E	.	0100 1110	78	4E	N	0110 1110	110	6E	n
0010 1111	47	2F	/	0100 1111	79	4F	O	0110 1111	111	6F	o
0011 0000	48	30	0	0101 0000	80	50	P	0111 0000	112	70	p
0011 0001	49	31	1	0101 0001	81	51	Q	0111 0001	113	71	q
0011 0010	50	32	2	0101 0010	82	52	R	0111 0010	114	72	r
0011 0011	51	33	3	0101 0011	83	53	S	0111 0011	115	73	s
0011 0100	52	34	4	0101 0100	84	54	T	0111 0100	116	74	t
0011 0101	53	35	5	0101 0101	85	55	U	0111 0101	117	75	u
0011 0110	54	36	6	0101 0110	86	56	V	0111 0110	118	76	v
0011 0111	55	37	7	0101 0111	87	57	W	0111 0111	119	77	w
0011 1000	56	38	8	0101 1000	88	58	X	0111 1000	120	78	x
0011 1001	57	39	9	0101 1001	89	59	Y	0111 1001	121	79	y
0011 1010	58	3A	:	0101 1010	90	5A	Z	0111 1010	122	7A	z
0011 1011	59	3B	;	0101 1011	91	5B	[	0111 1011	123	7B	{
0011 1100	60	3C	<	0101 1100	92	5C	\	0111 1100	124	7C	|
0011 1101	61	3D	=	0101 1101	93	5D	]	0111 1101	125	7D	}
0011 1110	62	3E	>	0101 1110	94	5E	^	0111 1110	126	7E	~
0011 1111	63	3F	?	0101 1111	95	5F	_				

Tabel 5.
Extended ASCII *Characters*

Bin	Oct	Dec	Hex	Char	Bin	Oct	Dec	Hex	Char	Bin	Oct	Dec	Hex	Char	Bin	Oct	Dec	Hex	Char

Tabel 5.
Extended ASCII *Characters*

Bin	Oct	Dec	Hex	Char	Bin	Oct	Dec	Hex	Char	Bin	Oct	Dec	Hex	Char	Bin	Oct	Dec	Hex	Char
00000	200	128	80	Ç	00000	240	160	A0	á	00000	300	192	C0	└	00000	340	224	E0	α
00001	201	129	81	ü	00001	241	161	A1	í	00001	301	193	C1	┴	00000	341	225	E1	ß
00010	202	130	82	é	00010	242	162	A2	ó	00010	302	194	C2	┬	00000	342	226	E2	Γ
00011	203	131	83	â	00011	243	163	A3	ú	00011	303	195	C3	├	00000	343	227	E3	π
00100	204	132	84	ä	00100	244	164	A4	ñ	00100	304	196	C4	─	00000	344	228	E4	Σ
00101	205	133	85	à	00101	245	165	A5	Ñ	00101	305	197	C5	┼	00000	345	229	E5	σ
00110	206	134	86	å	00110	246	166	A6	ª	00110	306	198	C6	╞	00000	346	230	E6	µ
00111	207	135	87	ç	00111	247	167	A7	º	00111	307	199	C7	╟	00000	347	231	E7	τ
01000	210	136	88	ê	01000	250	168	A8	¿	01000	310	200	C8	╚	00000	340	232	E8	Φ
01001	211	137	89	ë	01001	251	169	A9	⌐	01001	311	201	C9	╔	00000	341	233	E9	Θ
01010	212	138	8A	è	01010	252	170	AA	¬	01010	312	202	CA	╩	00000	342	234	EA	Ω
01011	213	139	8B	ï	01011	253	171	AB	½	01011	313	203	CB	╦	00000	343	235	EB	δ
01100	214	140	8C	î	01100	254	172	AC	¼	01100	314	204	CC	╠	00000	344	236	EC	∞
01101	215	141	8D	ì	01101	255	173	AD	¡	01101	315	205	CD	═	00000	345	237	ED	φ
01110	216	142	8E	Ä	01110	256	174	AE	«	01110	316	206	CE	╬	00000	346	238	EE	ε
01111	217	143	8F	Å	01111	257	175	AF	»	01111	317	207	CF	╧	00000	347	239	EF	∩
10000	210	144	90	É	10000	260	176	B0	░	10000	320	208	D0	╨	00000	340	240	F0	≡
10001	221	145	91	æ	10001	261	177	B1	▒	10001	321	209	D1	╤	00000	341	241	F1	±
10010	222	146	92	Æ	10010	262	178	B2	▓	10010	322	210	D2	╥	00000	342	242	F2	≥
10011	223	147	93	ô	10011	263	179	B3	│	10011	363	211	D3	╙	00000	343	243	F3	≤
10100	224	148	94	ö	10100	264	180	B4	┤	10100	324	212	D4	╘	00000	344	244	F4	⌠
10101	225	149	95	ò	10101	265	181	B5	╡	10101	325	213	D5	╒	00000	345	245	F5	⌡
10110	226	150	96	û	10110	266	182	B6	╢	10110	326	214	D6	╓	00000	346	246	F6	÷
10111	227	151	97	ù	10111	267	183	B7	╖	10111	327	215	D7	╫	00000	347	247	F7	≈
11000	230	152	98	ÿ	11000	270	184	B8	╕	11100	330	216	D8	╪	00000	340	248	F8	°
11001	231	153	99	Ö	11001	271	185	B9	╣	11101	331	217	D9	┘	00000	341	249	F9	·
11010	232	154	9A	Ü	11010	272	186	BA	║	11110	332	218	DA	┌	00000	342	250	FA	·
11011	233	155	9B	¢	11011	273	187	BB	╗	11111	333	219	DB	▉	00000	343	251	FB	√
11100	234	156	9C	£	11100	274	188	BC	╝	11100	334	220	DC	▄	00000	344	252	FC	ⁿ
11101	235	157	9D	¥	11101	275	189	BD	╜	11101	335	221	DD	▌	00000	345	253	FD	²
11110	236	158	9E	Pts	11110	276	190	BE	╛	11110	336	222	DE	▐	00000	346	254	FE	■
11111	237	159	9F	ƒ	11111	277	191	BF	┐	11111	337	223	DF	▀	00000	347	255	FF	

4. *Field/ Attribute* (Atribut)

Di *computer science*, data yang memiliki beberapa bagian dapat dibagi menjadi atribut. Sebagai contoh, data mahasiswa, dapat dibagi menjadi beberapa atribut yang berbeda, misalkan nama, alamat, tempat dan tanggal lahir, jenis kelamin, status, dan sebagainya. Jadi, atribut merupakan ciri atau karakteristik dari suatu data, dan ia menjadi bagian (sandangan) dari data tersebut. Atribut juga merupakan kumpulan dari

byte (karakter). Bukankah nama mahasiswa merupakan kumpulan dari huruf atau karakter ?.

Ada beberapa istilah lain untuk *field*, selain *attribute*, ada juga yang menyebut dengan *column* (kolom), *data member*, dan *variable* (variabel), baik *instance variable* maupun *class variable*. Mengapa bisa berbeda-beda istilahnya ?, karena bidang kajian (ilmu)nya juga berbeda-beda. Misalkan, jika kita membuat program (dalam bahasa pemrograman tertentu), akan memasukkan suatu nilai data (*data value*) 90 untuk nilai mata pelajaran matematika, maka, jika data itu hanya digunakan sementara, ia dimasukkan dalam variabel *internal memory* (biasa disebut dengan variabel saja).

NM = 90

NM adalah nama variabel yang disingkat saja (dari kata "nilai matematika"), dan 90 adalah nilai datanya. Jika komputer dimatikan, maka nilai itu akan terhapus dari *internal memory* komputernya (tetapi tidak terhapus dari programnya, jika program itu sudah disimpan ke dalam *external memory*, misalkan disket). Istilah variabel ini umumnya digunakan dalam bahasa pemrograman (yang programnya akan disimpan ke dalam sebuah *file* program di dalam *external memory*).

Jika nilai data itu disimpan ke dalam *external memory* dengan aturan tertentu (akan dibahas di sub-materi *database* setelah ini), maka ia dinamakan atribut atau *field*, Jadi, atribut atau apapun sebutannya di atas, adalah karakteristik dari suatu data, atau sekumpulan (bisa juga satu) karakter yang sudah memiliki arti.

Contoh sebuah atribut yang nilai datanya hanya satu karakter, bisa karakternya berupa huruf seperti "A", "B", dan sebagainya untuk atribut nilai (di Perguruan Tinggi). Bisa karakternya berupa angka, misalkan 1, 2, hingga 9 untuk atribut jumlah anak, dan sebagainya.Bisa juga nilai datanya berupa kode, misalkan "P" untuk nilai data dari atribut jenis kelamin, yang diartikan dengan pria, dan "W" untuk wanita. Dan sebagainya.

Penjelasan mendalam tentang atribut, akan dibahas di bab tentang *database*.

5. *Record/ Tuple* (Tupel)

Sekarang, mari kita pandang di suatu lingkungan (*enterprise*), misalkan di kampus.Di sana (di dalam kampus), kita melihat beberapa objek yang kasat mata, seperti, ada orang-orang, dan ada bangunan. Kita identifikasi lagi, siapa (statusnya sebagai apa) orang-orang yang ada di dalam kampus tersebut. Ternyata ada yang sebagai dosen, ada yang sebagai mahasiswa, ada yang sebagai orang-tua, ada yang sebagai supir, ada yang sebagai pegawai kantin, ada yang sebagai pegawai tata usaha, dan sebagainya.

Kita pilih status mereka (hanya bagi mereka yang benar-benar berkaitan langsung dengan proses belajar-mengajar di kampus). Kita dapatkan, mahasiswa, dosen, dan pegawai tata usaha. Kita perkecil lingkupnya, pertanyaannya sekarang adalah, apa saja atribut yang berhak dimiliki oleh seorang mahasiswa ?.

Tentu saja, jika kita pandang seorang mahasiswa, ia pasti punya identitas diri seperti nama, alamat rumah, tempat dan tanggal lahir, jenis kelamin, dan jika kita lihat dari penampilannya, tentulah ia memiliki atribut tinggi badan, berat badan, warna rambut, warna kulit, bentuk muka, dan sebagainya, masih banyak lagi.

Tentu, tidak semua atribut yang ia miliki lantas harus kita masukkan ke dalam pendataan kita di komputer. Cukuplah atribut-atribut yang sangat diperlukan dalam kegiatan belajar-mengajar saja yang perlu didata. Misalkan nama, alamat rumah, tempat

dan tanggal lahir, dan jenis kelamin. Tentu boleh ditambah lagi dengan identitas yang diperlukan, misalkan nomor induk mahasiswa (NIM), nomor telepon, dan sebagainya.

Satu rangkaian data identitas mahasiswa di atas (terdiri atas beberapa atribut yang dipilih) disebut dengan *record*.

6. *Data File*, *Entity Set*, *Object*, *Table*, atau Berkas Elektronik

Jika kita ingin menyimpan (*save*) data teman sekelas kita yang berjumlah 40 orang, maka kita akan memasukkan data tentang nama, alamat, kota dan tanggal kelahiran dari teman-teman kita itu ke dalam 40 *record*. Kumpulan *record* tersebut akan membentuk sebuah *data file* (biasa disebut dengan *file* saja). Dalam satuan *file* inilah, data bisa disimpan di media penyimpan elektronik *external memory* seperti disket, *hard disk*, *flash disk*, dan sebagainya.

Jadi, bila sebuah *file* kita gambarkan sebagai sebuah tabel, maka, *record* adalah barisnya (*row*), dan *field* adalah kolomnya (*column*). Selain *data file*, ada ribuan jenis *file* lain yang ada di sebuah *hard disk* misalnya. Kita bisa mengetahui sebuah *file* di *hard disk* merupakan jenis *file* apa, bisa kita lihat dari nama *file*-nya.

file name.extension file name

Umumnya, nama *file* terdiri atas dua bagian, yaitu nama *file* (*file name*), dan nama panjang dari *file* (*extension file name*). Nah, nama panjang dari *file* yang merupakan petunjuk jenis *file* apa dia. Misalkan, nama panjangnya adalah .BAS, maka ia merupakan program dari bahasa pemrograman BASIC, misalkan lagi .DBF, maka ia merupakan *data file* dari *software* dBase, dan sebagainya.

Seluruh *file* yang disimpan ke komputer, diatur oleh yang namanya *File systems* dan *File managers*. *File systems* adalah sebuah sistem (yang umumnya dibuat oleh pabriknya) yang menentukan cara, metode, atau prosedur komputer dalam mengorganisasi, memberikan (ketentuan) penamaan *file* (utamanya jika bila *user* tidak menuliskan nama panjang *file*nya), penyimpanan, dan pengaksesan *file* secara umum. Selah satu tujuan dari *file system* adalah agar para pengguna bisa kembali mendapatkan dan mengakses *file* yang pernah ia simpan, dengan mudah dan cepat. Fungsi lainnya adalah menjaga dan mengatur lokasi fisik dari *file* di media penyimpanannya (terhadap perubahan data yang dilakukan pemakai).

Ada dua *file managers* yang umum digunakan, yakni FAT pada sistem operasi MS-DOS (Microsoft Disk Operating System) tempo dulu, dan NTFS yang digunakan di sistem operasi Windows saat ini. Sedikit perbedaan yang jelas adalah, pada sistem FAT, nama *file* maksimum hanya bisa 8 karakter (ditambah dengan nama panjang sebanyak 3 karakter), tanpa spasi . Sedangkan di sistem NTFS, nama *file* bisa jauh lebih panjang (maksimum 255 karakter) dan boleh dengan spasi.

FAT (*File Allocation Table*), adalah metode yang digunaKan Microsoft *operating systems* untuk menjaga *track* dari isi sebuah *disk*. Di sini dibuat bagan yang menghubungkan ke *cluster addresses* di dalam sebuah *hard drive*. FAT juga mengalami berkali-kali perkembangan, mulai dari FAT12 (12-*bit*) yang hanya dapat menjangkau sekitar 16 MB (16,736,256) dari volume *hard disk*. Kalau hard disk kita sekarang berukuran *Gigabyte*, tentu tidak bisa, dianggap *hard disk drive is corrupted*, *bad or has a computer virus*. Selanjutnya diciptakan FAT16 (16-*bit*) dan digunakan pada masa Windows 3.x sampai Windows 95. FAT16, dan yang terakhir FAT32 (28-*bit*) yang digunakan pertama kali di Windows 95 hingga Windows 98.

Di Tabel 6. digambarkan karakteristik dari NTFS :

File manager (disebut juga dengan *file browser*) adalah program utilitas (*utility program*) yang menuntun kita dalam memanipulasi *file*. Kegiatan memanipulasi *file* antara lain : melakukan pemindahan *file* atau bahkan pemindahan *folder* dari satu *directory* (*folder*) ke *directory* (*folder*) lain, bisa di dalam satu media penyimpanan maupun ke media penyimpanan lain, membuat *file* (atau *folder*) baru, menghapus *file* (atau *folder*), mencetak, mengganti nama, meng*copy file*, tetapi ia tidak dibuat untuk dapat melihat isi *file* kita tanpa *software* pendukung lain.Contoh *file manager* untuk sistem operasi Windows adalah Windows Explorer (WE). WE dilengkapi pula dengan fasilitas untuk melihat/ mengubah *properties* dari *file*, mencari *file*, mengirim *file* ke *e-mail*, dan sebagainya.

Pembagian Jenis *Data File*

Data file sendiri dibagi menjadi dua jenis, yaitu *master file*, dan *transaction file*. Bila kita kembali mengingat ke jaman *baheula* (dan sekarang juga mungkin masih ada), jaman di mana belum dikenal komputer, pegawai tata usaha di suatu sekolah memiliki data siswa yang ditulis di Buku Induk Siswa. Buku tersebut disimpan di dalam rak yang tersedia. Nah, yang disebut dengan *master file* serupa dengan itu, tetapi tempat menulis data dan tempat penyimpanannya sudah di dalam komputer. Buku Induknya adalah *master file*, dan tempat menyimpannya adalah *database* (penjelasan mengenai *database* akan dibahas berikutnya).

Secara prinsip dapat kita jabarkan bahwa *master file* adalah catatan mengenai objek-objek yang **harus ada** di suatu lingkungan (*enterprise*) yang bila objek-objek itu tidak ada, maka lingkungan itu tidak dapat berfungsi sebagaimana mestinya. Lingkungan itu bisa berupa sebuah bank, sebuah sekolah, sebuah kantor, dan sebagainya, atau bisa diperkecil lagi (sebagian) dari itu, misalkan di perpustakaannya saja, di bagian Personalianya saja, dan sebagainya.

Seberapa besar dan kecilnya lingkungan, tergantung seberapa besar dan kecilnya bagian yang (akan) dikomputerisasi. Misalkan, di lingkungan perpustakaan saja, pertanyaannya adalah "*Master file* apa saja yang dibutuhkan di perpustakaan ?." Tentu jawabannya akan kita pikirkan dengan cara bertanya pada diri sendiri "Apa saja objek yang harus ada di suatu perpustakaan sehingga memang tempat itu pantas disebut sebuah perpustakaan."

Tabel 6.
Karakteristik dari NTFS

NTFS		Features	
Developer	Microsoft	**Dates recorded**	Creation, modification, POSIX change, access
Full name	New Technology File System	**Date range**	1 January 1601 - 28 May 60056
Introduced	July 1993 (Windows NT 3.1)	**Forks**	Yes
Partition identifier	0x07 (MBR) EBD0A0A2-B9E5-4433-87C0-	**Attributes**	Read-only, hidden, system, archive

NTFS		Features	
	68B6B72699C7 (GPT)		
Structures		**File system permissions**	ACLs
Directory contents	B+ tree	**Transparent compression**	Per-file, LZ77 (Windows NT 3.51 onward)
File allocation	B+ tree	**Transparent encryption**	Per-file, DESX (Windows 2000 onward), Triple DES (Windows XP onward), AES (Windows XP Service Pack 1, Windows Server 2003 onward)
Bad blocks	Bitmap/Extents	**Supported operating systems**	Windows NT family (Windows NT 3.1 to Windows NT 4.0, Windows 2000, Windows XP, Windows Server 2003, Windows Vista)
Limits			
Max file size	16 TiB with current implementation(16 EiB architecturally)		
Max number of files	4,294,967,295 (2^{32}-1)		
Max filename size	255 characters		
Max volume size	256 TiB with current implementation (16 EiB architecturally)		
Allowed characters in filenames	any character except '\0' (NULL) and '/'[11] Windows also excludes the use of \ : * ? " < > and pipe		

Jawaban pertama yang terpikir adalah “buku” (baik buku konvensional maupun buku elektronik), karena memang kalau di perpustakaan itu tidak ada buku, namanya bukan perpustakaan. Tetapi, apakah hanya dengan adanya buku, lantas suatu tempat pantas disebut perpustakaan ?, bagaimana dengan toko buku ?. Tentu, *master file* di perpustakaan bukan hanya buku, satu objek yang merupakan pembeda dengan toko buku adalah adanya peminjam atau anggota (di toko buku namanya pembeli). Objek lain yang merupakan *master file* adalah pegawai, rak penyimpanan buku, distributor buku, dan sebagainya.

> *Master file is a collection of records pertaining to one of the main subjects of an information system, such as customers, employees, products and vendors. Master files contain descriptive data, such as name and address, as well as summary information, such as amount due and year-to-date sales.*
>
> http://www.pcmag.com/encyclopedia_term/0,2542,t=master+file&i=46618,00.asp

Sedangkan *transaction file* adalah catatan mengenai transaksi-transaksi yang terjadi di lingkungan tersebut. Misalkan di perpustakaan di atas, maka transaksi yang terjadi adalah, bisa berupa “anggota meminjam buku,” maka data peminjaman tersebut harus dicatat. Bisa juga berupa transaksi “anggota mengembalikan buku,” atau “anggota mengganti buku,” atau “anggota membayar denda buku,” atau “distributor memberi buku,” dan sebagainya. Bisa kita tarik kesimpulan, bahwa *transaction file* adalah berelasinya (berhubungannya) satu atau lebih *master file*.

Apa contoh satu *master file* bisa berhubungan dengan dirinya sendiri untuk menghasilkan *transaction file* ?. Ada banyak, misalkan di lingkungan (*enterprise*) pemain sepak bola. Di sana terjadi transaksi pemilihan pemain yang akan menjadi kapten kesebelasan, maka transaksinya adalah “Pemain memilih kapten dari pemain.” Berbeda hal jika lingkungannya adalah sebuah klub sepak bola yang lengkap, ada pelatih, ada

manajer, ada penyandang dana, dan sebagainya. Bisa saja transaksinya menjadi "Pelatih menentukan kapten dari pemain."

> *Transaction file is a collection of transaction records. The data in transaction files is used to update the master files, which contain the data about the subjects of the organization (customers, employees, vendors, etc.). Transaction files also serve as audit trails and history for the organization. Where before they were transferred to offline storage after some period of time, they are increasingly being kept online for routine analyses.*
>
> http://www.pcmag.com/encyclopedia_term/0,2542,t=transaction+file&i=53075,00.asp

Master file itu sendiri terdiri atas dua jenis, yakni *reference master file* dan *dynamic master file*. Disebut dengan *reference master file* karena, jika terjadi suatu transaksi, nilai datanya tidak ada yang berubah sedikitpun (ia hanya dijadikan referensi saja), sedangkan disebut dengan *dynamic master file* karena, jika terjadi suatu transaksi, ada nilai datanya yang berubah.

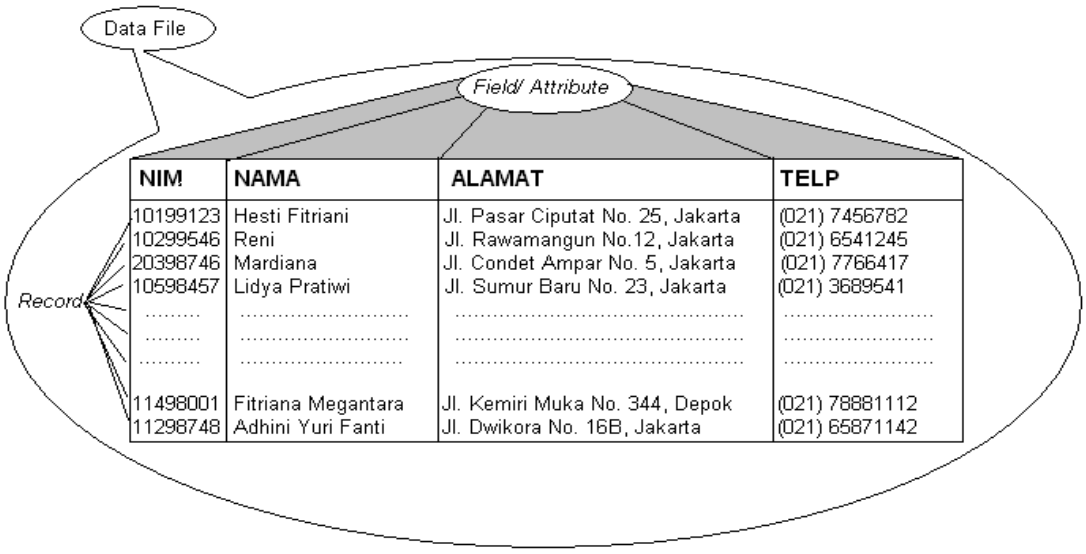
Kita kembali mengingat di lingkungan perpustakaan di atas, jika terjadi transaksi "**Anggota** meminjam **Buku**," ada atau tidak nilai data dari **Anggota** atau nilai data dari **Buku** yang berubah ?. Kita ingat bahwa nilai data adalah isi dari suatu atribut (*field*). Sepantas-nyalah setiap anggota memiliki *fields* : **No_Ang** (nomor anggota), **Nama** (nama anggota), **Alamat** (alamat rumah anggota), dan **Telp** (nomor telpon anggota). Adapun *data file* buku memiliki *fields* : **Kd_Buku** (kode buku), **Jd_Buku** (judul buku), **Nm_Peng** (nama pengarang), **Jns_Buku** (jenis buku), dan **Jml_Buku** (jumlah buku yang tersedia).

Mana isi *field* yang akan berubah ketika ada anggota meminjam buku ?, apakah nama anggota akan berubah ketika meminjam buku ?, tentu tidak. Apakah judul buku akan berubah ketika dipinjam anggota ?, tentu juga tidak. Ya, yang berubah adalah jumlah buku yang tersedia (**Jml_Buku**). Jika ada anggota yang meminjam buku, maka jumlah buku yang tersedia di perpustakaan itu akan berkurang, dan sebaliknya, jika ada anggota yang mengembalikan buku pinjamannya, maka jumlah buku yang tersedia akan bertambah. Dengan demikian, *data file* **Buku** adalah *master file* yang bersifat *dynamic*, dan *data file* **Anggota** adalah *master file* yang bersifat *reference*.

Mengapa disebut *reference*?, *referensi* buat siapa? Jawaban atas pertanyaan-pertanyaan seperti ini akan lebih jelas jika kita sudah membahas *database* di bab tersendiri.

Mana yang lebih sering diubah datanya, *master file* atau *transaction file* ? Jika kita kembali ke sekolah, di sana ada transaksi "**Murid** mengikuti **mata pelajaran**," atau "**Murid** diajar **Guru**," dan sebagainya. Pertanyaan di atas bisa diperjelas dengan pertanyaan "mana yang lebih sering, **Murid** diajar **Guru** atau **Pegawai** menambah data **Murid** (karena ada murid baru) ?," tentu, *transaction file* akan lebih sering diubah datanya ketimbang *master file*.

Contoh sebuah *data file* :



- NIM, NAMA, ALAMAT, dan TELP adalah nama-nama atribut (*field name*)
- 10199123, dan seterusnya ke bawah adalah nilai data (*data value*) dari atribut NPM
- Hesti Firtriani, Reni, Mardiana, Lidya Pratiwi, dan seterusnya ke bawah adalah nilai data (*data value*) dari atribut NAMA
- dan seterusnya.

Gambar 1. Contoh sebuah *data file*

7. *Database* atau Basis Data

Database yang dibicarakan di sub-bab ini adalah *database* dalam pengertian ukuran dari data (yang dimulai dari ukuran data terkecil di komputer, yaitu *bit*). Namun demikian, beberapa teori awal tetap diberikan di sini sebagai pendukung pengetahuan untuk pembahasan *database* di bab tersendiri.

Kumpulan dari *data file* di dalam suatu *enterprise* disebut dengan *database*. Jadi, jika dimisalkan sekolah itu adalah suatu *enterprise* (lingkungan), maka *database* **Sekolah** adalah kumpulan *data file* dari : data **Murid**, data **Guru**, data **Karyawan**, data **Nilai** murid, data **Pembayaran** uang sekolah, **Gaji** guru, dan sebagainya. Jadi, kesimpulannya adalah segala sesuatu catatan (*data file*) yang diperlukan dari suatu lingkungan, dibuat dan disatukan di dalam satu tempat (penyimpanan data eksternal), disebut dengan *database*. Pengertian ini akan lebih dikembangkan di pembahasan mengenai *database* di bab khusus tentang *database*.

8. *Data Bank* atau Bank Data

Jika di setiap sekolah telah memiliki *database*, maka Departemen yang mengurusi bidang pendidikan (sekarang Departemen Pendidikan Nasional/ Depdiknas) dapat mengumpulkan seluruh *database* tersebut dan disatukan di sana. Kumpulan dari *database* tersebut, selanjutnya disebut dengan bank data.

Tentu saja, untuk penyatuan tersebut diperlukan syarat-syarat yang tidak mudah, misalkan, penyediaan kapasitas memori eksternal yang sangat besar, penyeragaman seluruh tipe data dari seluruh sekolah yang ada (salah satu misal penyeragaman data adalah, semua sekolah harus menuliskan nama siswanya maksimal 25 karakter, tidak ada yang boleh lebih).

Contoh perusahaan yang menggunakan bank data saat ini adalah bank pengelola kartu kredit yang berlaku di berbagai negara di seluruh dunia, atau perusahaan jasa pengiriman barang yang memiliki cabang di banyak kota di seluruh dunia.

Soal-soal dan Tugas

1. Mengapa hampir di seluruh perusahaan menengah dan besar menggunakan komputer sebagai alat bantu dalam proses administrasinya ?
2. Buatlah 10 contoh data dan 10 contoh informasi dan berikan alasannya.
3. Mengapa data di komputer harus diberikan ukuran ?
4. Jika Anda berada di sebuah Departemen di dalam pemerintahan, tentukan beberapa *data file*, *database*, dan bank data yang harus ada.
5. Tentukan *master file* dan *transaction file* yang ada di sebuah *supermarket* ?
6. Catatlah beberapa *extension file name* yang berhubungan dengan *database*.
7. Catatlah beberapa kondisi di mana digunakan istilah *bit* atau *byte*.

Diambil dari buku ”Konsep Sistem Informasi: dari Bit sampai ke Database”
Bambang Wahyudi
Andi Offset
2007

STRUKTUR DATA

Jadi, struktur data adalah susunan data yang disusun sedemikian rupa agar:

1. Penyimpanan datanya menggunakan memori seefisien mungkin;
2. Data dapat diraih (*retrieve*) kembali dengan tepat.

Nanti, di materi struktur data, akan ada penggambaran bagaimana memori diibaratkan sebagai tumpukan (*stack*), antrean (*queue*), struktur kait (*linked list*), struktur pohon (*tree*), dan jejaring (*graph*). Pengibaratan/ perumpamaan tersebut dilakukan untuk memper-mudah penggambaran bagaimana proses dilakukan terhadap data yang akan dimasukkan atau dikeluarkan dari memori komputer.

Proses yang dilakukan harus memiliki ketentuan baku (pasti dan konsisten), sehingga dapat memiliki algoritma (aturan-aturan logis) yang diberlakukan terhadap prosesnya. Karenanya, materi struktur data sangat erat kaitannya dengan materi algoritma pemrograman.

MATERI-MATERI TERKAIT

Selain materi algoritma pemrograman, materi-materi lain yang sangat erat kaitannya dengan materi struktur data adalah : database, teknik kompilasi, berkas dan akses, dan lain sebagainya.