RINGKASAN BUKU TEKS

SAINS KOMPUTER

TINGKATAN 4

MOHD RAMADAN BIN SAIDIN

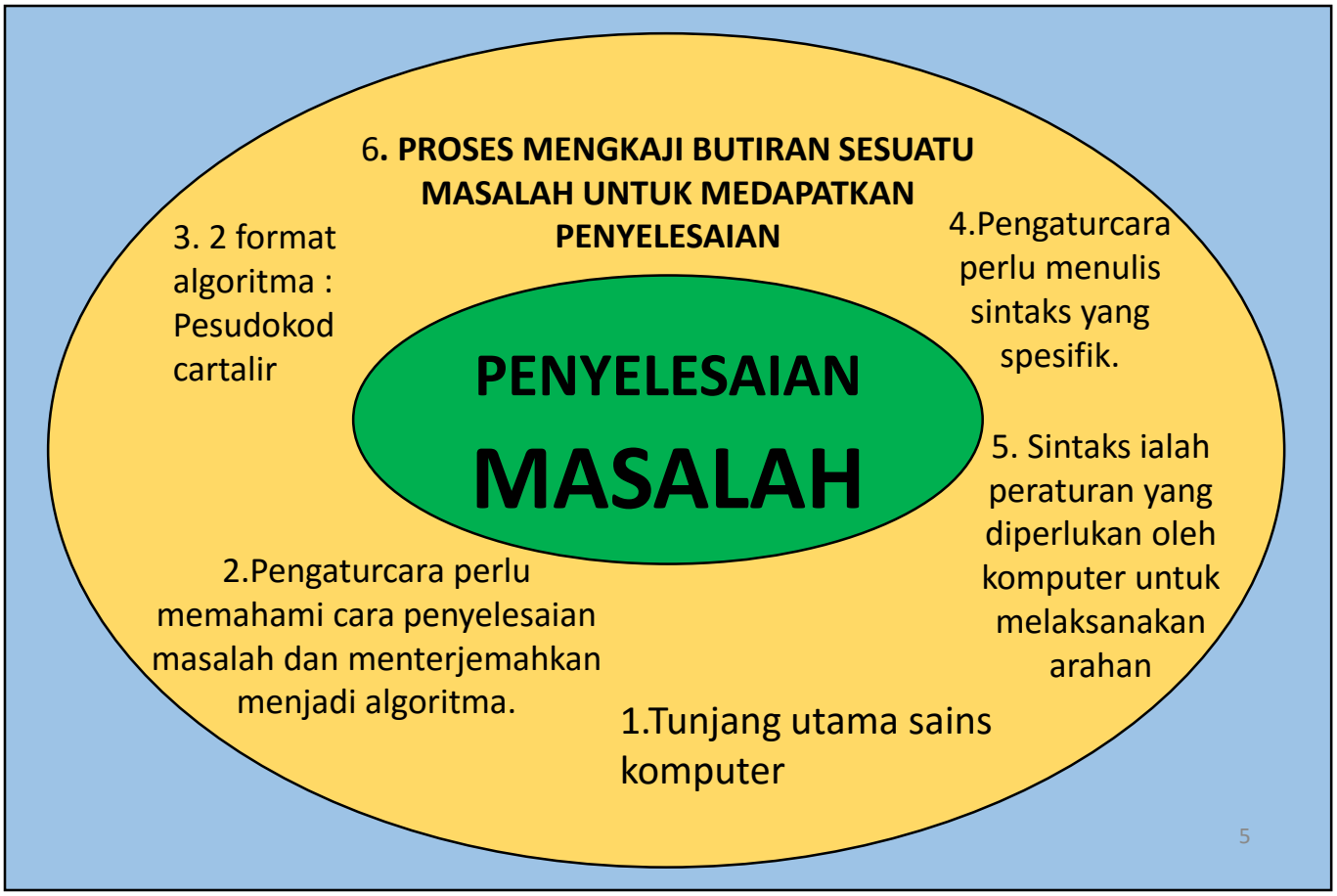
1.0

PENGATURCARAAN

1.0 PENGATURCARAAN

- 1.1 Strategi Penyelesaian Masalah
- 1.2 Algoritma
- 1.3 Pemboleh Ubah, Pemalar dan Jenis Data
- 1.4 Struktur Kawalan
- 1.5 Amalan Terbaik Pengaturcaraan
- 1.6 Struktur Data dan Modular
- 1.7 Pembangunan Aplikasi

1.1 STRATEGI PENYELESAIAN MASALAH



1.1

1.1.1 MENERANGKAN KEPERLUAN PENYELESAIAN MASALAH BERSTRATEGI

1.1

1.1.2 MENJELASKAN CIRI-CIRI PENYELESAIAN MASALAH BERKESAN

CIRI PENYELESAIAN MASALAH BERKESAN

- KOS
- MASA
- SUMBER

KOS

- Harga yang perlu dibayar untuk memperoleh,mengeluarkan dan menyelenggara.
- Biasanya wang,masa,tenaga dan perbelanjaan.
- PROJEK NORMAL – Projek yang disiapkan mengikut masa dan kos yang diperuntukkan.
- CRASHING COST – Usaha yang maksimum untuk menyelesaikan projek dalam masa yang terpendek.
- Kos meningkat apabila tempoh masa menurun.

MASA

- Merujuk kepada projek disiapkan mengikut tempoh masa yang ditetapkan.
- Aktiviti perlaksanaan yang tertunda/lambat akan meningkatkan kos.
- Keperluan menyiapkan projek dalam masa tercepat/terhad juga meningkatkan kos.

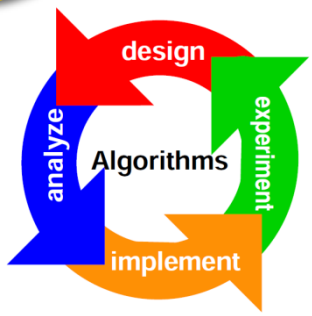
SUMBER

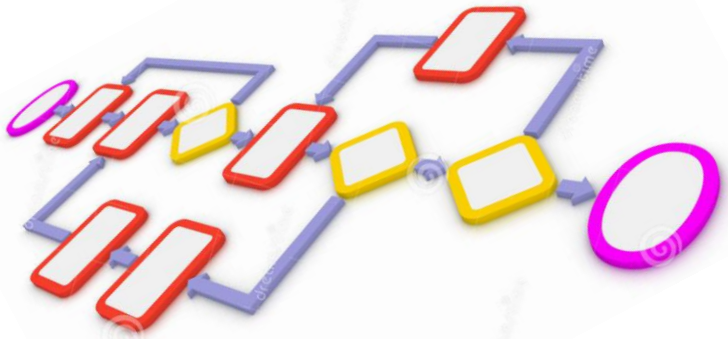
- Stok/wang, bahan-bahan mentah, staf dan asset lain yang boleh digunakan oleh organisasi supaya dapat berfungsi dengan efektif.
- Diperlukan untuk menjana hasil atau perkhidmatan.
- CONTOH : Sumber manusia,sumber kewangan.
- Perancangan Sumber – Tanggungjawab pihak pengurusan untuk mendapatkan keputusan yang optimum.
- Perancangan rapi dapat elak pembaziran sumber.
- Kekurangan sumber akan melambatkan masa menyiapkan projek dan meningkatkan kos.
- Penjadualan sumber – elak kelewatan projek.

1.1 1.1.3 MENGGUNAKAN PROSES PENYELESAIAN MASALAH

PROSES	AKTIVITI
MENGUMPUL DAN MENGANALISIS DATA	• Mengumpul data tentang punca dan skop masalah. • Mengenalpasti hal yang berkaitan dengan situasi.
MENENTUKAN MASALAH	• Mengenalpasti masalah utama yang perlu diselesaikan. • Mengenalpasti masalah seterusnya.
MENJANA IDEA	• Menyenaraikan beberapa idea yang dapat digunakan untuk menyelesaikan masalah.
MENJANA PENYELESAIAN	• Menyenaraikan idea atau langkah semasa merancang penyelesaian.
MENJANA TINDAKAN	• Membuat pilihan terbaik daripada senarai idea yang dibuat.
MELAKSANAKAN PENYELESAIAN	• Menggunakan pelbagai alat dan teknik yangtelah dipilih untuk melaksanakan penyelesaian.
MEMBUAT PENILAIAN	• Penilaian dilaksanakan terhadap langkah penyelesaian.
MEMBUAT PENAMBAHBAIKAN	• Setiap penyelesaian perlu ditambah baik jika ada kekurangan dan mengikut keperluan.

1.2 ALGORITMA





ALGORITMA

1. Satu set arahan untuk menyelesaikan masalah.
2. Arahan-arahan terperinci yang dapat diikuti oleh pembaca
3. Algoritma dapat dihalusi dengan menambahkan butiran.

CIRI-CIRI ALGORITMA

- Butiran Jelas
- Boleh dilaksanakan
- Mempunyai batasan.

1.2 1.2.1 MENGGUNAKAN ALGORITMA UNTUK MENYATAKAN PENYELESAIAN KEPADA MASALAH

KONSEP INPUT-PROSES-OUTPUT (IPO) UNTUK PERISIAN KOMPUTER

ANALISIS IPO

- Mengenalpasti data input, proses, untuk mengubah nilai data kepada maklumat dan paparan output maklumat seterlah proses.
- Carta Input-Proses-Output(IPO) boleh digunakan untuk menganalisis masalah.

CARTA IPO

INPUT	* Harus mengenalpasti data yang perlu dibaca daripada pengguna atau persekitaran.
PROSES	* Langkah-langkah ataupun rumusan untuk memproses data input kepada output.
OUTPUT	*Harus mengenalpasti output yang dikehendaki, yakni apa yang perlu dipaparkan pada skrin diakhir aturcara.

Contoh CARTA IPO

INPUT	Tahun_kelahiran
PROSES	1. Baca Input Tahun_kelahiran. 2. Dapatkan tahun semasa daripada sistem komputer, tahun_semasa. 3. Umur = Tahun_semasa – Tahun_kelahiran
OUTPUT	Umur

1.2

1.2.1 MENGGUNAKAN ALGORITMA UNTUK MENYATAKAN PENYELESAIAN KEPADA MASALAH

PERWAKILAN ALGORITMA

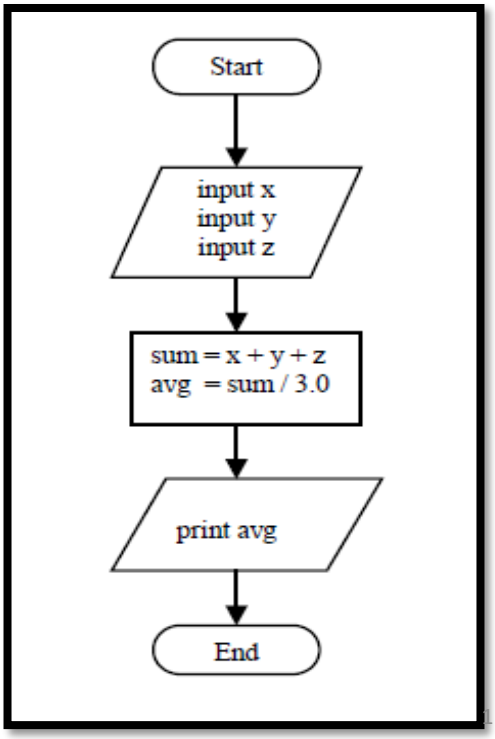
Pseudokod

Senarai struktur kawalan komputer yang ditulis dalam bahasa pertuturan manusia dan mempunyai nombor turutan.

```
Begin
    input x
    input y
    input z
    sum = x + y + z
    avg = sum / 3.0
    print avg
End
```

Carta Alir

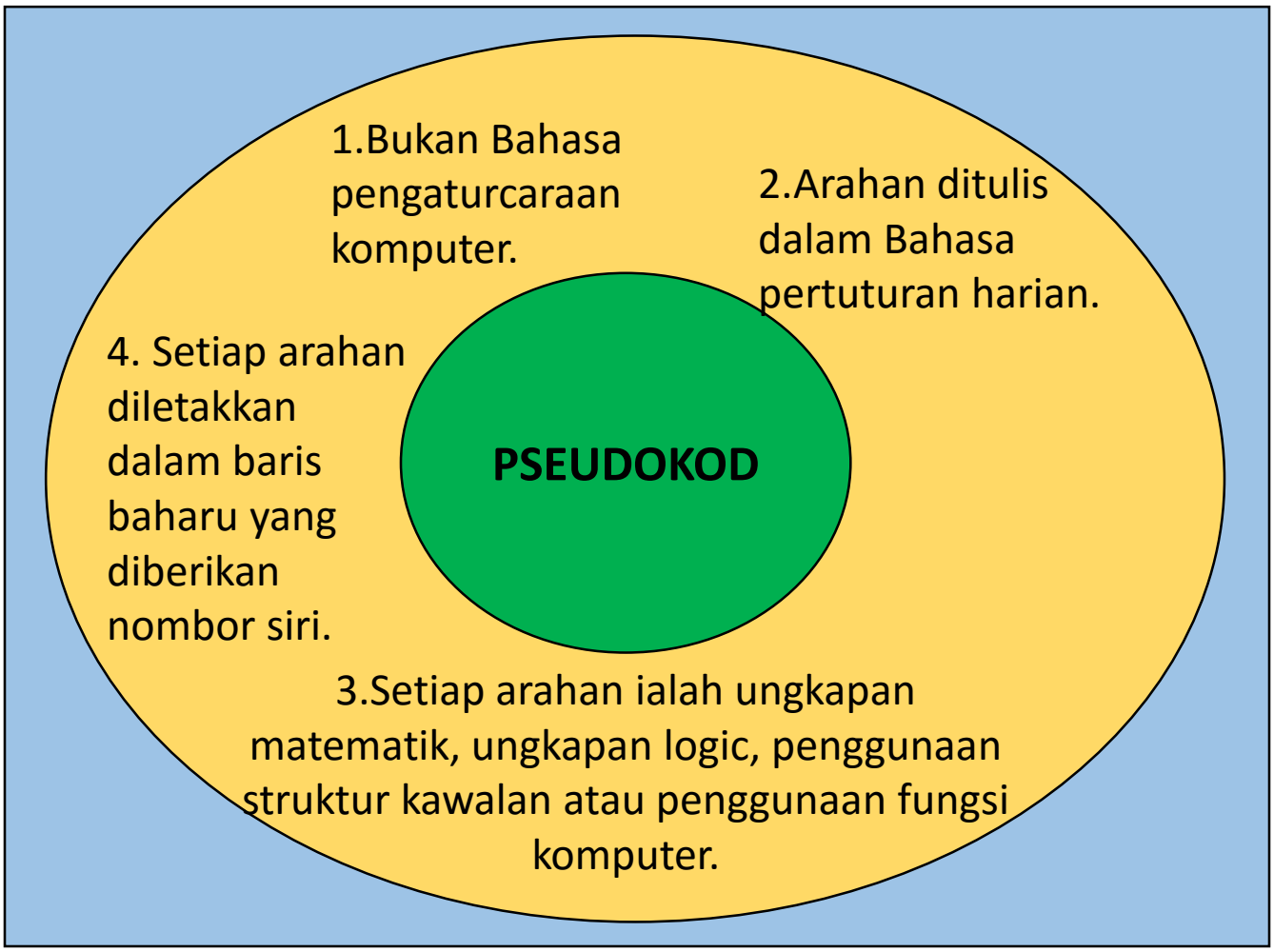
Alternatif kepada pseudokod, menggunakan simbol grafik untuk mewakili arahan-arahan penyelesaian.



1.2

1.2.1 MENGGUNAKAN ALGORITMA UNTUK MENYATAKAN PENYELESAIAN KEPADA MASALAH

PSEUDOKOD



LANGKAH-LANGKAH MENULIS PSEUDOKOD

- Tulis kenyataan **MULA.**
- Baca **INPUT.**
- Proses data menggunakan ungkapan logik atau matematik.
- Papar **OUTPUT.**
- Tulis kenyataan **TAMAT.**

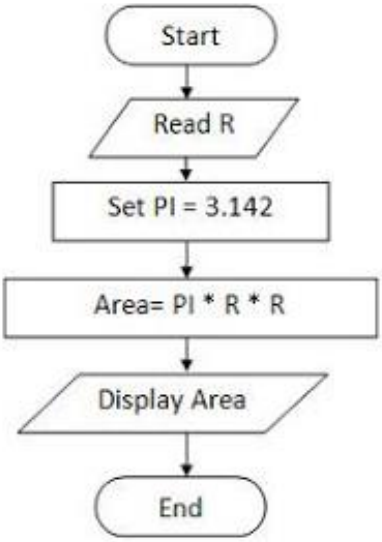
Pseudocode

```
BEGIN
   input hours
   input rate
   pay = hours * rate
   print pay
END
```

1.2

1.2.1 MENGGUNAKAN ALGORITMA UNTUK MENYATAKAN PENYELESAIAN KEPADA MASALAH

CARTA ALIR



SIMBOL	NAMA NOD	FUNGSI
MULA	Terminal Mula	Permulaan algoritma dalam carta alir.
TAMAT	Terminal Tamat	Penamat algoritma dalam carta alir.
	Input/Output	Membaca input atau memaparkan output ke skrin
	Proses	Arahan untuk memproses input dalam bentuk ungkapan, memproses fail dan sebagainya.
	Penghubung	Titik sambungan untuk menyambungkan carta alir yang terpisah.
	Syarat	• Menguji syarat yang terkandung dalam syarat. • Terdapat satu anak panah masuk dan 2 anak panah keluar.
	Aliran aktiviti	Menghubungkan nod-nod untuk menunjukkan aliran proses.

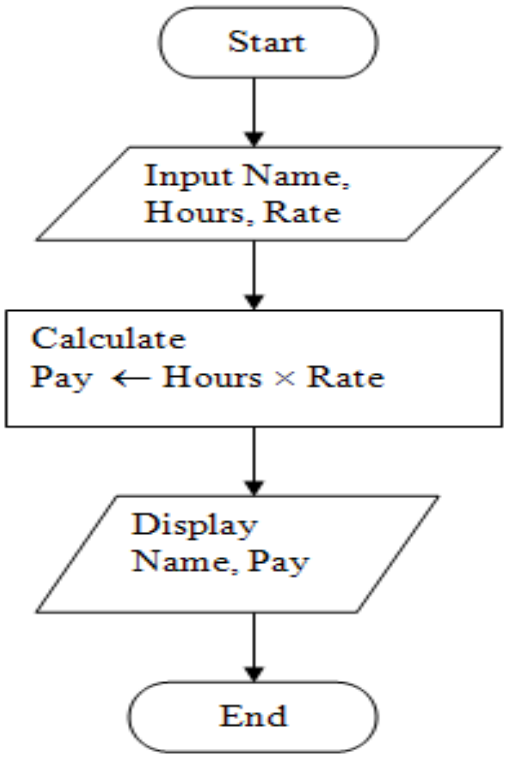
1.2

1.2.1 MENGGUNAKAN ALGORITMA UNTUK MENYATAKAN PENYELESAIAN KEPADA MASALAH

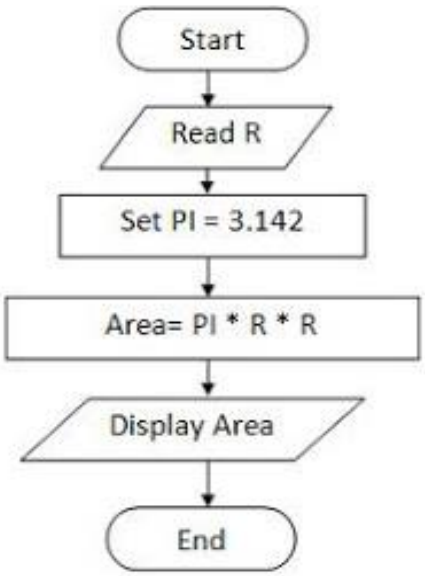
CARTA ALIR

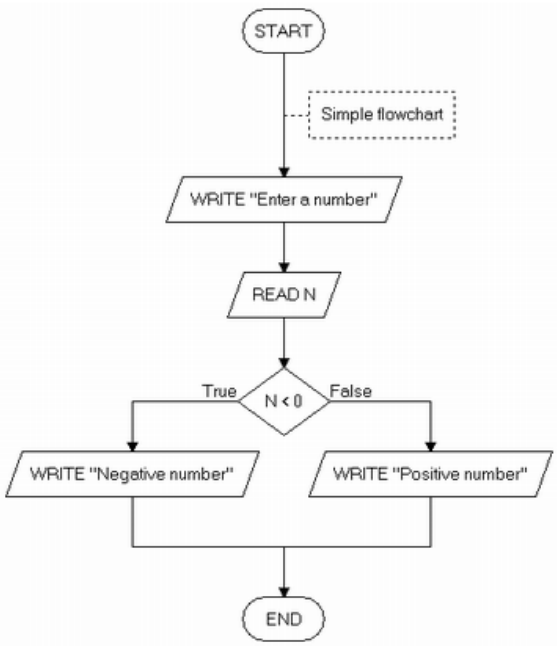
LANGKAH-LANGKAH MEMBINA CARTA ALIR

- Lukis nod terminal MULA.
- Lukis garis penghubung.
- Lukis nod input, masukkan butiran seperti umpukan data.
- Lukis garis penghubung.
- Lukis nod proses, Masukkan butiran seperti ungkapan matematik.
- Lukis garis penghubung.
- Sekiranya perlu, lukis nod proses atau nod input lain-lain yang diperlukan,
- Sekiranya tiada, lukis nod terminal tamat.



CONTOH CARTA ALIR





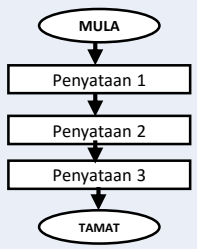
1.2

1.2.2 MEMBINA ALGORITMA PENYELESAIAN MASALAH DENGAN MENGGUNAKAN STRUKTUR KAWALAN DALAM PENGATURCARAAN

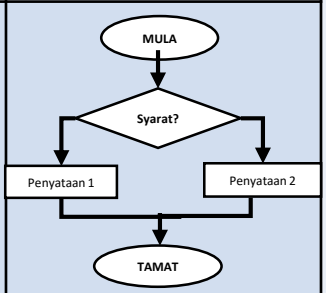
STRUKTUR KAWALAN

- Struktur Kawalan Urutan
- Struktur Kawalan Pilihan
- Struktur Kawalan Pengulangan

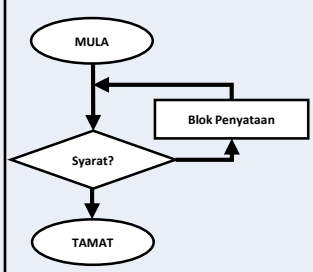
STRUKTUR KAWALAN URUTAN

PSEUDOKOD	CARTA ALIR	
Mula Penyataan 1 Penyataan 2 Penyataan 3 Tamat		• Melaksanakan arahan-arahan Komputer satu per satu. • Urutan arahan yang betul penting kerana urutan yang berlainan boleh memberikan output yang berlainan. • Setiap arahan adalah satu pernyataan algoritma. • Urutan algoritma disusun secara linear.

STRUKTUR KAWALAN PILIHAN

PSEUDOKOD	CARTA ALIR	
JIKA syarat benar MULA_JIKA BLOK Penyataan 1 TAMAT_JIKA JIKA_TIDAK MULA_JIKA TIDAK BLOK Penyataan 2 TAMAT_JIKA_TIDAK		• Memberikan perisian komputer keupayaan untuk membuat keputusan berasaskan syarat yang telah ditentukan oleh pengatur cara. • Struktur ini membolehkan arahan-arahan lain komputer dilaksanakan dalam situasi masalah yang berbeza. • Ungkapan logik digunakan dalam syarat. • Operator Aritmetik digunakan dalam ungkapan logik.

STRUKTUR KAWALAN PENGULANGAN

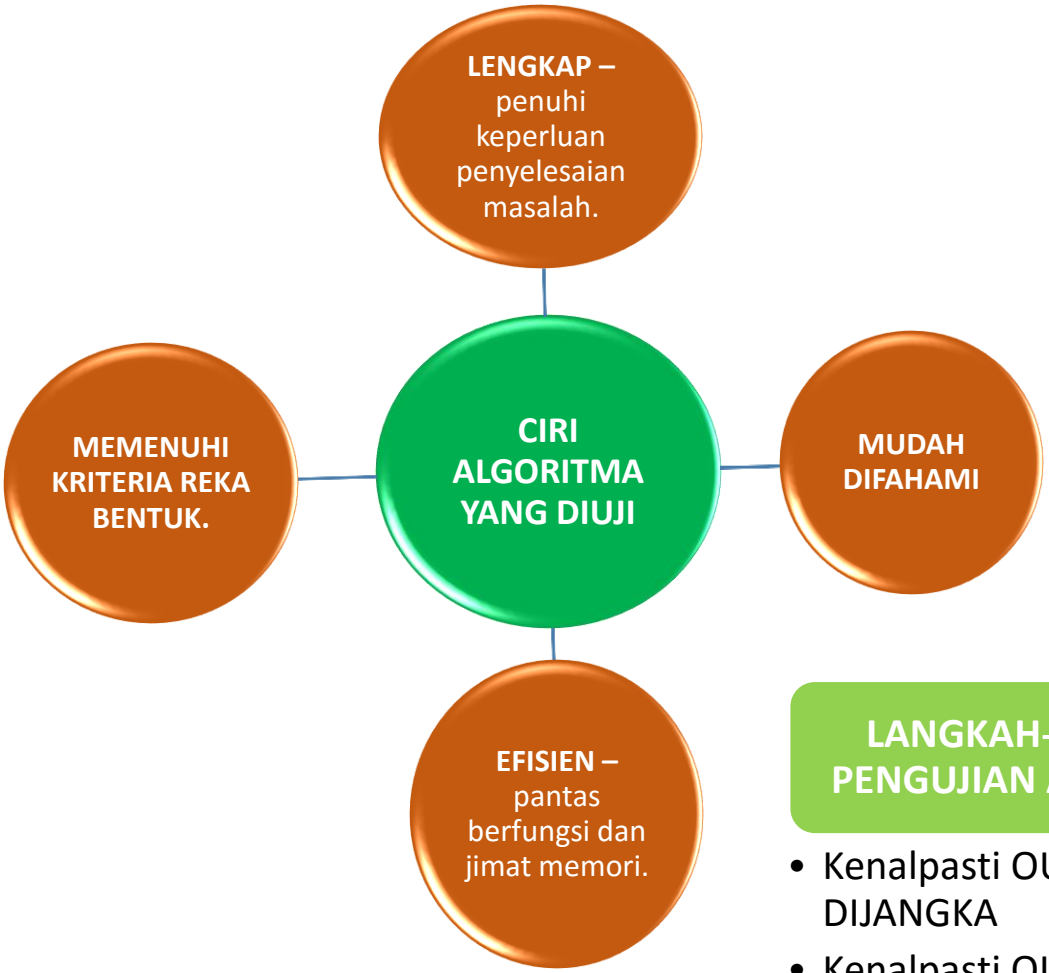
PSEUDOKOD	CARTA ALIR	
SELAGI Syarat MULA_SELAGI BLOK Penyataan TAMAT_SELAGI		• Mengulang arahan-arahan komputer. • Ulangan boleh berlangsung sehingga menerima syarat berhenti atau mencapai bilangan yang ditetapkan.

1.2 1.2.3 MENGUJI DAN MEBAIKI RALAT DALAM ALGORITMA

PENGUJIAN ALGORITMA

- Algoritma diuji untuk tujuan pembaikan.
- Dibuat sebelum algoritma ditulis sebagai kod komputer.
- MATLAMAT PENGUJIAN - untuk memastikan logik algoritma adalah betul dan memikirkan pembaikan algoritma supaya lebih efisien.

TULIS ALGORITMA → UJI ALGORITMA → PEMBETULAN → PENGATURCARAAN



LANGKAH-LANGKAH PENGUJIAN ALGORITMA

- Kenalpasti OUTPUT DIJANGKA
- Kenalpasti OUTPUT DIPEROLEH
- Bandingkan OUTPUT DIPEROLEH dengan OUTPUT DIJANGKA.
- Analisis dan baiki algoritma

1.2 1.2.3 MENGUJI DAN MEBAIKI RALAT DALAM ALGORITMA

JENIS RALAT ALGORITMA

- RALAT SINTAKS
- RALAT LOGIK
- RALAT MASA LARIAN

RALAT SINTAKS

- Tidak wujud dalam algoritma.
- Berlaku kerana **cuai semasa menggunakan Bahasa pengaturcaraan.**
- Biasanya ditemui secara automatik oleh perisian compiler Bahasa pengaturcaraan.
- Ralat algoritma tidak menyebabkan ralat sintaks.

RALAT LOGIK

- Berlaku kerana **perisian** yang dihasilkan **tidak menjalankan fungsi yang sepatutnya, tidak lengkap atau menghasilkan output yang tidak tepat.**
- PUNCA - Ungkapan/formula yang salah, kecuaian, jenis data tidak sesuai, umpukan tidak betul.

RALAT MASA LARIAN

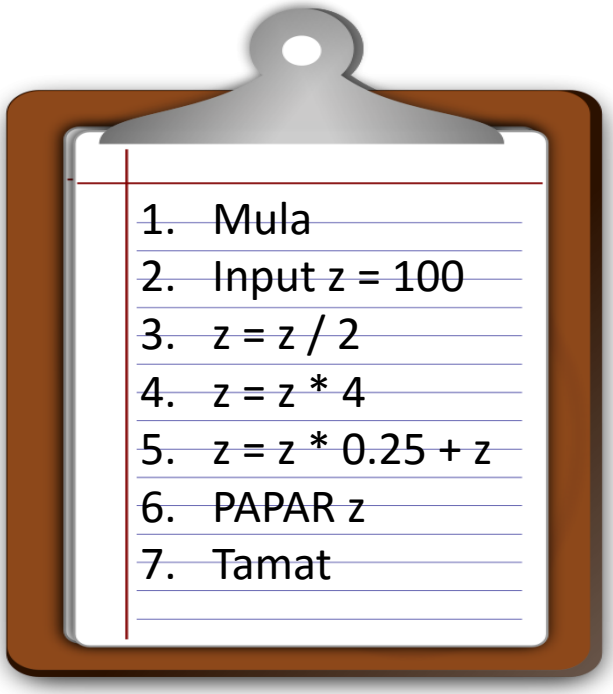
- Ralat yang **timbul apabila aturcara dijalankan**.
- Contoh – Aturcara tidak dapat dimulakan, sangat perlahan atau tidak responsive.
- Boleh dikenalpasti daripada kegagalan output dan paparan amaran dalam aturcara.
- Boleh dikesan melalui reka bentuk algoritma yang tidak efisien atau salah.
- CONTOH – Struktur kawalan tidak betul, pembolehubah tiada nilai,pembahagian dengan sifar, logik syarat salah dalam pengulangan.

1.2 1.2.4 MENGESAN NILAI PEMBOLEH UBAH PADA SETIAP TAHAP DALAM ALGORITMA

PEMBOLEH UBAH

- Algoritma mengumpuk dan mengubah nilai sesuatu pembolehubah.
- Nilai pembolehubah adalah tidak tetap.
- Setiap baris algoritma mungkin membuat perubahan pada pemboleh ubah tertentu.
- Jadual pemboleh ubah digunakan untuk mengesan nilai pemboleh ubah pada setiap tahap algoritma.

CONTOH



JADUAL BERNOMBOR

NO	Z	I /O
1	-	-
2	100	100
3	50	-
4	200	-
5	250	-
6	250	250
7	-	-

1.2

1.2.5 MENGENAL PASTI OUTPUT YANG BETUL DARI ALGORITMA BERDASARKAN INPUT YANG DIBERI

OUTPUT

- Output yang betul bergantung kepada pemboleh ubah sewaktu algoritma papar dipanggil.
- Membandingkan output dijangka merupakan satu-satunya cara menentukan kesahihan output algoritma.
- Output dijangka ditentukan secara hitungan manual.
- Jika output algoritma adalah betul, output algoritma sepatutnya bersamaan dengan outpun dijangka.

CONTOH

JADUAL BERNOMBOR

NO	x	y	Jumlah	I /O
1	-	-	-	-
2	12	-	-	-
3	12	88	-	-
4	12	88	12 + 88 = 100	
5	12	88	100	Jumlah = 100
6	-	-	-	-

1.2

1.2.6 TERJEMAHAN ALGORITMA KE DALAM BAHASA PENGATURCARAAN

TERJEMAHAN ALGORITMA

- Setiap baris algoritma yang direka bentuk dapat ditukarkan kepada kod computer.
- Algoritma berbentuk universal.
- Oleh itu, simbol dan perkataan yang digunakan tidak perlu bersandarkan kepada mana-mana Bahasa pengaturcaraan.
- Contoh Bahasa pengaturcaraan ialah Visual Basic (VB), Java,C# dan lain-lain.

CONTOH

ALGORITMA	KOD KOMPUTER (JAVA)
Mula	Public static void main (String[] args) {
PAPAR " Nama Pengguna"	System.out.print (" Masukkan nama pengguna : ");
INPUT Nama	String nama new java.util. Scanner (System.in) nextLine();
PAPAR " Apa Khabar". Nama , "?"	System.out.println (" Hello " + nama);
Tamat	;

1.3 PEMBOLEH UBAH, PEMALAR DAN JENIS DATA

- **Pemboleh Ubah** atau **pemala**r dalam *Java* perlu diisytiharkan sebelum digunakan.
- Dalam proses pengaturcaraan, seorang pengatur cara perlu mengisytiharkan **jenis data** yang diperlukan dalam sesuatu program yang hendak dilaksanankan.

```
public class Class{
    public static void method() {
        ArrayList list = new ArrayList();
        list.add("string");
        anotherMethod("string");
    }
    public static Strin                    String s) {
        return s;
    }
```

Expressions

"string"

anotherMethod(...)

```
public class Demo {
    public static void main(String[] args) throws IOException {
        //declare new File and Scanner objects
        File file = new File("input.txt");
        Scanner inputFile = new Scanner(file);
        //loop through txt file
        while(inputFile.hasNext()){
            //read next line
            String line = inputFile.nextLine();
            System.out.print(line);
            //call check method to determine balance
            if(check(line))
                System.out.print("\t--> correct\n");
            else
                System.out.print("\t--> incorrect\n");
        }
        inputFile.close();
    }
```

1.3

PEMBOLEH UBAH, PEMALAR DAN JENIS DATA

PEMBOLEH UBAH

PEMBOLEH UBAH

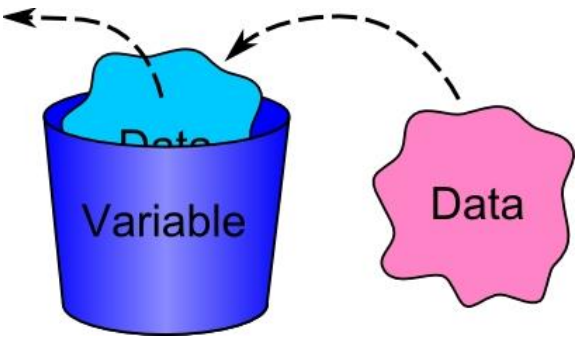
- Ruang simpanan sementara untuk nombor , teks dan objek.
- Nilai pemboleh ubah sentiasa berubah semasa berlakunya pemprosesan data dan tidak akan memegang sebarang nilai selepas program tamat.
- Pengatur cara perlu memberikan nama kepada setiap pemboleh ubah yang digunakan untuk menyelesaikan sesuatu masalah dalam program yang dibangunkan.
- Pengatur cara menggunakan nama pemboleh ubah sebagai nama rujukan untuk nilai spesifik pemboleh ubah tersebut.
- Komputer menggunakan nama pemboleh ubah sebagai rujukan untuk mencari nilai pemboleh ubah itu dalam memorinya

When you declare a variable the computer:

- **Allocates some memory** large enough to hold the data
- Assigns that memory block the **identifier** you entered

```
int num1;
int num2;
int result;
```

MEMORY	
num1	
result	
num2	



1.3

PEMBOLEH UBAH, PEMALAR DAN JENIS DATA

PEMALAR

PEMALAR

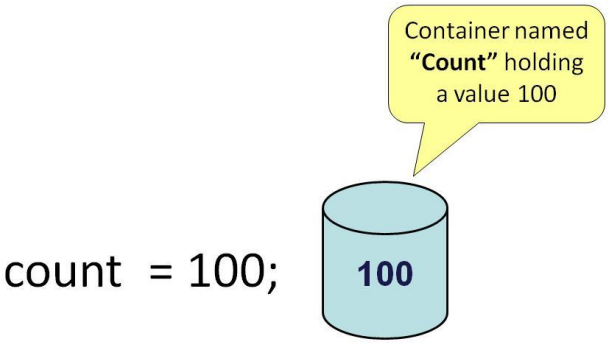
- Nilai pemalar adalah tetap dan tidak akan berubah sewaktu proses pengaturcaraan dilaksanakan.
- Digunakan semasa pengatur cara ingin mengisytiharkan nilai yang tidak akan berubah.
- Jenis data yang diisytiharkan mestilah sepadan dengan nilai.

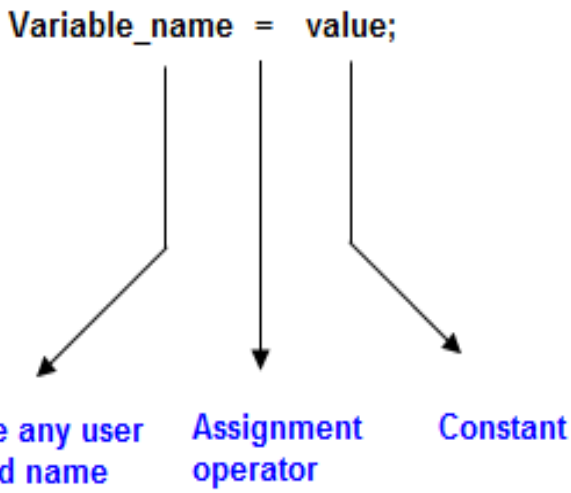
When you assign a value with the = sign

- The value is stored in the memory location for that variable

```
num1 = 10; // assigning
num2 = 7;
result = num1 + num2;
```

MEMORY	
num1	10
result	17
num2	7





1.3

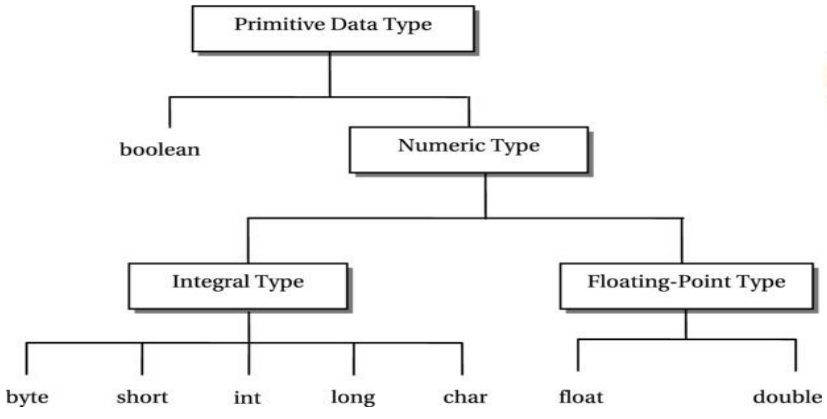
PEMBOLEH UBAH, PEMALAR DAN JENIS DATA

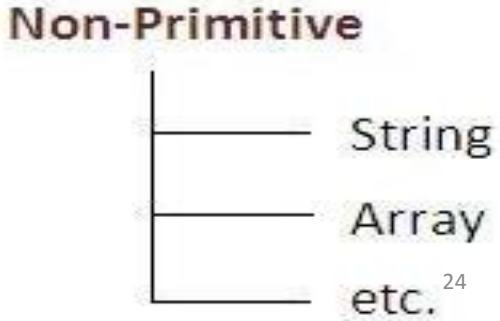
JENIS DATA

JENIS DATA

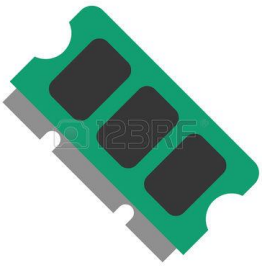
- Satu set data yang mempunyai nilai dan ciri-ciri yang telah ditetapkan.
- Data akan diproses menghasilkan output.
- Jenis data boleh dikategorikan kepada 2 kelas iaitu data primitive dan data bukan primitive.

JENIS DATA	CONTOH NILAI		KAPASITI INGATAN KOMPUTER
Integer	Minimum	-2147483648	4 bait
	Maksimum	2147483647	
float	Minimum	-3.4e38	4 bait
	Maksimum	3.4e38	
double	Minimum	-1.7e308	8 bait
	Maksimum	3.4e38	
char	Satu karakter sahaja		2 bait
String	Bermula dari 0 hingga tiada had.		> 10 bait
Boolean	Benar (true)		1 bait
	Palsu (false)		





1.3 1.3.1 MENJELASKAN HUBUNGAN ANTARA JENIS DATA, SAIZ DATA DAN KAPASITI INGATAN KOMPUTER



- Setiap jenis data akan disimpan dalam ingatan komputer.
- Nama pemboleh ubah memainkan peranan yang penting dalam menentukan saiz data dalam ingatan.
- Sistem operasi akan menentukan apa-apa yang boleh disimpan dalam ingatan computer berdasarkan jenis data yang digunakan oleh pemboleh ubah.
- Kepelbagaian penggunaan jenis data pada pemboleh ubah dapat menjimatkan ruang pada ingatan komputer.

1.3 1.3.2 MEMILIH DAN MENGGUNAKAN JENIS DATA YANG BERSESUAIAN

- Pemilihan dan penggunaan data yang sesuai amat penting supaya aturcara dapat dibangunkan tanpa ralat sintaks.
- Jenis data bagi pemboleh ubah dan pemalar yang menentukan jenis maklumat akan disimpan dalam ruang ingatan yang diperuntukkan.
- Penggunaan jenis data yang sesuai berfungsi sebagai penanda aras kepada sesuatu pemboleh ubah itu sama ada pemboleh ubah tersebut menyimpan data yang tetap, data yang boleh dikira, huruf, nombor perpuluhan atau data yang mempunyai nilai benar atau palsu.

JENIS DATA

INTEGER	NOMBOR NYATA	AKSARA / RENTETAN	BOOLEAN
Int Merangkumi semua nombor yang tidak mempunyai nilai pecahan atau perpuluhan.	***float,double*** Terdiri daripada semua nombor yang mempunyai titik perpuluhan atau bahagian pecahan.	***char,string*** Data dalam bentuk ruang kosong (space), teks, perkataan atau nilai yang mengandungi susunan aksara atau simbol.	Data dalam bentuk pilihan iaitu memilih salah satu daripada sesuatu yang *benar* atau *palsu.*

1.3 1.3.3 MEMBEZAKAN PEMBOLEH UBAH SEJAGAT (GLOBAL) DAN SETEMPAT (LOCAL)

- Setiap pemboleh ubah dan pemalar mungkin wujud dan akan digunakan untuk keseluruhan aturcara atau hanya bagi satu fungsi.
- Kewujudan pemboleh ubah atau pemalar dikenal sebagai kawasan yang kedua-duanya boleh digunakan secara sah.

SKOP PEMBOLEH UBAH

PEMBOLEH UBAH SEJAGAT (GLOBAL)

- Hanya berfungsi dalam aturcara sahaja.
- Apabila tatacara tamat, ruang memori juga tamat.
- **PENGISYTIHARAN :** Diluar di mana-mana fungsi.
- **AKSES** : Boleh diakses dimana-mana fungsi.
- Boleh digunakan hingga ke akhir aturcara.
- Jika pemboleh ubah setempat mempunyai nama yang sama dengan pemboleh ubah sejagat , rujukan hanya dibuat terhadap pemboleh ubah terdekat (setempat).

PEMBOLEH UBAH SETEMPAT (LOCAL)

- Hanya berfungsi dalam subatur cara yang diisytiharkan.
- Digunakan dalam fungsi dimana pemboleh ubah diisytiharkan bermula dari mana pemboleh ubah diisytiharkan dan bila penamat akhir tatacara tersebut.
- **PENGISYTIHARAN :** Dalam sebuah fungsi dalam atur cara.
- **AKSES** : Tidak boleh diakses diluar fungsi itu.
- Tiada masalah jika dua fungsi menggunakan pemboleh ubah tempatan yang sama.

```
1
2                          global variable
3   var global = 10;
4
5   function fun() {
6
7       var local = 5;
8
9   }                      local variable
10
```

1.3

1.3.4 MENGISYTIHARKAN, MEMULAKAN DAN MENETAPKAN NILAI PADA PEMBOLEH UBAH DAN PEMALAR

PENGISYTIHARAN NILAI PEMBOLEH UBAH

- Pemboleh ubah dikenal sebagai tempat untuk menyimpan data.
- Setiap pemboleh ubah dalam *Java* mempunyai jenis yang tertentu yang menentukan saiz dan susun atur memori dan set operasi yang boleh digunakan.
- Semua pemboleh ubah perlu diisytiharkan sebelum boleh digunakan.
- Pengisyiharan pemboleh ubah perlu dilaksanakan dengan memberikan jenis data dan nama pemboleh ubah.

JENIS DATA	NAMA PEMBOLEH UBAH	CONTOH

INTEGER	X	public class pembolehUbah {
		public static void main(String[] args {
		int x ;
		}
		}

STRING	nama	public class pembolehUbah {
		public static void main(String[] args {
		String nama ;
		}
		}

DOUBLE	s , t, u	public class pembolehUbah {
		public static void main(String[] args {
		double s,t,u ;
		}
		}

BOOLEAN	v	public class pembolehUbah {
		public static void main(String[] args {
		boolean v ;
		}
		}

CHAR	w	public class pembolehUbah {
		public static void main(String[] args {
		char w ;
		}
		}

FLOAT	y	public class pembolehUbah {
		public static void main(String[] args {
		float y ;
		}
		}

1.3 1.3.4 MENGISYTIHARKAN, MEMULAKAN DAN MENETAPKAN NILAI PADA PEMBOLEH UBAH DAN PEMALAR

PERMULAAN DAN PENETAPAN NILAI PEMBOLEH UBAH

- Selepas pengisytiharan jenis dan nama pemboleh ubah, nilai kepada pemboleh ubah boleh ditetapkan atau diumpukkan.

JENIS DATA	NILAI	CONTOH

INTEGER	10	public class pembolehUbah {
		public static void main(String[] args {
		int x = 10 ;
		}
		}

STRING	hardeep	public class pembolehUbah {
		public static void main(String[] args {
		String nama = "hardeep" ;
		}
		}

DOUBLE	s = 0.123
	t = 1.1
	u = s + t
		public class pembolehUbah {
		public static void main(String[] args {
		double s = 0.123, ,t = 1.1 ,u = s + u ;
		}
		}

BOOLEAN	true	public class pembolehUbah {
		public static void main(String[] args {
		boolean v = true ;
		}
		}

CHAR	y	public class pembolehUbah {
		public static void main(String[] args {
		char w = 'y' ;
		}
		}

FLOAT	342.234f	public class pembolehUbah {
		public static void main(String[] args {
		float y = 342.234f ;
		}
		}

1.3

1.3.4 MENGISYTIHARKAN, MEMULAKAN DAN MENETAPKAN NILAI PADA PEMBOLEH UBAH DAN PEMALAR

PENGISYTIHARAN DAN PENETAPAN NILAI PEMALAR

- Pemalar ialah pemboleh ubah yang mempunyai nilai yang tidak berubah semasa menjalankan atur cara.

```
final int BILANGAN_HARI_DALAM_SEMINGGU = 7
```

```
final double Pi = 3.14
```

```
Static final double = 4.14
```

1.3 1.3.5 MENGGUNAKAN PERNYATAAN UMPUKAN DAN PERNYATAAN ARITMETIK

CONTOH JENIS PERNYATAAN

PERNYATAAN UMPUKAN

- Boleh terdiri daripada satu atau lebih ungkapan yang lain.
- Merujuk " sama dengan" atau simbol " = " .
- Pernyataan umpukan akan memberikan nilai kepada pembolehubah.

CONTOH

String x; x = " Hello ";
double p; p = 0.123;
float w; w = 561.123f;
int p; int q; int r; p = q =r = 255;

PERNYATAAN ARITMETIK

- Boleh terdiri daripada satu atau lebih ungkapan aritmetik.
- Merujuk kepada operasi aritmetik (+ , - , *, /).
- Unit pemprosesan utama boleh membaca operasi aritmetik dari kiri ke kanan sahaja.
- Jika pengaturcara ingin menjalankan operasi darab terlebih dahulu, penggunaan simbol kurungan () perlu diutamakan.

CONTOH

(30 – 5 + ((2*3) / 3)

CONTOH PENYATAAN ARITMETIK DALAM ATURCARA

NOTASI	CONTOH ATURCARA	CONTOH OUTPUT

OPERASI TAMBAH
+
	public class operasiTambah {
	public static void main(String[] args {
	int i = 10;
	int j = 20;
	i = i + j;
	System.out.println (“ Hasil Tambah ialah : “ + i) ;
	}
	}
		Hasil Tambah ialah : 30

OPERASI TOLAK
-
	public class operasiTolak {
	public static void main(String[] args {
	int i = 30;
	int j = 20;
	i = i - j;
	System.out.println (“ Hasil Tolak ialah : “ + i) ;
	}
	}
		Hasil Tolak ialah : 10

OPERASI DARAB
*
	public class operasiDarab {
	public static void main(String[] args {
	int i = 30;
	int j = 20;
	i = i * j;
	System.out.println (“ Hasil Darab ialah : “ + i) ;
	}
	}
		Hasil Darab ialah : 600

OPERASI BAHAGI
/
	public class operasiBahagi {
	public static void main(String[] args {
	int i = 30;
	int j = 2;
	i = i / j;
	System.out.println (“ Hasil Bahagi ialah : “ + i) ;
	}
	}
		Hasil Bahagi ialah : 15

1.3

1.3.6 MENULIS ATUR CARA UNTUK MEMASUKKAN INPUT DARI PAPAN KEKUNCI DAN MEMAPARKAN OUTPUT

ATUR CARA YANG MEMBENARKAN INPUT DIMASUKKAN MENGGUNAKAN PAPAN KEKUNCI

```
import java.util.Scanner;
Scanner input = new Scanner (System.in);
```

CONTOH #1 (Input dari pengguna)

```
import java.util.Scanner ;
public class luasSegiTiga {

        public static void main (String [] args) {
                Scanner input = new Scanner (System.in);
                System.out.println ("Sila masukkan nilai tapak : " );
                int tapak = input.nextInt();

                System.out.println ("Sila masukkan nilai tinggi  : " );
                int tinggi = input.nextInt();

                double luas = (1.0/2) * tapak * tinggi ;

                System.out.println ("Luas segi tiga ialah : " + luas );
        }
}
```

OUTPUT

```
Sila masukkan nilai tapak :
6
Sila masukkan nilai tinggi  :
4
Luas segi tiga ialah : 12.0
```

CONTOH #1 (Nilai dalam atur cara)

```
public class luasSegiTiga2 {

        public static void main (String [] args) {

                int tapak = 6;

                int tinggi = 4;

                double luas = (1.0/2) * tapak * tinggi ;

                System.out.println ("Luas segi tiga ialah : " + luas );
        }
}
```

OUTPUT

Luas segi tiga ialah : **12.0**

CONTOH #2 (Input dari pengguna)

```
import java.util.Scanner ;

        public class infoAnda {

                public static void main (String [] args) {

                        Scanner input = new Scanner (System.in);
                        System.out.println ("Siapakan nama anda ? : " );
                        String nama = input.nextInt();

                        System.out.println ("Berapakah umur anda ? : " );
                        int umur = input.nextInt();

                        System.out.println ("Apakah hobi anda ? : " );
                        String hobi = input.nextInt();

                        nama = nama;
                        umur = umur;
                        hobi = hobi;

                        System.out.println ("Nama :  " + nama );
                        System.out.println ("Umur :  " + umur );
                        System.out.println ("Hobi :  " + hobi );
        }
}
```

OUTPUT

```
Siapakan nama anda ? :
Haziq
Berapakah umur anda ? :
7
Apakah hobi anda ? :
Bermain
Nama : Haziq
Umur : 7
Hobi : Bermain
```

CONTOH #2 (Nilai dalam atur cara)

```
public class infoAnda2 {

        public static void main (String [] args) {

            String nama ;
            int umur ;
            String hobi;

            nama = “Haziq”;
            umur = 7;
            hobi = “Bermain”;

            System.out.println (“Nama :  ” + nama );
            System.out.println (“Umur :  ” + umur );
            System.out.println (“Hobi :  ” + hobi );
    }
}
```

OUTPUT

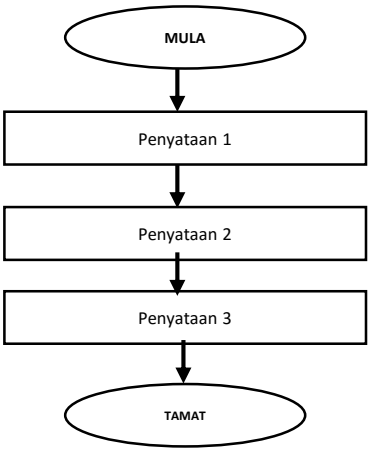
Nama : Haziq
Umur : 7
Hobi : Bermain

1.4 STRUKTUR KAWALAN

STRUKTUR KAWALAN

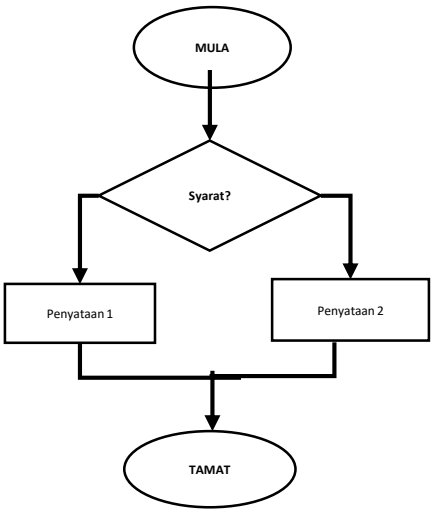
Struktur Kawalan Urutan

- Tidak bervariasi.
- Hanya mengikut urutan.



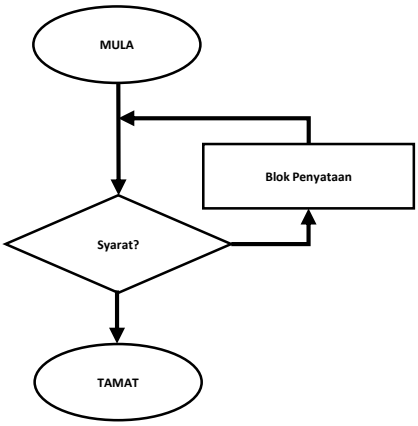
Struktur Kawalan Pilihan

- if – else - if
- Switch-case



Struktur Kawalan Pengulangan

- For
- While
- Do-while



1.4 1.4.1 STRUKTUR KAWALAN PILIHAN

KAWALAN PILIHAN

- Mekanime yang membolehkan keputusan atau pemilihan dibuat secara automatik..
- PERNYATAAN SYARAT BOOLEAN – Digunakan untuk menguji nilai input yang dimasukkan dan ini seterusnya akan menentukan set atau blok arahan yang akan dilaksanakan.
- Syarat boolean membolehkan perbandingan pemboleh ubah, sifat objek atau nilai yang dilakukan melalui operator hubungan atau operator logikal.
- Perbandingan ini memberikan keputusan dalam bentuk data jenis boolean.

BENTUK STRUKTUR KAWALAN PILIHAN

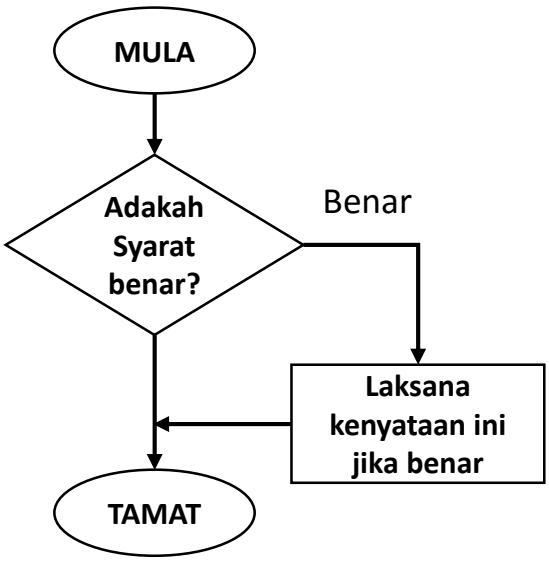
- if
- If-else
- If-else-if
- switch-case

1.4 1.4.1 MENERANGKAN ALIRAN STRUKTUR KAWALAN PILIHAN

KAWALAN PILIHAN *if*

- Hanya akan melaksanakan pernyataan-pernyataan tertentu seperti memproses data melalui ungkapan sekiranya syarat adalah benar.

CARTA ALIR	SINTAKS

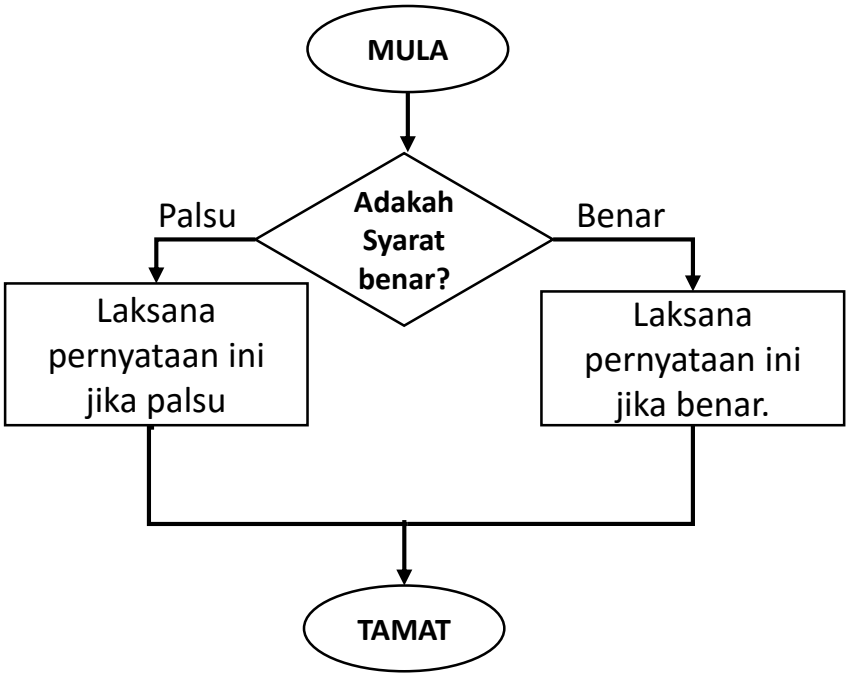
	If *(<syarat Boolean>)* {
	
	< *Arahan-arahan jika benar* >
	
	}

	CONTOH
	
	If (umur > 20) {
	
	System.out.println (“Anda layak mengundi. ”) ;
	
	}

1.4 1.4.1 MENERANGKAN ALIRAN STRUKTUR KAWALAN PILIHAN

KAWALAN PILIHAN *if-else*

- Digunakan untuk membuat keputusan dalam sesuatu atur cara.
- Menunjukkan hasil Boolean – Ya (Benar) atau Tidak (Palsu).
- Pernyataan susulan bergantung kepada hasil Boolean tersebut.

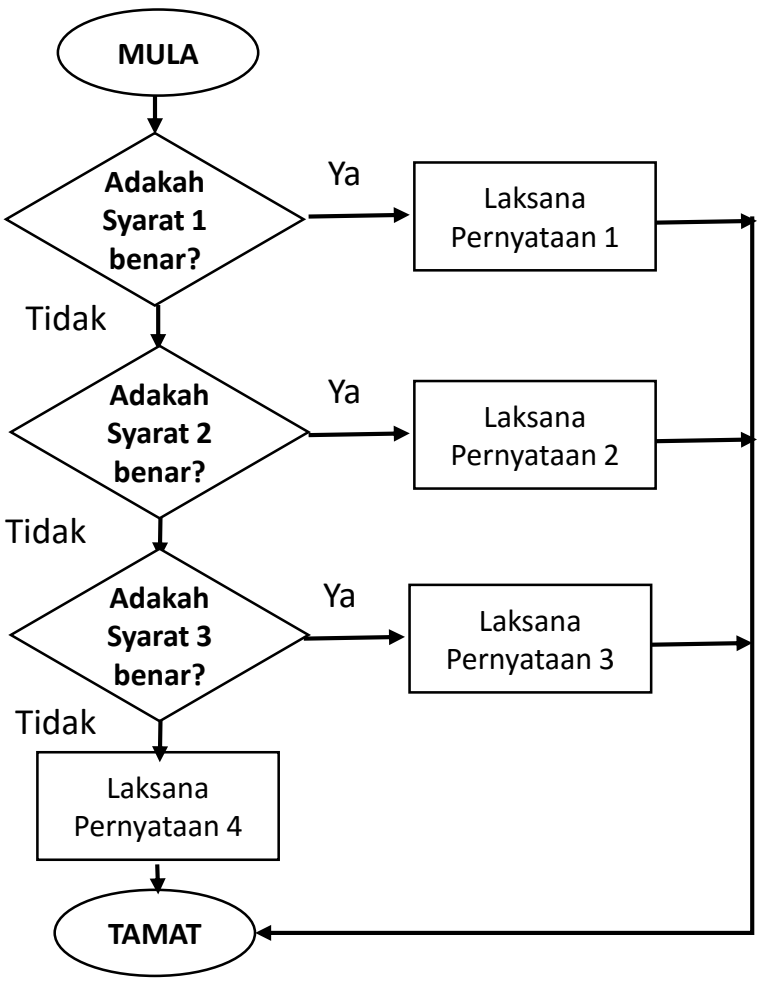
CARTA ALIR	SINTAKS
	*If (<syarat Boolean>) {* *< Arahan-arahan jika benar >* *} else* *< Arahan-arahan jika palsu >* *}*

	CONTOH
	
	If (umur > 20) {
	
	System.out.println (“Anda layak mengundi. ”) ;
	
	} else
	
	System.out.println (“Anda tidak layak mengundi. ”) ;
	}

1.4 1.4.1 MENERANGKAN ALIRAN STRUKTUR KAWALAN PILIHAN

KAWALAN PILIHAN *if-else-if*

- Untuk membuat keputusan yang lebih kompleks.
- Mencuba syarat Boolean yang baharu sekiranya syarat terdahulu menghasilkan keputusan palsu.
- Sekiranya syarat Boolean menghasilkan keputusan benar, pernyataan akan dilaksanakan. Syarat Boolean lain tidak akan diuji.

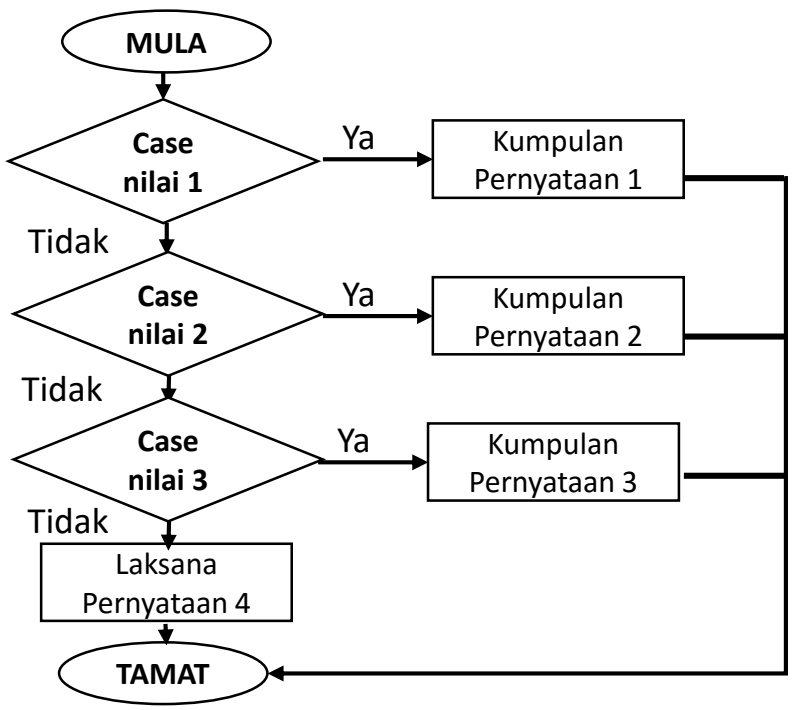
CARTA ALIR	SINTAKS
	*If (<syarat Boolean1_benar>)* *{* *Pernyataan01* *} else if (<syarat Boolean2_benar>)* *{* *Pernyataan02* *} else if (<syarat Boolean3_benar>)* *{* *Pernyataan03* *} else* *{* *Pernyataan04* *}*

1.4 1.4.1 MENERANGKAN ALIRAN STRUKTUR KAWALAN PILIHAN

KAWALAN PILIHAN *switch-case*

- Untuk mengatasi kekurangan penggunaan *if-else-if* (pernyataan perlu diulang banyak kali dan boleh mengelirukan pengguna).
- Struktur ini lebih mudah difahami.
- *Ujian Switch* : ungkapan nombor, abjad atau rentetan.
- *Case* : Mengandungi nilai yang akan dipadankan dengan ujian switch.
- *Break* : Digunakan sebagai arahan untuk keluar dari blok *switch*. Jika ungkapan *break* tidak disertakan, pernyataan selepas *break* akan dilaksanakan.
- *Default* : Kadang-kadang dimasukkan sebagai langkah tambahan. Pernyataan *default* akan dilaksanakan jika ujian *switch* tidak bersamaan dengan mana-mana nilai *case.*

CARTA ALIR



SINTAKS

```
switch (ujian) {

case : nilai1 {
   Kumpulan Pernyataan 1
break ;

} case : nilai2 {
Kumpulan Pernyataan 2
break ;

} case : nilai3 {
Kumpulan Pernyataan 3
break ;

} default : {
Kumpulan Pernyataan 4
}}
```

1.4

1.4.2 MENULIS ATUR CARA MENGUNAKAN STRUKTUR KAWALAN PILIHAN DENGAN MENGGABUNGKAN OPERATOR HUBUNGAN DAN OPERATOR LOGICAL

OPERATOR

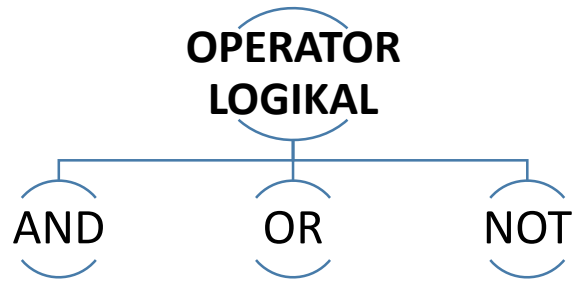
OPERATOR HUBUNGAN

- Digunakan untuk membandingkan dua nilai bagi menghasilkan keputusan Boolean.

OPERATOR HUBUNGAN	
==	sama dengan
!=	tidak sama dengan
>	lebih besar daripada
>=	lebih besar daripada atau sama dengan
<	Kurang daripada
<=	Kurang atau sama dengan

OPERATOR LOGIKAL

Digunakan untuk menghasilkan beberapa ungkapan Boolean bagi menghasilkan syarat yang lebih kompleks.



Operator Logikal AND

- Digunakan apabila dua atau lebih syarat Boolean perlu digabungkan dan kesemua syarat perlu benar.
- Ditulis dengan simbol “&&”.

Operator Logikal OR

- Digunakan apabila dua atau lebih syarat Boolean perlu digabungkan dan hanya salah satu syarat perlu benar.
- Ditulis dengan simbol “ ||”.

Operator Logikal NOT

- Menukarkan nilai Boolean kepada lawannya.
- Ditulis dengan simbol “ !”.

x	NOT x
TRUE	FALSE
FALSE	TRUE

CONTOH ATURCARA OPERATOR HUBUNGAN

CONTOH ATURCARA	OUTPUT

public class contoh31a {
 	
public static void main (String [] args) {
int nombor = 15;
 	
if (nombor > 0)
System.out.println (“Nombor ini adalah integer positif”) ;
 	
else
System.out.println (“Nombor ini adalah bukan integer positif”) ;
}
}
	Nombor ini adalah integer positif

public class contoh31b {
 	
public static void main (String [] args) {
int nombor = -7;
 	
if (nombor > 0)
System.out.println (“Nombor ini adalah integer positif”) ;
 	
else if (nombor == 0)
System.out.println (“Nombor ini adalah sifar”) ;
 	
else
System.out.println (“Nombor ini adalah integer negatif”) ;
 	
}
}
	Nombor ini adalah integer negatif

1.4

1.4.2 MENULIS ATUR CARA MENGUNAKAN STRUKTUR KAWALAN PILIHAN DENGAN MENGGABUNGKAN OPERATOR HUBUNGAN DAN OPERATOR LOGICAL

CONTOH ATURCARA OPERATOR LOGIKAL

CONTOH ATURCARA	OUTPUT

public class contoh32 {
 	
public static void main (String [] args) {
int markah = 55;
 	
if (markah >= 0 && markah <= 100)
System.out.println (“Markah yang dimasukkan adalah sah.”) ;
 	
else
System.out.println (“Markah yang dimasukkan adalah tidak sah”)
;
}
}
	Markah yang dimasukkan adalah sah.

public class contoh33 {
 	
public static void main (String [] args) {
boolean malam = true;
boolean hujan = false;
 	
if (malam || hujan)
System.out.println (“Angkat baju”) ;
}
}
	Angkat baju

public class contoh34 {
 	
public static void main (String [] args) {
boolean lulus;
int markah = 83;
 	
if (markah >= 40)
lulus = true;
else
lulus = false;
if (!lulus)
System.out.println (“Anda perlu mengulangi ujian”) ;
else
System.out.println (“Anda lulus”) ;
}
}
	Anda lulus

1.4

1.4.2 MENULIS ATUR CARA MENGUNAKAN STRUKTUR KAWALAN PILIHAN DENGAN MENGGABUNGKAN OPERATOR HUBUNGAN DAN OPERATOR LOGICAL

PENGGABUNGAN OPERATOR HUBUNGAN DAN OPERATOR LOGIKAL DALAM STRUKTUR KAWALAN PILIHAN

- Operator hubungan dan operator logikal boleh digabungkan dalam struktur kawalan pilihan.
- CONTOH : Markah >= 0 && Markah <= 100

CONTOH ATURCARA	OUTPUT

public class contoh35 {
 	
public static void main (String [] args) {
double celcius = 39 , Fahrenheit = 97 ;
 	
if (celcius > 39 || Fahrenheit > 98.6)
System.out.println (“Anda mungkin demam panas.”) ;
 	
}
}
	Anda mungkin demam panas.

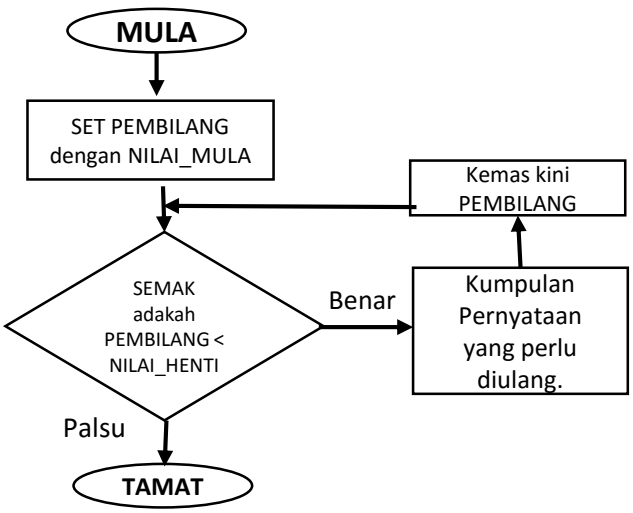
1.4 1.4.3 MENERANGKAN ALIRAN STRUKTUR KAWALAN ULANGAN

KAWALAN ULANGAN

Ulangan berasaskan pembilang (*For*)

- Untuk bilangan tertentu.
- Ditentukan oleh pemboleh ubah pembilang yang bermula dengan nombor indeks tertentu seperti 0 dan 1.
- Nombor indeks akan ditambah secara automatik pada akhir blok pernyataan.
- Penambahan dibuat setiap kali blok kenyataan telah diulang dan akan berlanjutan sehingga syarat Boolean berulang menjadi tidak benar.

CARTA ALIR



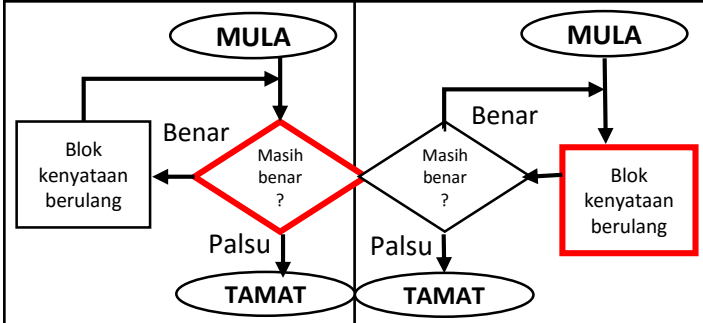
SINTAKS

```
for ( pemula ; penamat ; penambah)
{
< Blok penyataan yang perlu diulang>
}
```

Ulangan berasaskan syarat (While, Do-While)

While	*Do-While*
• Membuat ujian terlebih dahulu ke atas input. • Jika memenuhi syarat, blok arahan dalam gelung akan dilaksanakan.	• Membuat ujian selepas blok arahan dalam gelung dilaksanakan.

CARTA ALIR



SINTAKS

While (<Syarat Boolean >)
{
<Blok kenyataan berulang>
}
	Do
	<Blok kenyataan berulang>
	
	Loop While
	(<Syarat Boolean >)
	}

1.4 1.4.3 MENERANGKAN ALIRAN STRUKTUR KAWALAN ULANGAN

CONTOH ATURCARA *for*

CONTOH ATURCARA	OUTPUT

public class contoh36 {
 	
public static void main (String [] args) {
int i ;
for (i = 1 ; i <=100 ; i+=1) {
System.out.print (“i”) ;
}
}
}
	12345678910.

public class contoh37 {
 	
public static void main (String [] args) {
int jumlah = 0 ;
for (i = 2 ; i <=100 ; i+=1) {
jumlah = jumlah + i ;
}
System.out.println (“Jumlah : ” + jumlah) ;
}
}
	5049

public class contoh38 {
 	
public static void main (String [] args) {
for (i = 0 ; i <= 30 ; i+=1) {
if ((i % 2) == 2)
System.out.print (i + “, ”) ;
}
}
}
	1,3,5,7,9,11,15,17,19,23,25,27,29

public class contoh39 {
 	
public static void main (String [] args) {
double baki = 500.0;
for (i = 0 ; i <= 5 ; i+=1) {
baki = baki + (0.1 * baki);
}
System.out.printn (“ Baki 5 tahun : “ + baki) ;
}
}
	Baki 5 tahun : 805.255

CONTOH ATURCARA *while*

CONTOH ATURCARA	OUTPUT

public class contoh40 {
 	
public static void main (String [] args) {
int n = 5 ;
while (n > 0) {
System.out.println (n + " , ") ;
n = n-1;
}
}
}
	5,4,3,2,1

public class Module1 {
 	
public static void main (String [] args) {
Scanner scanner = new Scanner (System.in);
int nom = scanner.nextInt ();
 	
while (nom > 0) {
nom - = 1;
System.out.println (nom + " ") ;
}
}
}	

public class contoh42 {
 	
public static void main (String [] args) {
String strPassword = new String ();
Scanner scanner = new Scanner (System.in);
final String RekodLaluanRahsia = "Pisang" ;
 	
while (!strPassword.equal (RekodLaluanRahsia)) {
System.out.print (" Sila masukkan password : ") ;
strPassword = scanner.next();
System.out.print ( )
}
}
}
	Sila masukkan password : Pisang

CONTOH ATURCARA *do-while*

CONTOH ATURCARA	OUTPUT

public class contoh43 {
public static void main (String [] args) {
int no = 1 ;
do {
System.out.println (no + " X 3 = " + no * 3) ;
no = no + 1;
} while (n <= 12)
}
}
	1 X 3 = 3
	2 X 3 = 6
	3 X 3 = 9
	4 X 3 = 12
	5 X 3 = 15
	6 X 3 = 18
	7 X 3 = 21
	8 X 3 = 24
	9 X 3 = 27
	10 X 3 = 30
	11 X 3 = 33
	12 X 3 = 36

public class contoh44 {
public static void main (String [] args) {
double no ;
double sum;
int counter = 1;
Scanner scanner = new scanner (System.in);
do {
no = scanner.nextDouble();
sum += no ;
counter = counter + 1;
} while (counter <= 5)
System.out.println (" The total is " + sum) ;
}
}
	6
	7
	9
	3
	6
	The total is 31.0

public class contoh45 {
public static void main (String [] args) {
string input ;
Scanner scanner = new scanner (System.in);
int no1;
int no2;
do {
no1 = scanner.nextInt();
no2 = scanner.nextInt();
System.out.println (" Beza = " + Math.abs (no1 –no2)) ;
System.out.println ("Taip YA untuk teruskan") ;
input = scanner.next();
} while (input.equal("YA")) ;
}
}
	5
	7
	Beza = 2
	Taip YA untuk teruskan
	YA
	5
	9
	Taip YA untuk teruskan
	t

1.4 1.4.4 MENULIS ATUR CARA MENGGUNAKAN STRUKTUR KAWALAN ULANGAN YANG MELIBATKAN - *OPERATOR INCREMENT DAN DECREMENT*

OPERATOR *INCREMENT (++)* DAN *DECREMENT (--)*

- Lazimnya digunakan sebagai pembilang.
- **Operator Increment (++)** : penambahan nilai pemboleh ubah bagi bilangan nombor tertentu.
- **Operator Decrement (--)** : mengurangkan nilai pemboleh ubah bagi bilangan nombor tertentu.

OPERATOR INCREMENT (++)

UNGKAPAN	MAKNA	CONTOH

i += 1	i = i + 1	while (<syarat Boolean>) {
		<Blok kenyataan berulang>
		<Kemaskini nilai dalam syarat>
		}

i += 2	i = i + 2	
i += 3	i = i + 3	

OPERATOR DECREMENT (--)

UNGKAPAN	MAKNA	CONTOH

i -= 1	i = i – 1	while (<syarat Boolean>) {
		<Blok kenyataan berulang>
		<Kemaskini nilai dalam syarat>
		}

i -= 2	i = i – 2	
i -= 3	i = i - 3	

Math.random [ ]

- Math.random [ ] ialah subaturcara java untuk menjana nombor secara rambang.
- Menggunakan waktu sistem sebagai nilai benih untuk memulakan penjanaan nombor secara rambang.
- CONTOH :
  (int) (Math.random()*10) + 1 *(nombor rambang 1 hingga 10)*

MENJANA 20 NOMBOR RAMBANG BAGI DADU

CONTOH ATURCARA

```
public class contoh46 {
    public static void main (String [] args) {
        int i ;
        Scanner scanner = new Scanner (System.in);
        Boolean flag = true;
            do {
                for (i = 1; i <=20 ; i++ ) {
                    System.out.println ((int) (Math.random() * 6) + 1)  +  “   ” )  ;
                }
                System.out.println () ;
                System.out.println (“ ---------------------------------”);
                System.out.println (“ Taip ya untuk teruskan, tidak untuk henti”);
                    if (Scanner.next().equals(“ya”)) {
                        flag = true;
                    else {
                        flag = false;
                    }
        } while ( flag)
    }
}
```

BENDERA BOOLEAN

- Satu pemboleh ubah Boolean digunakan untuk mengawal ulangan.
- Pengguna ditanya untuk meneruskan atur cara itu lagi atau tidak.

MENJANA 20 NOMBOR RAMBANG BAGI DADU

CONTOH ATURCARA

```
public class contoh47 {
    public static void main (String [] args) {
        int i ;
        int diceNo;
        int count1 =0, count2 =0, count3 =0, count4 =0, count5 =0, count6 =0;
       Scanner scanner = new Scanner (System.in);
        Boolean flag = true;
            do {
                for (i = 1; i <=20 ; i++ ) {
                    diceNo = (int) (Math.random() * 6 + 1) ;
                    System.out.println ( diceNo + "   " )  ;
                    Switch (diceNo) {
                        case 1 : count1++ ; break;
                        case 2 : count2++ ; break;
                        case 3 : count3++ ; break;
                        case 4 : count4++ ; break;
                        case 5 : count5++ ; break;
                        case 6 : count6++ ;
                     }
                System.out.println () ;
                System.out.println (" * Dice Number 1 =  " + count1  +  " % ");
                System.out.println (" * Dice Number 2 =  " + count2  +  " % ");
                System.out.println (" * Dice Number 3 =  " + count3  +  " % ");
                System.out.println (" * Dice Number 4 =  " + count4  +  " % ");
                System.out.println (" * Dice Number 5 =  " + count5  +  " % ");
                System.out.println (" * Dice Number 6 =  " + count6  +  " % ");
                System.out.println () ;
                System.out.println (" ----------------------------------");
                System.out.println (" Taip ya untuk teruskan, tidak untuk henti");
                   if (Scanner.next().equals("ya")) {
                       flag = true;
                   else {
                        flag = false;
                   }
        } while ( flag)
    }
```

1.4

1.4.4 MENULIS ATUR CARA MENGGUNAKAN STRUKTUR KAWALAN ULANGAN YANG MELIBATKAN *-BOOLEAN*

PEMBILANG

- Digunakan untuk membuat pengiraan dalam penyelesaian masalah.

KEKERAPAN NOMBOR DADU DALAM 100 LAMBUNGAN

CONTOH ATURCARA

```
import java.util.Scanner;
public class dadurambang {
    public static void main (String [] args) {
        int i ;
        int diceNo;
        int Kira1 =0, Kira2 =0, Kira3 =0, Kira4 =0, Kira5 =0, Kira6 =0;
        double percen1 = 0 , percen2 = 0 , percen3 = 0 , percen4 = 0 , percen5 = 0 , percen6 = 0;
        int jumlahKiraan = 0;
        Scanner scanner = new Scanner (System.in);
        Boolean flag = true;
            do {
                for (i = 1; i <=20 ; i++ ) {
                    diceNo = (int) (Math.random() * 6 + 1) ;
                    switch (diceNo) {
                       case 1 : Kira1++ ; break;
                       case 2 : Kira2++ ; break;
                       case 3 : Kira3++ ; break;
                       case 4 : Kira4++ ; break;
                       case 5 : Kira5++ ; break;
                       case 6 : Kira6++ ;
                     }
                 }
               jumlahKiraan = Kira1 + Kira2 + Kira3 + Kira4 + Kira5 +  Kira6;

               percen1 = (double) Kira1/jumlahKiraan * 100;
               percen2 = (double) Kira2/jumlahKiraan * 100;
               percen3 = (double) Kira3/jumlahKiraan * 100;
               percen4 = (double) Kira4/jumlahKiraan * 100;
               percen5 = (double) Kira5/jumlahKiraan * 100;
               percen6 = (double) Kira6/jumlahKiraan * 100;

               System.out.println () ;
               System.out.println (" Nombor  1 dadu = " + percen1  +  " % ");
               System.out.println (" Nombor  2 dadu = " + percen2  +  " % ");
               System.out.println (" Nombor  3 dadu = " + percen3  +  " % ");
               System.out.println (" Nombor  4 dadu = " + percen4  +  " % ");
               System.out.println (" Nombor  5 dadu = " + percen5  +  " % ");
               System.out.println (" Nombor  6 dadu = " + percen6  +  " % ");

               System.out.println () ;
               System.out.println (" ---------------------------------");
               System.out.println (" Taip ya untuk teruskan");
                  if (!scanner.next().equals("ya")) {
                      flag = true;
                  }
        } while ( flag)
    }
}
```

1.5 AMALAN TERBAIK PENGATURCARAAN

- **AMALAN TERBAIK** : Teknik atau methodologi yang telah dibuktikan melalui suatu pengalaman atau kajian yang boleh dipercayai, untuk mendapatkan hasil yang diinginkan.
- Satu komitmen untuk menggunakan semua pengetahuan dan teknologi yang ada untuk memastikan keberhasilan yang baik.
- **AMALAN TERBAIK PENGATURCARAAN** : Apabila pengaturcara dapat mempraktikkan amalan-amalan yang biasa diikuti untuk menghasilkan atur cara yang baik.

INDEN YANG KONSISTEN

- Inden yang konsisten membuatkan kod atur cara mudah dibaca dan difahami oleh pengguna lain.
- Cara menulis inden konsisten dari mula hingga akhir kod.

JENIS DATA

- Pilih jenis data yang bersesuaian supaya saiz pemboleh ubah tidak terlampau kecil atau besar dan memulihara sumber.

PEMBOLEH UBAH YANG BERMAKNA

- Tidak bermula dengan nombor. CTH : 1cara
- Tiada ruang kosong antara perkataan. Gunakan underscore atau rapatkan perkataan. CTH : cara_1 / cara1.
- Tidak sama dengan kata kekunci. CTH : integer, double.
- Penggunaan huruf besar dan huruf kecil. CTH : caraPertama tidak sama dengan CaraPertama.
- Nama yang bermakna dan mudah difahami. Penggunaan singkatan tidak digalakkan.
- Tidak menggunakan perkataan rezab khas. CTH : print,value.

KOMEN

- Ditulis dengan jelas dalam dua hingga tiga baris pendek untuk menerangkan fungsi kod dan memenuhi ruang lajur pengekodan.

1.5

1.5.1 MEMBEZAKAN JENIS RALAT DALAM ATUR CARA

JENIS RALAT ALGORITMA

- RALAT SINTAKS
- RALAT LOGIK
- RALAT MASA LARIAN

RALAT SINTAKS

- Kesalahan tatabahasa seperti salah ejaan atau tatatanda.
- Penggunaan objek atau aksara yang tidak dikenali.

RALAT LOGIK

- Berlaku apabila atur cara tidak berfungsi seperti yang diingini.
- Jarang atau tidak dapat dikesan oleh pengkompil.
- Hanya pengaturcara yang boleh mengesan melalui output yang dihasilkan.
- Pengatur cara perlu memeriksa semua aspek output projek.

RALAT MASA LARIAN

- Ralat yang ditemui ketika aturcara yang sedang berjalan terganggu akibat beberapa faktor.
- Berlaku sekiranya pengatur cara cuba melaksanakan operasi aritmetik yang mustahil.

1.5 1.5.2 MENGESAN, MENGENALPASTI, MENTERJEMAH MESEJ RALAT DAN MEMBAIKI RALAT

Semak semula atur cara pada bahagian pengisytiharan

Pastikan semua tatatanda ditaip dengan lengkap

Pastikan nama pemboleh ubah yang diisytiharkan adalah sama dengan nama yang yang akan dipanggil balik dalam atur cara (semak ejaan & penggunaan huruf besar/kecil)

Baiki ralat yang dikenalpasti

1.5 1.5.3 MENGENALPASTI NILAI BAGI PEMBOLEH UBAH PADA BAHAGIAN TERTENTU ATUR CARA

PEMBOLEH UBAH	INPUT	OUTPUT
Item Pemboleh Ubah		
Nilai (Data Pemboleh Ubah)		

1.5

1.5.4 MENGHASILKAN ATUR CARA YANG MUDAH DIBACA DENGAN MENGGUNAKAN GAYA YANG BAIK

<table>
<tr><th></th><th></th></tr>
<tr><td>KOMEN</td><td>

Penanda yang dibuat oleh pengatur cara untuk setiap atur cara yang di bina.
Dalam Java , kod komen perlu mengikut sintaks yang ditetapkan untuk mengelak ralat sintaks.

<table>
<tr><th>JENIS KOMEN</th><th>HURAIAN</th></tr>
<tr><td>//</td><td>Pengkompil mengabaikan semua teks bermula dengan // hingga teks terakhir ayat yang sama.</td></tr>
<tr><td>/* */</td><td>Pengkompil mengabaikan semua teks yang berada dalam /* hingga ke /* walaupun pada baris berlainan.</td></tr>
<tr><td>/** */</td><td>Komen dokumentasi. Pengkompil mengabaikan komen ini sama seperti komen /*.</td></tr>
</table>
</td></tr>
<tr><td>PEMBOLEH UBAH YANG BERMAKNA</td><td>

Nama pemboleh ubah yang mempunyai ejaan yang ringkas dan bermakna.

</td></tr>
<tr><td>INDEN</td><td>

Merujuk kepada cara penulisan atur cara yang memudahkan pembacaan.
Pembacaan atur cara akan dimulakan dengan inden iaitu barisan teks berada di beberapa kedudukan aksara ke dalam, dari jidar kiri atau kanan halaman.

</td></tr>
</table>

1.5 1.5.4 MENGHASILKAN ATUR CARA YANG MUDAH DIBACA DENGAN MENGGUNAKAN GAYA YANG BAIK

CONTOH

```
/* Program ringkas untuk mengira Luas Segitiga
Langkah 1: Baca Tapak
Langkah 2: Baca Tinggi
Langkah 3: Hitung Luas
Langkah 4: Papar Luas
*/*/
public class LuasSegitiga {
    public static void main (String[] args){
        /**Pengisytiharan pemboleh ubah input dan output*/
        int Tapak = 6;
        int Tinggi = 4;
        double Luas;

              // Proses yang terlibat dalam penghitungan
              // luas segitiga
              Luas = (1.0 / 2) * Tapak * Tinggi;

        //Paparan output
        System.out.println ("Luas Segitiga ialah : " + Luas);
    }
}
```

Komen

Komen

Komen

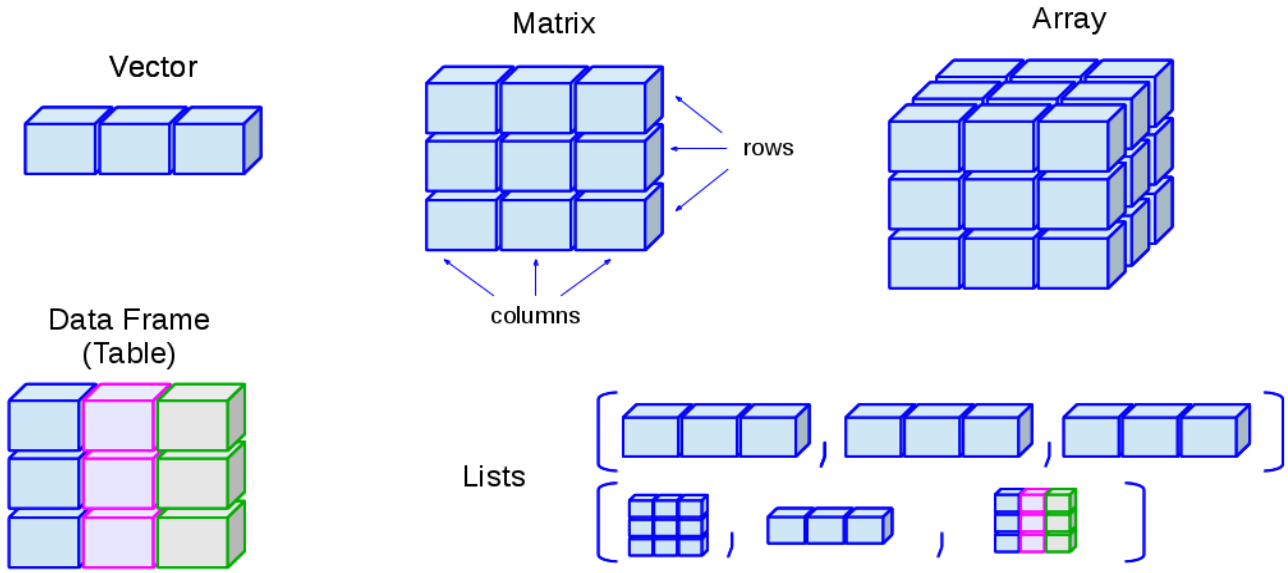
Komen

```
/* Program menghitung luas segitiga*/
public class LuasSegitiga {
    public static void main (String[] args){
        /**Pengisytiharan pemboleh ubah input dan output*/
        int Tapak = 6;
        int Tinggi = 4;
        double Luas;

              // Proses yang terlibat dalam penghitungan
              // luas segitiga
              Luas = (1.0 / 2) * Tapak * Tinggi;

        //Paparan output
        System.out.println ("Luas Segitiga ialah : "
        + Luas);
    }
}
```

Inden

Pemboleh ubah yang bermakna

1.6 STRUKTUR DATA DAN MODULAR



- **STRUKTUR DATA** : Satu kaedah tertentu untuk menyimpan secara tersusun data-data dalam ingatan supaya senang dicapai untuk diproses menjadi maklumat mengikut kehendak pengguna.

- Data boleh disusun dalam bentuk tatasusunan (*Array*) dan vector (*vector*), senarai pautan (*linked list),* timbunan (*stack)* dan giliran (*queue*).

- Gunakan struktur yang sistematik untuk pemboleh ubah dan arahan semasa membangunkan atur cara.

- Pemboleh ubah boleh dipecahkan kepada *"kumpulan-kumpulan kecil"* yang dipanggil **TATASUSUNAN.**

- Arahan-arahan komputer juga boleh dipecahkan kepada *kumpulan-kumpulan kecil* yang dipanggil **FUNGSI.**

- Apabila diperlukan sahaja, kumpulan yang berkaitan akan dipanggil.

1.6

1.6.1 MENERANGKAN STRUKTUR TATASUSUNAN (ARRAY) SATU DIMENSI

TATASUSUNAN

- **TATASUSUNAN** ialah pemboleh ubah yang membolehkan koleksi beberapa nilai data (elemen) dalam satu-satu masa dengan menyimpan setiap elemen dalam ruang memori berindeks.
- Pemboleh ubah ialah slot memori yang telah dikhaskan untuk menyimpan data.
- Kebiasaanya, pemboleh ubah mudah cuma menyimpan satu nilai data dalam satu-satu masa.

Pengisytiharan Tatasusunan

```
jenisData [ ] namaTatasusunan;
namaTatasusunan = new jenisData [ saizTatasusunan];
```

Contoh :

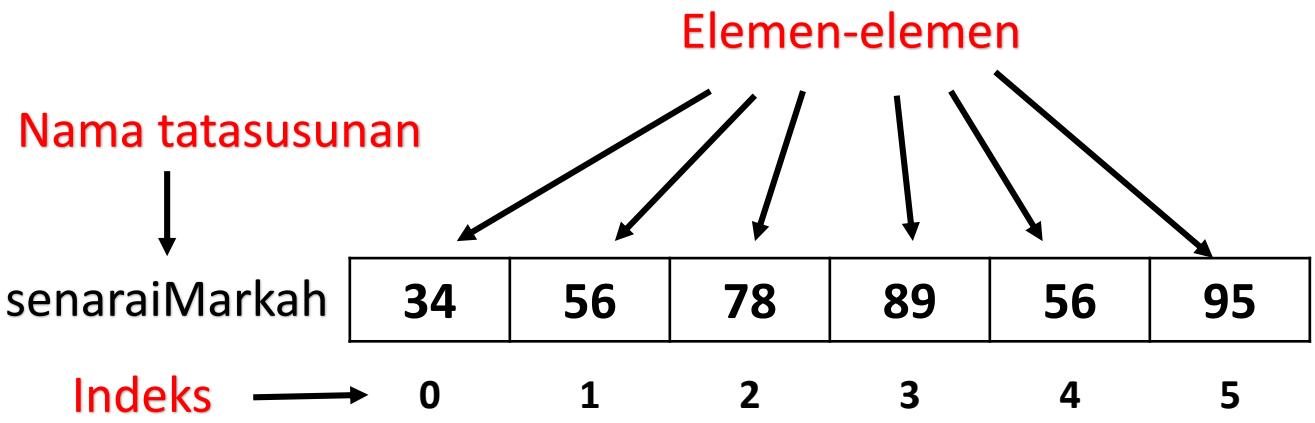
```
int [ ] senaraiMarkah ;
senaraiMarkah = new int [ 6 ];
```

Pengumpukan Tatasusunan

- Pengisytiharan tatasusunan menyediakan ruang memori yang masih kosong.
- Nilai perlu diberikan melalui proses pengumpukan.
- Selepas diisytihar, nilai diumpuk dengan memanggil elemen –elemen tatasusunan satu-persatu.

Contoh :

```
senaraiMarkah [ 0 ]  = 34 ;
senaraiMarkah [ 1 ]  = 56 ;
senaraiMarkah [ 2 ]  = 78 ;
senaraiMarkah [ 3 ]  = 89 ;
senaraiMarkah [ 4 ]  = 56 ;
senaraiMarkah [ 5 ]  = 95 ;
```



Pengumpukan Nilai Awal Tatasusunan

Contoh :

```
int senaraiMarkah [ ]  = { 34,56,78,89,56,95};
```

- Umpukan dibuat ketika melakukan pengisytiharan.
- Saiz dalam tatasusunan tidak perlu dimasukkan dalam tanda [ ] .
- Saiz tatasusunan ditentukan secara automatik berdasarkan bilangan data dalam kurungan { }.
- Semua data yang hendak disimpan ialah satu jenis yang sama.

Perbezaan Struktur Memori antara Pemboleh Ubah Mudah dengan Memori Tatasusunan.

Pemboleh Ubah Mudah
int markah1 = 56, markah2 = 78, markah3 = 34;
Tatasusunan
int markah [ ] = { 56,78,34};

1.6

1.6.1 MENERANGKAN STRUKTUR TATASUSUNAN (ARRAY) SATU DIMENSI

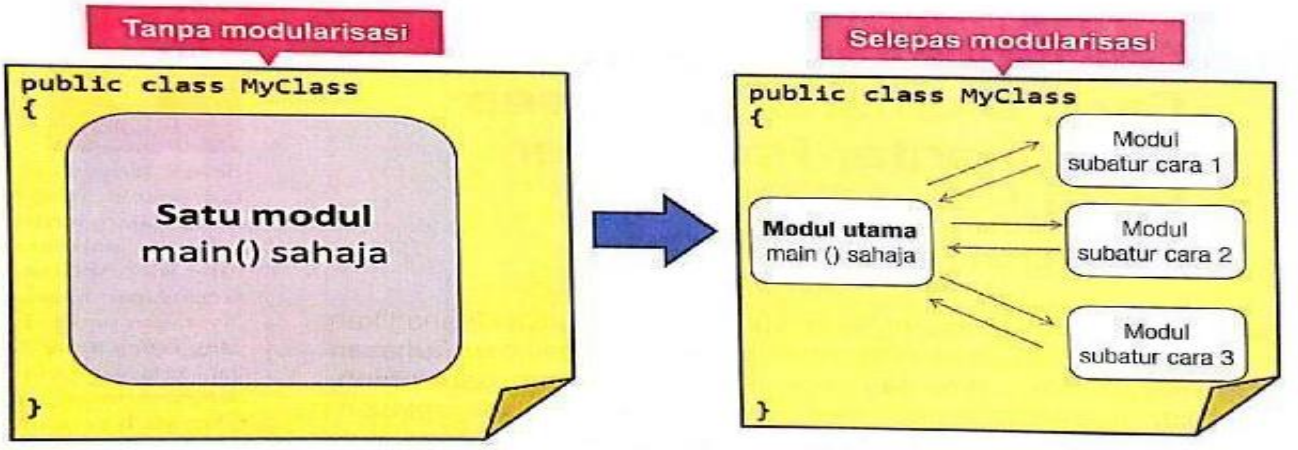
ATURCARA YANG MENGGUNAKAN TATASUSUNAN

CONTOH ATURCARA

```
package perisianSaya;
public class MyClass {
    public static void main (String [] args) {

            String [] senaraiNama = new String[4];

            senaraiNama [0] = “Adam”;
            senaraiNama [1] = “Alia”;
            senaraiNama [2] = “Wong”;
            senaraiNama [3] = “Devi”;

            int [ ] senaraiUmur = {16,17,16,17};

            double [] senaraiTinggi = {182.1,172.5,173.2,175.0}

             System.out.println (“ NAMA\tUmur\tTinggi(cm)”);

              for (int i = 0; i<4;i++) {
                    System.out.print (senaraiNama [i] + “\t”);
                    System.out.print (senaraiUmur [i] + “\t”);
                    System.out.print (senaraiTinggi [i] + “\t”);

                 System.out.println () ;
            }
        }
}
```

1.6

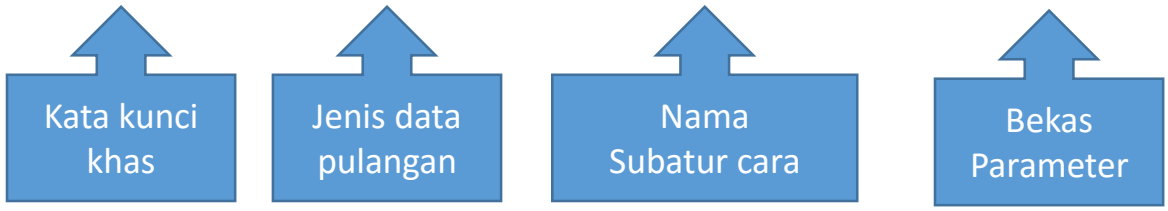
1.6.2 MENGGUNAKAN SUBATUR CARA DAN MEMAHAMI KONSEP MENGHANTAR PARAMETER KE SUBATUR CARA DAN MENGEMBALIKAN DATA



- Fail kod komputer yang panjang adalah sukar ditulis, dibaca, diulangkaji atau dikemaskini.
- Oleh itu, baris-baris kod komputer yang berkait boleh dihimpunkan dalam satu **modul.**
- Dengan itu, kod komputer yang panjang dapat dibahagi-bahagikan kepada modul-modul.
- Setiap modul adalah lebih pendek dan mengkhususkan kepada tujuan tertentu.
- Modul-modul ini dipanggil **subatur cara,** struktur untuk himpunan kod komputer.

KOMPONEN *HEADER* SUBATUR CARA

```
static  void  subAtur01  ( )
```

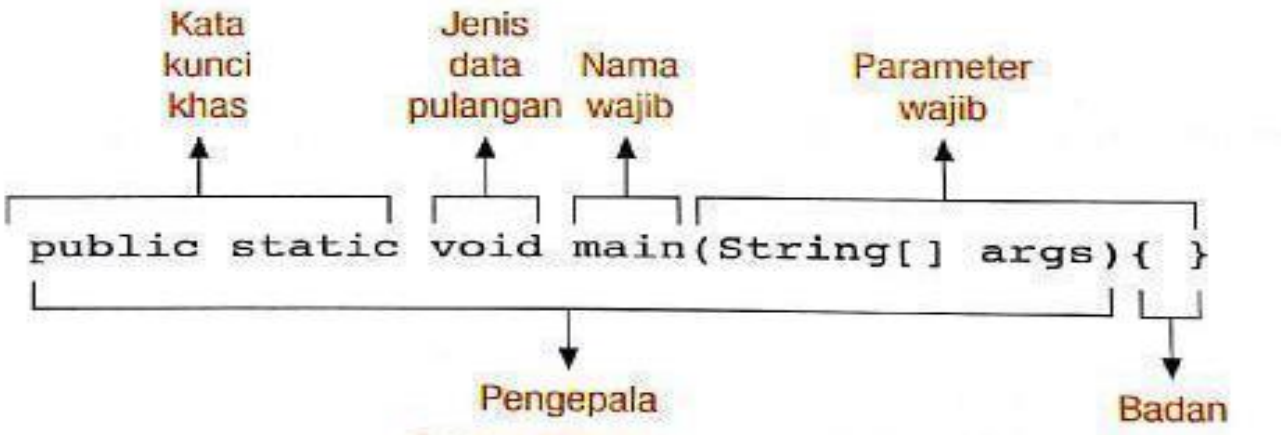


Kata Kunci Khas	• Kata kunci *static* diletakkan dihadapan nama subatur cara. • Tanpa kata kunci ini, subatur cara tidak dapat digunakan secara langsung. • Tanpa static, subatur cara memerlukan penghasilan objek sebelum data digunakan.
Jenis Data Pulangan	• Subatur cara biasanya akan memulangkan hasil setelah badan subatur cara selesai memproses data. • Jenis data pulangan di *header* bergantung kepada jenis data yang ingin dipulangkan oleh *body*. • Ini termasuk ***int, double, string*** dan ***char.*** • Jika tidak ada keperluan memulangkan data, gunakan kata kunci ***void.***
Nama Subatur Cara	• Diberikan oleh pengatur cara. • Mestilah bermula dengan huruf (biasanya huruf kecil) . • Boleh mengandungi angka tetapi bukan symbol.
Bekas Parameter	• Simbol () digunakan jika parameter kosong. • Nama parameter akan dikepilkan jika bekas menerima parameter. • CONTOH : (int kuantiti)

1.6 | **1.6.2** | **MENGGUNAKAN SUBATUR CARA DAN MEMAHAMI KONSEP MENGHANTAR PARAMETER KE SUBATUR CARA DAN MENGEMBALIKAN DATA**

SUBATUR CARA main ( )

- Subatur cara wajib dengan nama ***main ().***
- boleh wujud tanpa subatur cara yang lain.
- mengandungi baris pertama pernyataan yang mesti dilaksanakan oleh komputer.
- mengandungi baris terakhir pernyataan yang mesti dilaksanakan oleh komputer.
- *Header* subatur cara jarang diubah.
- Pernyataan-pernyataan dalam subatur cara ***main ()*** akan menentukan sifat atur cara.
- Pernyataan-pernyataan ini seharusnya ditulis berasaskan algoritma yang telah diuji.



Kata Kunci Khas	• Public membolehkan subatur cara diakses dari mana-mana kod projek • Semua subatur cara mempunyai static supaya dapat digunakan secara langsung tanpa objek..
Jenis Data Pulangan	• Kata kunci ***void*** digunakan kerana tidak memulangkan data.
Nama Subatur Cara	• Nama wajib ialah **main.**
Bekas Parameter	• Bekas parameter mesti mengandungi parameter tatasusunan string dengan nama “args”.

1.6

1.6.2 MENGGUNAKAN SUBATUR CARA DAN MEMAHAMI KONSEP MENGHANTAR PARAMETER KE SUBATUR CARA DAN MENGEMBALIKAN DATA

SUBATUR CARA main ( )

```
1  package perisianSaya;
2  public class MyClass {
3
4      public static void main(String[] args){
5          System.out.println("Hello Malaysia.");
6      }
7
8  }
9
10
```

Subatur cara main()

```
C:\WINDOWS\system32\cmd.exe
D:\Java Projects\HelloMalaysia\dist>java -jar HelloMalaysia.jar
Hello Malaysia.
```

Output

SUBATUR CARA LAIN

```
static  void  subAtur01  ( )
```

- Kata kunci khas
- Jenis data pulangan
- Nama Subatur cara
- Bekas Parameter

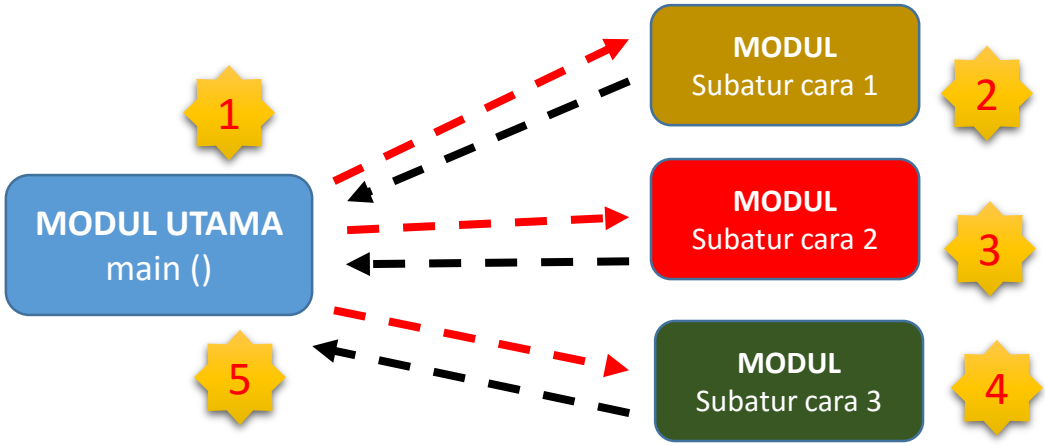
- Pengatur cara boleh menulis subatur cara yang lain – dipanggil **petakrifan subatur cara.**
- Subatur cara lain adalah serupa dengan **main()** tetapi lebih ringkas.

MEMANGGIL SUBATUR CARA main ( )

- Subatur cara boleh menggunakan subatur cara lain.
- Tujuannya supaya kod pernyataan dalam subatur cara lain turut dilaksanakan.
- Hubungan dua subatur cara – *pemanggil* dan *dipanggil.*
- *Pemanggil* memanggil nama *subatur cara dipanggil* dalam badan subatur cara badan pemanggil.

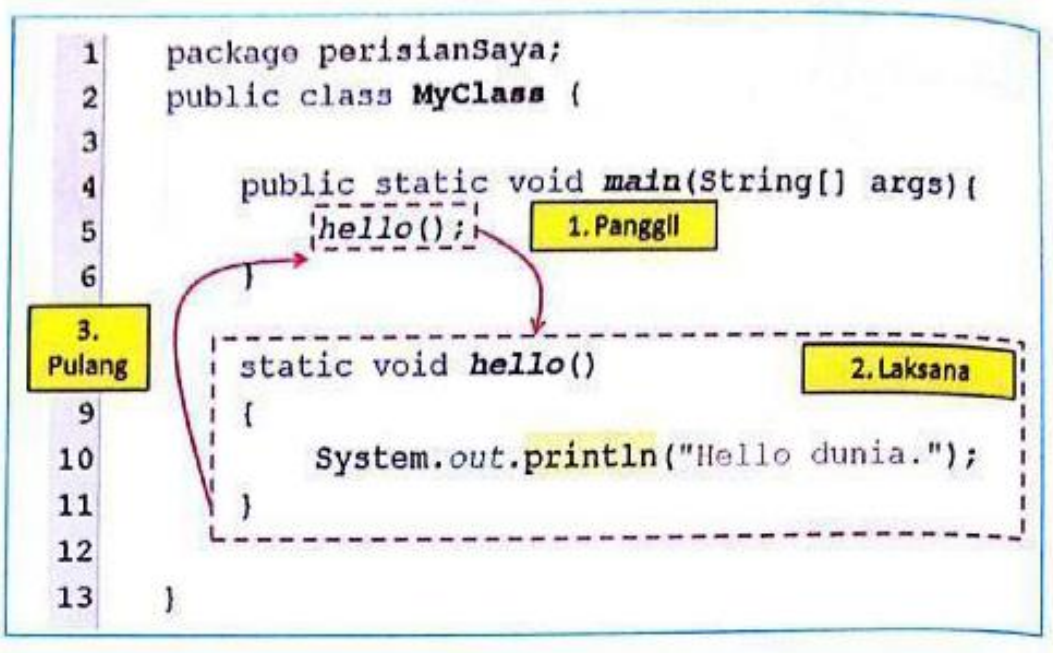
MEMANGGIL SUBATUR CARA DARIPADA main ( )

- Kebiasaannya, subatur cara main () menggunakan subatur cara-subatur cara yang lain.
- Apabila main() memerlukan bantuan subatur cara lain untuk proses tertentu , kawalan dipindahkan kepada subatur cara tersebut.
- Setelah subatur cara tersebut selesai, kawalan dikembalikan kepada subatur cara main ().



1.6

1.6.2 MENGGUNAKAN SUBATUR CARA DAN MEMAHAMI KONSEP MENGHANTAR PARAMETER KE SUBATUR CARA DAN MENGEMBALIKAN DATA



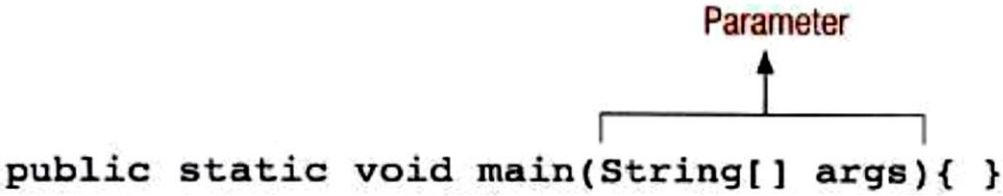
CONTOH ATURCARA

```
public class contohpg148 {
    static java.util.Scanner
    scanner = new java.util.Scanner (System.in);

    public static void main (String [] args) {
            mintaNama ();
            mintaMarkah();
    }

    public static void mintaNama(){
          String nama;
          System.out.println ("Masukkan Nama : ");
          nama = scanner.nextLine();
           System.out.println ("Terima Kasih" + nama);
      }

     public static void mintaMarkah() {
            int markah;
            System.out.println ("Masukkan Markah : ");
            markah = scanner.nextInt();
            System.out.println ("Markah Anda : " + markah);
           }
}
```

1.6

1.6.2 MENGGUNAKAN SUBATUR CARA DAN MEMAHAMI KONSEP MENGHANTAR PARAMETER KE SUBATUR CARA DAN MENGEMBALIKAN DATA

PARAMETER



- Parameter ataupun argumen ialah pemboleh ubah yang membolehkan subatur cara menerima nilai daripada pemanggil.
- Dengan ini, subatur cara- subatur cara masih dapat berkongsi nilai-nilai pemboleh ubah melalui parameter.
- Parameter rasmi (formal parameter) : merujuk kepada parameter subatur cara.
- Parameter sebenar (actual parameter) : merujuk kepada pemboleh ubah di dalam subatur cara pemanggil.
- Penggunaan parameter perlu diisytiharkan sewaktu pentakrifan subatur cara- subatur cara.
- Jika parameter diperlukan, parameter perlu diisytiharkan dalam kurungan bekas parameter dalam subatur cara.
- Pengisyiharan parameter sama seperti pengisytiharan pemboleh ubah.
- Tiada had untuk bilangan parameter dan turutan parameter bergantung kepada pengatur cara.

TIADA PARAMETER	MENGANDUNGI PARAMETER
static void subAtur01 ( ) { }	static void subAtur01 (int x) { }
static void subAtur02 ( ) { }	static void subAtur02 (int x ; double y) { }
static void subAtur03 ( ) { }	static void subAtur03 (int [ ] x ; string z) { }

1.6

1.6.2 MENGGUNAKAN SUBATUR CARA DAN MEMAHAMI KONSEP MENGHANTAR PARAMETER KE SUBATUR CARA DAN MENGEMBALIKAN DATA

CONTOH ATURCARA

```
static void kuasaDua (int nom) {
      double jawapan = nom*nom;
      System.out.print (jawapan);
}
```

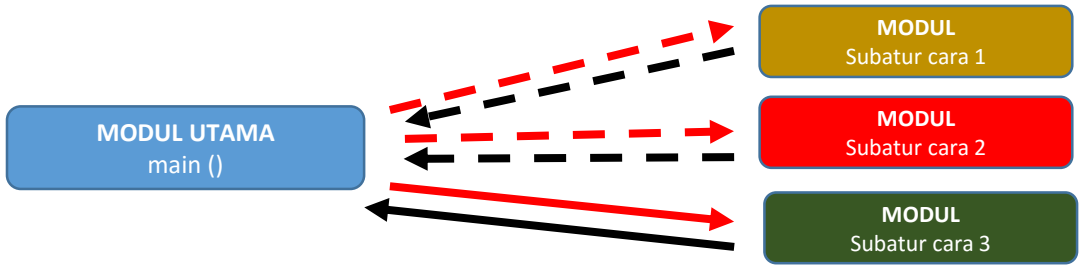
```
static void cariJumlah (int x, int y, int z) {
     int jawapan = x + y + z;
      System.out.print (jawapan);
}
```

```
public static void main (String [ ] args)  {
     cariJumlah (1,2,7);
    }

static void cariJumlah (int x, int y, int z) {
     int jawapan = x + y + z;
      System.out.print (jawapan);
}
```

```
public static void main (String [ ] args)  {
     paparHarga (“telefon pintar”, 1500.0);
    }

static void paparHarga (string item, double h) {
      System.out.print (“ Harga “ + item “ adalah “ +h);
}
```

MENGEMBALIKAN DATA

- Semua subatur cara akan mengembalikan kawalan kepada pemanggil.
- Sesetengah subatur cara bukan sahaja mengembalikan kawalan tetapi juga data.
- Bagi yang memulangkan kawalan sahaja, kata kunci ***void*** digunakan.
- Sekiranya subatur cara mengembalikan data, baris akhir dalam badan subatur cara mempunyai pernyataan ***return*** dan data yang ingin dipulangkan kepada pemanggil.
- Nama subatur cara tidak mempunyai void sebagai jenis data pulangan. Sebaliknya, jenis data yang bersesuaian dengan data dipulang diisytihar dalam sintaks.
- Data boleh dipulangkan sebagai hasil ungkapan, nilai dalam pemboleh ubah, nilai pemalar ataupun nilai data itu sendiri.
- Setiap subatur cara cuma boleh mengembalikan satu jenis data sahaja.
- Nama subatur cara yang memulangkan data biasanya diberikan *prefix get*. Contoh : getNama, getAlamat, getTelefon.



DEFINISI SUBATUR CARA	PENJELASAN
static void subAturcara ( ) { }	Tidak memulangkan data. Cuma kawalan dipulangkan.
static int subAturcara ( ) { }	Memulangkan data jenis integer.
static double subAturcara ( ) { }	Memulangkan data jenis double.
static string subAturcara ( ) { }	Memulangkan data jenis string.

CONTOH ATURCARA

```
public class contohpg153 {
  static java.util.Scanner scanner = new java.util.Scanner (System.in);

  public static void main (String[] args){
    String nama, alamat, telefon;

    nama = getNama();
    alamat = getAlamat(nama);
    telefon = getTelefon(nama);

    System.out.println();
    System.out.println(" Terima Kasih, Sila semak , ");
    System.out.println(" Nama : " + nama);
    System.out.println(" Alamat : " + alamat);
    System.out.println(" Telefon : " + telefon);
  }

  static String getNama (){
    System.out.print(" Masukkan nama : ");
    return scanner.nextLine();
  }

  static String getAlamat (String n){
    System.out.print(n + " , Sila Masukkan alamat : ");
    return scanner.nextLine();
  }

  static String getTelefon (String n){
    System.out.print(n + " , Sila Masukkan telefon : ");
    return scanner.nextLine();
  }
}
```

1.6

1.6.3 MEMBEZAKAN ANTARA *FUNCTION* DAN *PROCEDURE* PADA SUBATUR CARA

FUNGSI

- Data yang telah diproses perlu disimpan dalam badan pemanggil.
- Data yang telah diproses akan digunakan lagi oleh pemanggil.
- Data tersebut mungkin digunakan oleh subatur cara main () atau subatur cara-subatur cara lain.
- Subatur cara digunakan untuk meminta input.
- Subatur cara tidak memaparkan hasil atau menyimpan hasil di mana-mana.

PROSEDUR

- Subatur cara digunakan untuk membuat paparan sahaja, seperti mesej kepada pengguna.
- Hasil proses digunakan dalam subatur cara sekali sahaja dan tidak diperlukan lagi.

ASPEK PERBANDINGAN	FUNGSI	PROSEDUR
Persamaan	Mengembalikan kawalan	Mengembalikan kawalan
Perbezaan	• Mengembalikan data. • Badan diakhiri dengan pernyataan ***return*** diikuti data yang dipulangkan.	• Tidak Mengembalikan data. • Badan **tidak diakhiri** dengan pernyataan *return.*
Jenis data pulangan	int, double , char,string, tatasusunan atau objek java.	void
Sintaks definisi	static jenisData namaFungsi ([jenisData namaParameter]){}	static void namaProsedur ([jenisData namaParameter]){}

Contoh definisi	static int cariJumlah (int x, int y){
	int jawapan = x + y;
	return jawapan;
	}
		static void cariJumlah (int x, int y)
		{
		int jawapan = x + y;
		System.out.print (jawapan);
		}

Sintaks panggilan	jenisData pembolehUbah; pembolehUbah = namaFungsi ([jenisData namaParameter]);	namaProsedur ([jenisData namaParameter]);
Contoh panggilan	int jumlah = cariJumlah (5,8);	cariJumlah (5,8);

1.6

1.6.3 MEMBEZAKAN ANTARA *FUNCTION* DAN *PROCEDURE* PADA SUBATUR CARA

CONTOH ATURCARA	
FUNGSI	**PROSEDUR**

static int mintaNombor ( ) {
int nom;
java,util.Scanner sc;
sc = new Java.util.Scanner (System,.in);
nom = sc.nextInt();
Return nom
}
	static void hello () {
	System.out.print (“Hello dunia.”);
	}

int jumlahNombor (int x, int y) {
Int jumlah;
Jumlah = x + y;
Return jumlah;
}
	static void hello (String nama) {
	System.out.print (“Hello “ + nama);
	}

	static void cariJumlah (int x, int y)
	{
	Int jawapan = x + y;
	System.out.print (jawapan);
	}

1.6

1.6.3 MEMBEZAKAN ANTARA *FUNCTION* DAN *PROCEDURE* PADA SUBATUR CARA

CONTOH ATURCARA

```
public class contoh61 {
  static java.util.Scanner scanner = new java.util.Scanner (System.in);

  public static void main (String [] args){
    String x;
    x = getNama ();
    System.out.println ("Salam sejahtera, " + x);
  }

  static String getNama () {
    System.out.print ("Masukkan nama : ");
    return scanner.nextLine();
  }}
```

```
public class contoh62{
  static java.util.Scanner sc = new java.util.Scanner (System.in);

  public static void main (String []args){
      double nom1,nom2;
      System.out.println("Masukkan nombor 1: ");
      nom1 = sc.nextInt();
      System.out.println("Masukkan nombor 2: ");
      nom2 = sc.nextInt();
      System.out.println (Math.max(nom1,nom2) + " adalah lebih besar");
}
}
```

CONTOH FUNGSI-FUNGSI UTILITI DALAM JAVA

SINTAKS FUNGSI	PENJELASAN
Math.sqrt (double n)	Memulangkan hasil punca kuasa dua untuk nilai n. **CONTOH :** Math.sqrt (100);
Math.floor(double n)	Memulangkan interger paling dekat tetapi kurang atau sama dengan nilai n. **CONTOH :** Math.floor (2.7); *[memulangkan 2]* Math.floor (-2.7); *[memulangkan -3]*
Math.round (double n)	Memulangkan nombor n stelah dibulatkan kepada interger terdekat. **CONTOH :** Math.floor (2.7); *[memulangkan 3]* Math.floor (2.3); *[memulangkan 2]*
Math.max (double m , double n)	Memulangkan nombor yang lebih besar antara m dan n **CONTOH :** Math.max(100,10); *[memulangkan 100]*
Math.min (double m , double n)	Memulangkan nombor yang lebih kecil antara m dan n **CONTOH :** Math.max(100,10); *[memulangkan 10]*

1.6

1.6.4 MENULIS ATUR CARA BERMODULAR YANG MENGANDUNGI STRUKTUR TATASUSUNAN

```
1  package perisianSaya;
2  public class MyClass {
3
4      static String nama;
5      static int markah;
6      static String gred;
7      static java.util.Scanner scanner = new java.util.Scanner(System.in);
8
9      public static void main(String[] args){
10
11         getNama();
12         getMarkah();
13         setGred();
14         paparInfo();
15     }
16
```

1

```
static void getNama() {
    System.out.print("Masukkan nama:");
    nama = scanner.nextLine();
}
```

2

```
static void getMarkah() {
    System.out.print("Masukkan markah, " + nama + ":");
    markah = scanner.nextInt();
}
```

3

```
static void setGred() {
    if(markah>=90){gred = "A+";}
    else if(markah>=80){gred = "A";}
    else if(markah>=75){gred = "A-";}
    else if(markah>=70){gred = "B+";}
    else if(markah>=65){gred = "B";}
    else if(markah>=60){gred = "C+";}
    else if(markah>=50){gred = "C";}
    else if(markah>=45){gred = "D";}
    else if(markah>=40){gred = "E";}
    else {gred = "F";}
}
```

4

```
static void paparInfo() {
    System.out.print("Terima kasih, " + nama + ". ");
    System.out.println("Gred untuk " + markah + " adalah " + gred);
}
```

1.6 1.6.4 MENULIS ATUR CARA BERMODULAR YANG MENGANDUNGI STRUKTUR TATASUSUNAN

```
1  package perisianSaya;
2  public class MyClass {
3
4      static String nama;
5      static int markah;
6      static String gred;
7      static java.util.Scanner scanner = new java.util.Scanner(System.in);
8
9      public static void main(String[] args){
10
11         getNama();
12         getMarkah();
13         setGred();
14         paparInfo();
15     }
16
```

- Atur cara bagi contoh diatas ialah atur cara ***console.***
- Pemboleh ubah (*nama, gred, markah*) diisyihar sebagai **Pemboleh ubah sejagat (global)** – boleh dicapai oleh semua subatur cara.
- Prosedur **main(), getNama(),getMarkah(), setGred() dan paparInfo()** dapat membaca dan menulis kepada set pemboleh ubah-pe,boleh ubah yang sama.
- **Prosedur main()** – tidak terdapat banyak pernyataan algoritma kerana pernyataa-pernyataan tersebut telah diletakkan ke dalam subatur cara.
- Memanggil subatur cara harus mengikut urutan logik.

CONTOH ATURCARA

```
public class contohpg159 {
  static String nama;
  static int markah;
  static String gred;
  static java.util.Scanner scanner = new java.util.Scanner (System.in);

  public static void main (String[]args){
    getNama();
    getMarkah();
    setGred();
    paparInfo();
  }

    static void getNama(){
      System.out.print ("Masukkan nama : ");
      nama = scanner.nextLine();
   }

    static void getMarkah() {
      System.out.print ("Masukkan markah , " + nama + " : " );
      markah = scanner.nextInt();
    }

    static void setGred() {
      if (markah >=90) {gred = "A+";}
      else if (markah >=80) {gred = "A";}
      else if (markah >=75) {gred = "A-";}
      else if (markah >=70) {gred = "B+";}
      else if (markah >=65) {gred = "B";}
      else if (markah >=60) {gred = "C+";}
      else if (markah >=50) {gred = "C";}
      else if (markah >=45) {gred = "D";}
      else if (markah >=40) {gred = "E";}
      else {gred = "F";}
        }

    static void paparInfo(){
      System.out.print ("Terima Kasih," +nama + " . ");
      System.out.print ("Gred untuk " + markah + " adalah " +gred);
    }
  }
```

STRUKTUR TATASUSUNAN DALAM ATUR CARA BERMODULAR

CONTOH ATURCARA

```
public class contoh63 {

  public static void main (String[] args){
    int[] senaraiNombor = {1,2,3,4,5,6,7,8,9,10};
    paparSenarai(senaraiNombor);
  }

    static void paparSenarai (int[] senaraiNombor){
      System.out.print ("Senarai nonbor dalam subatur cara : ");

     for (int i=0;i<10;i++)
        {System.out.print (senaraiNombor [i] + " , ");
         }
}
}
```

- Tatasusunan digunakan sebagai parameter untuk bilangan data yang banyak.
- Pastikan subatur cara mampu menerima parameter tatasusunan.
- Pengisytiharan dilakukan pada kepala subatur cara.
- Struktur kawalan ulangan *for* diperlukan untuk mengumpuk atau mengakses nilai elemen-elemen dalam tatasusunan.
- Elemen-elemen dalam satu-satu tatasusunan boleh diubah terus dari mana-mana subatur cara.
- Oleh itu, pemboleh ubah tatasusunan tidak perlu dikembalikan kepada pemanggil.

STRUKTUR TATASUSUNAN DALAM ATUR CARA BERMODULAR

CONTOH ATURCARA

```
public class contoh64 {

  public static void main (String[] args){
    int[] senaraiNombor = new int[10];
    setSenaraiRawak (senaraiNombor);
    System.out.println ("\n\nDalam subatur cara main :");
      for (int i=0;i<10;i++)
        {System.out.print (senaraiNombor [i] + " , ");
         }
  }

    static void setSenaraiRawak(int[] senaraiNombor){
      System.out.print ("Dalam subatur cara setSenaraiRawak  : ");
      for (int i=0;i<10;i++){
      senaraiNombor[i] =  (int)(Math.random()*10)+1;
        {System.out.print (senaraiNombor [i] + " , ");
}}
}
}
```

- Tatasusunan nombor yang kosong diisytiharkan dalam prosedur **main().**
- Dari **main()**, panggilan dibuat kepada prosedur setSenaraiRawak.
- Dalam prosedur setSenaraiRawak, tatasusunan kosong diumpukkan dengan nilai-nilai rawak yang dijanakan oleh **math.random.**
- Nilai-nilai elemen dipaparkan pada kedua-dua subatur cara dan prosedur **main().**

1.7

PEMBANGUNAN APLIKASI

KITAR HAYAT PEMBANGUNAN SISTEM

Software Development Life Cycle (SDLC)

- Juga dikenali sebagai Kitar Hayat Pembangunan Aplikasi.
- Istilah yang digunakan dalam kejuruteraan sistem dan perisian, sistem maklumat dan pembangunan aplikasi.
- Menjelaskan tentang proses merancang, mereka bentuk, menguji dan mengimplimentasi sesuatu aplikasi atau perisian.
- Terdiri daripada satu kitaran fasa berjujukan dan menjadikannya sebagai pelan tindakan yang berkesan kepada pasukan projek.
- SDLC membantu mengesan status bagi penyempurnaan projek tersebut.

Methodologi Umum SDLC

- **MODEL AIR TERJUN** (Waterfall)
- **MODEL RAD** (Rapid Application Development)
- **MODEL LELARAN** (Iterative)
- **MODEL LINGKARAN** (Spiral)
- **MODEL TANGKAS** (Agile)
- **MODEL HIBRID** (Kombinasi Model)

1.7

1.7 MENGHURAIKAN SETIAP FASA DALAM KITARAN HAYAT PEMBANGUNAN SISTEM (SDLC)

MODEL AIR TERJUN

- Digunakan sebagai model pembangunan aplikasi memandangkan model ini mirip kepada proses-proses dalam SDLC.
- Merupakan model yang terawal,mudah difahami dan mudah diuruskan.
- Mengandungi 5 fasa.
- Satu fasa perlu diselesaikan sebelum ke fasa seterusnya.
- Maklumat bagi setiap fasa diperlukan untuk fasa yang berikutnya dan tidak boleh berpatah balik.

Fasa Analisis Masalah

- Proses mengenal pasti keperluan program dan mencari sebab sesuatu program dibina.
- Langkah-langkah sistematik harus dipatuhi untuk menyelesaikan masalah dan penyting untuk kita memahami pernyataan masalah dengan jelas.
- Menggunakan analisis/carta IPO.

Tentukan penyelesaian yang dikehendaki.

↓

Kenalpasti formula untuk menghasilkan output.

↓

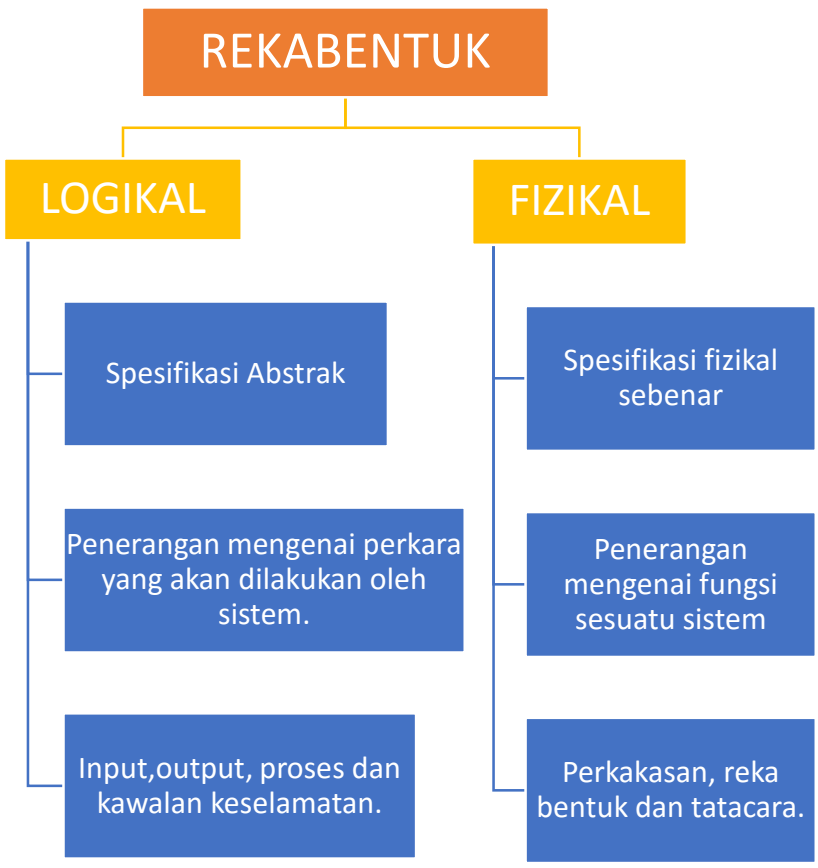
Kenalpasti input bagi formula.

Contoh CARTA IPO

INPUT	Kadar sejam , Bilangan Jam bekerja
PROSES	1. Dapatkan kadar sejam 2. Dapatkan bilangan jam bekerja. 3. Kirakan Gaji Staf = kadar sejam X bilangan jam bekerja
OUTPUT	Gaji Staf

Fasa Reka bentuk Penyelesaian

- Dibuat setelah analisis IPO.
- Fasa ini melihat kepada potensi penyelesaian yang wujud dan menentukan penyelesaian yang efektif dan efisien.
- Membina penyelesaian terbaik.
- **ALGORITMA** : Langkah awal penyelesaian masalah.
- **PSEUDOKOD :** Aturan langkah yang ditulis dalam Bahasa pertuturan.
- **CARTA ALIR :** Perwakilan grafik yang menunjukkan langkah penyelesaian sesuatu masalah dan mempunyai hubung kait antara satu sama lain.

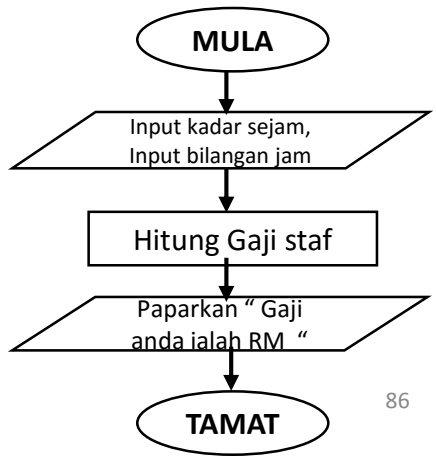


CONTOH

PSEUDOKOD

1.Mula
2. Baca kadar sejam (kadarsj)
3. Baca bilangan jam bekerja (bilJam)
4.Hitung gaji staf = kadarsj * bilJam
5. Paparkan gaji staf.
6. Tamat

CARTA ALIR



1.7

1.7.1 MENGHURAIKAN SETIAP FASA DALAM KITARAN HAYAT PEMBANGUNAN SISTEM (SDLC)

Fasa Pelaksanaan Penyelesaian

- **TUJUAN :** mengubah reka bentuk kepada program yang akan dipasang pada perkakasan dan bersedia melaksanakan penyelesaian.
- Membina dan menghasilkan sistem yang dapat menyelesaikan masalah yang dihadapi.
- Aktiviti pembangunan aplikasi melibatkan pengekodan.
- **PENGEKODAN :** Memerlukan Bahasa pengaturcaraan seperti Java, Javascript dan sebagainya. – mengubah spesifikasi program kepada kod sumber.
- **PENGKOMPILAN :** Proses menukar kod pengaturcaraan kepada kod boleh laksana (executable).

CONTOH ATURCARA

```
public class contohpg172 {
  public static void main (String [] args){
    int bilJam = 20;
    double gajistaf, kadarsj;
    kadarsj = 25.00;

    gajistaf = kadarsj * bilJam;

    System.out.println (" Gaji anda ialah RM " +gajistaf);
  }
}
```

1.7

1.7.1 MENGHURAIKAN SETIAP FASA DALAM KITARAN HAYAT PEMBANGUNAN SISTEM (SDLC)

Fasa Uji dan Nyah ralat

- Memastikan semua keperluan dipenuhi
- Memastikan semua pengekodan berfungsi seperi yang dikehendaki.
- Memastikan semua modul boleh berfungsi bila digabungkan.
- Memastikan maklum balas dari pengguna sistem untuk tujuan pembetulan dan penambahbaikan.
- Melibatkan pengguna sepenuhnya di peringkat pembangunan.
- Mengesan ralat yang tercicir.
- Membantu pasukan projek membuat dokumentasi dengan mengesan kesilapan oleh pengguna.
- Menyimpan keputusan ujian sebagai bukti penyempurnaan pembangunan sistem.
- **SEMAKAN KOD (Code Review) :** Dilakukan untuk mengesan ralat. Pengatur cara akan merujuk log yang dipaparkan untuk membetulkan dan membuang ralat yang dikesan.

Jenis-jenis semakan

JENIS SINTAKS	SIAPA ?	BILA ?
Sendiri	Pengarang	Semasa pengekodan.
Rakan Sebaya	Rakan Sebaya	Selepas tamat modul
Selepas tamat modul.	Pasukan projek yang diketuai oleh pakar Bahasa pengaturcaraan.	Selepas kedua-dua peringkat diatas.

1.7

1.7.1 MENGHURAIKAN SETIAP FASA DALAM KITARAN HAYAT PEMBANGUNAN SISTEM (SDLC)

Jenis pengujian dan perincian

JENIS PENGUJIAN

SENDIRI	SISTEM	INTEGRASI	PENERIMAAN
Memastikan setiap unit dalam sistem yang dibina berfungsi.	Menguji sistem secara keseluruhan selepas ujian atas unit individu.	Memastikan sistem dapat berfungsi dengan sistem sedia ada.	Ujian ini dijalankan semasa ujian integrasI oleh pengguna sistem untuk memastikannya memenuhi kehendak pengguna.

Contoh

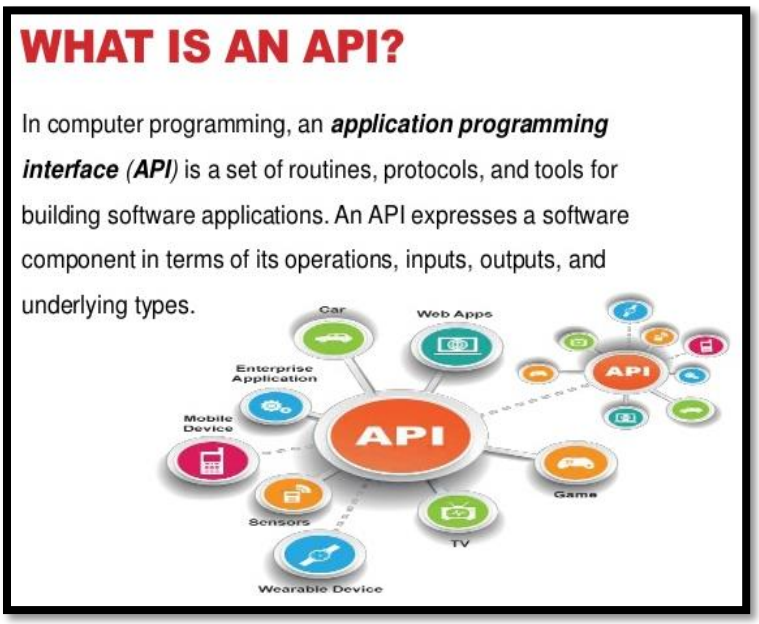
ITEM	AKTIVITI	TANDAKAN (/) ATAU (X)
Ralat Sintaks	Ejaan teks nama pemboleh ubah	
	Ejaan teks komen	
	Penggunaan objek atau aksara yang tidak dikenali	
	Pengisytiharan jenis data untuk bilangan jam bekerja.	
	Pengisytiharan jenis data untuk gaji staf.	
	Pengisytiharan jenis data kadar sejam	
Ralat Masa Larian	Input pengiraan untuk bilangan jam	
	Input pengiraan untuk kadar sejam	
Ralat Logik	Semak output gaji staf.	

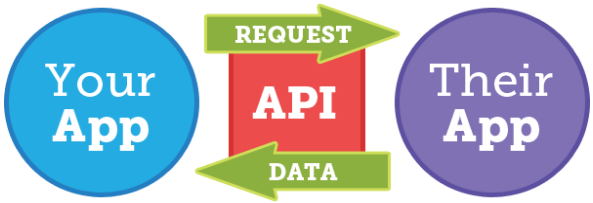
1.7

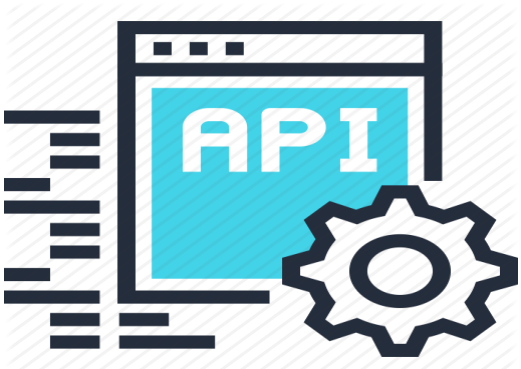
1.7.1 MENGHURAIKAN SETIAP FASA DALAM KITARAN HAYAT PEMBANGUNAN SISTEM (SDLC)

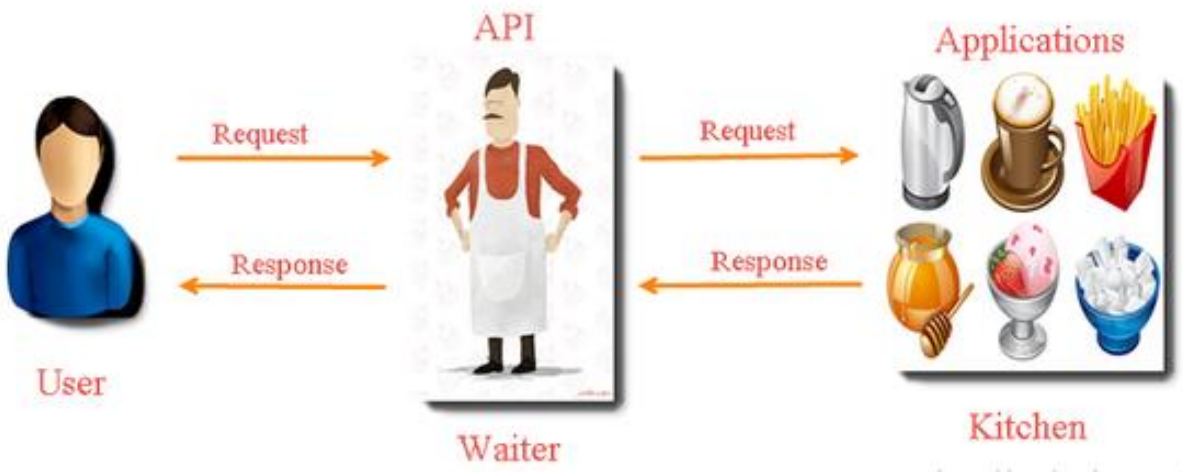
Application Programming Interface (API)

- Satu set rutin , protocol dan alat untuk membina aplikasi.
- Sesuatu API menentukan bagaimana komponen aplikasi harus berinteraksi.
- API yang baik memudahkan pembangunan aplikasi dengan menyediakan blok pembangunan, di mana pengatur cara komputer akan mencantumkan blok-blok tersebut.
- Contoh : API Google Maps, API Twitter.









1.7

1.7.1 MENGHURAIKAN SETIAP FASA DALAM KITARAN HAYAT PEMBANGUNAN SISTEM (SDLC)

Fasa Dokumentasi

- Satu proses mengutip dan mengumpulkan data, mengumpulkan maklumat dan ringkasan seperti laporan pengujian yang dijalankan, carta alir, kod atur cara dan juga carta IPO.
- Dokumen-dokumen ini penting untuk rujukan pengguna sistem, pegawai IT dan kakitangan baharu di setiap fasa.
- Dokumentasi yang sepenuhnya bagi fasa projek dari awal pada setiap fasa akan dijadikan rujukan untuk fasa seterusnya.

```
int bilJam = 20; //Pengisytiharan pemboleh ubah
double gajistaf, kadarsj; //Pengisytiharan pemboleh ubah
kadarsj = 25.0; // Mendapatkan data kadar bayaran sejam
```

(b) Carta Gantt untuk pembangunan aplikasi bagi mengira gaji staf

Senarai Aktiviti	20 Mac	21 Mac	22 Mac	23 Mac	24 Mac	25 Mac	26 Mac	27 Mac	28 Mac
1. Menyediakan borang maklumat staf									
2. Mendapatkan maklumat bagi kerja yang dilakukan oleh staf									
3. Membina algoritma untuk mengira gaji staf									
4. Membina kod atur cara									
5. Menjalankan pengujian dan nyah ralat									
6. Menambahbaik dan membetulkan ralat									
7. Menggunakan atur cara – gaji staf									
8. Dokumentasi									

(c) Pengujian dan nyah ralat

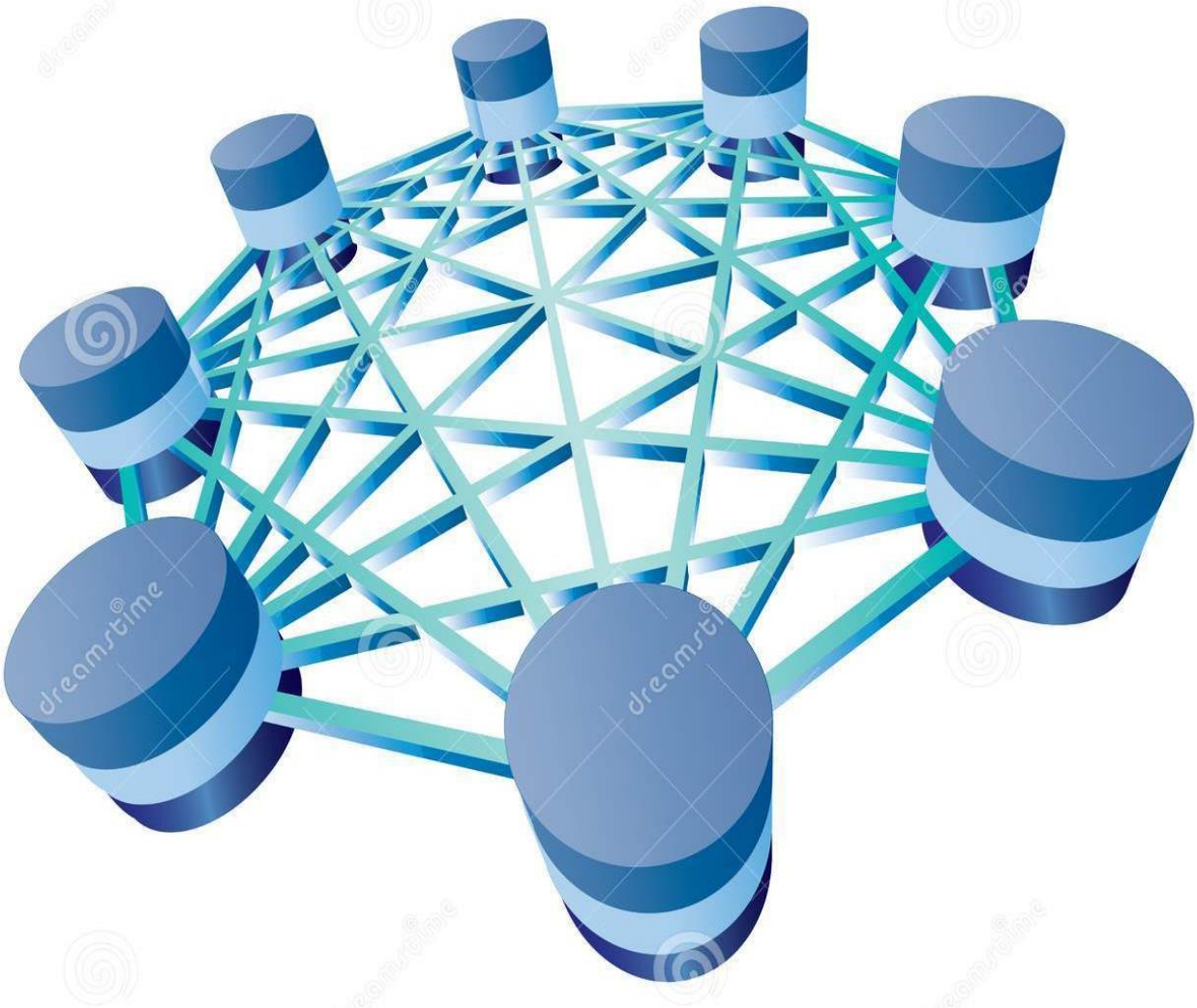
Nama	Jenis	Penerangan	Catatan
bilJam	integer	Bilangan jam bekerja	Tidak boleh angka negatif
kadarsj	*double*	Kadar sejam	Dinyatakan dalam RM
gajistaf	*double*	Hasil darab bilangan jam bekerja dengan kadar sejam	Dinyatakan dalam RM

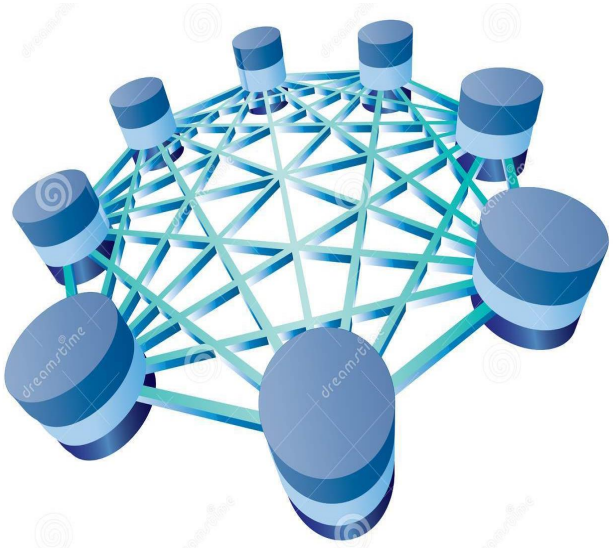
(d) API

Nama Staf	Input: Bilangan jam bekerja	Input: Kadar Sejam	Output: Gaji Staf	Catatan
A101	20	25.00	500.00	

2.0

PANGKALAN DATA





2.0 PANGKALAN DATA

- 2.1 Pangkalan Data Hubungan
- 2.2 Reka Bentuk Pangkalan data Hubungan
- 2.3 Pembangunan Pangkalan Data Hubungan.
- 2.4 Pembangunan Sistem Pangkalan Data

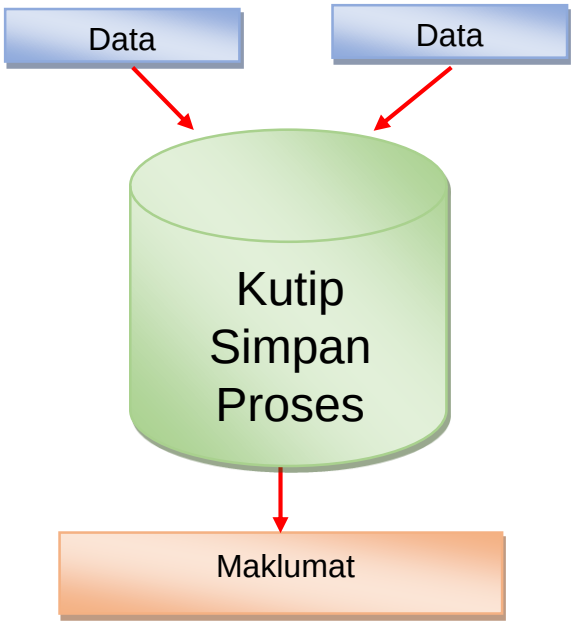
2.1 PANGKALAN DATA HUBUNGAN

- Boleh terdiri daripada perkataan, angka atau gambar.
- Boleh menjadi fakta bermakna atau tidak sehingga fakta itu diproses.
- CONTOH: Markah Ujian bulanan
- Himpunan fakta mentah mengenai sesuatu benda, kejadian, orang atau entiti.

DATA

- Maklumat menjadi lebih bermakna dan mudah difahami.
- CONTOH: Purata markah
- Hasil daripada pengumpulan, pemprosesan dan penganalisisan data.

MAKLUMAT



2.1 2.1.1 MENJELASKAN KEPENTINGAN INTEGRITI DATA, KETEKALAN DATA DAN KELEWAHAN DATA

- Pangkalan Data merupakan satu koleksi data yang disimpan dalam format piawaian (*standard*) dan direka khusus supaya mampu untuk berkongsi data dengan banyak pengguna secara efisien.
- Format piawai membolehkan data disimpan dan dicapai kembali daripada mana-mana perkakasan atau sistem maklumat.
- Data yang disimpan dalam pangkalan data boleh digunakan kembali oleh pengguna yang berbeza melalui sistem yang berbeza untuk tujuan yang berlainan.
- Simpanan data dalam pangkalan data adalah efisien kerana format dan struktur data hubungan dikhususkan untuk simpanan sistematik dan capaian pantas secara fleksibel.
- Data yang baik diperlukan untuk menghasilkan maklumat yang bermakna.

CIRI-CIRI SIMPANAN DATA

- INTEGRITI
- KETEKALAN
- KELEWAHAN

2.1 | **2.1.1** | **MENJELASKAN KEPENTINGAN INTEGRITI DATA, KETEKALAN DATA DAN KELEWAHAN DATA**

INTEGRITI DATA (*DATA INTEGRITY*)

INTEGRITI DATA

- Kesempurnaan, ketepatan dan kesahan data (*validation*) serta merujuk kepada ketepatan data yang sah melalui keseluruhan kitar hayat data.
- Untuk memastikan integriti data, input perlu tepat dan mematuhi peraturan yang telah ditetapkan.

INTEGRITI ENTITI
- Jadual mempunyai sekurang-kurangnya satu atribut data yang unik untuk setiap rekod.
- CONTOH : Nombor murid mesti unik.

INTEGRITI RUJUKAN
- Rekod-rekod dalam dua jadual yang berbeza boleh dihubungkan melalui perkongsian atribut data yang sama.

INTEGRITI DOMAIN
- Atribut-atribut data dalam jadual mestilah betul dengan berasaskan suatu domain masalah.

CONTOH

Jika seorang murid didaftarkan dengan nombor pendaftaran murid 123/16, pangkalan data sepatutnya tidak membenarkan pendaftaran murid lain dengan nombor pendaftaran yang sama.

KETEKALAN DATA (*DATA CONSISTENCY*)

KETEKALAN DATA

- Merujuk kepada konsistensi atau keseragaman data yang akan mempengaruhi kebolehpercayaan data.
- Proses kemas kini salinan data di semua lokasi simpanan perlu dilakukan dengan rapi.

KEPENTINGAN KETEKALAN DATA

- Memastikan data boleh dipercayai
- Memastikan tiada isu data yang sama berulang di beberapa lokasi simpanan

CONTOH

Tan mencatat nombor-nombor telefon pelanggannya di dalam dua buah buku iaitu diari dan buku nota secara berasingan. Sekiranya nombor telefon pelanggannya bertukar, Tan perlu mengemaskinikan nombor tersebut pada kedua-dua buah buku yang digunakannya.

Apakah yang terjadi sekiranya Tan hanya mengemaskinikan nombor telefon pelanggan yang berubah hanya pada sebuah buku sahaja ?

Disebabkan Tan hanya mengemaskini nombor telefon pelanggannya pada sebuah buku sahaja, maka data nombor telefon tersebut hilang ketekalannya.

KELEWAHAN DATA (*DATA REDUNDANCY*)

KELEWAHAN DATA

- Merujuk kepada pertindihan data yang berpunca daripada salinan data yang berulang tetapi di lokasi yang berlainan.

CONTOH

Kelewahan data

IC Murid	Nama Murid	No. Telefon
000405-11-5434	Alia bt Roslan	03-61402987
000213-03-5675	Ah Chong	03-77876789
000607-14-4343	Suraj A/L Ramu	03-43543456
000405-11-5434	Alia bt Roslan	03-61402997
001121-14-2312	Tan Mei Ling	03-69104356
000405-11-5434	Alia bt Roslan	03-61403997

Rajah 2.6 Kelewahan data dalam sistem fail

KESAN JIKA KELEWAHAN TIDAK DIATASI

- Kelewahan menggunakan lebihan sumber simpanan komputer.
- Kemaskini data yang lewah memakan banyak masa dan mungkin tidak menyeluruh
- Data menjadi korup
- Pangkalan data menjadi kurang efisien

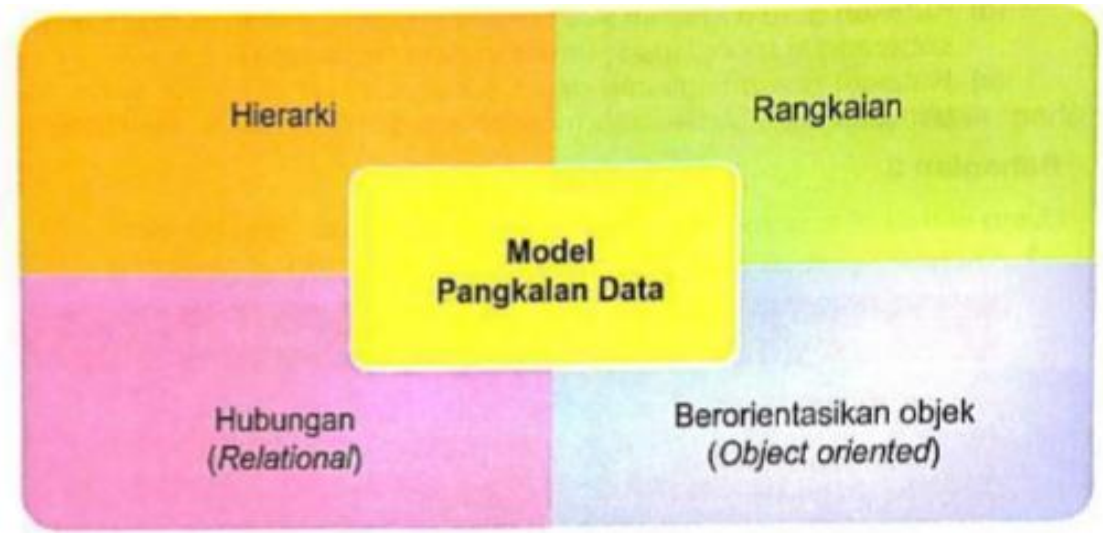
TUJUAN ELAK KELEWAHAN

- Elak pertambahan saiz pangkalan data yang tidak perlu.
- Elak data yang tidak konsisten.
- Elak data jadi tidak efisien.
- Elak data jadi korup.

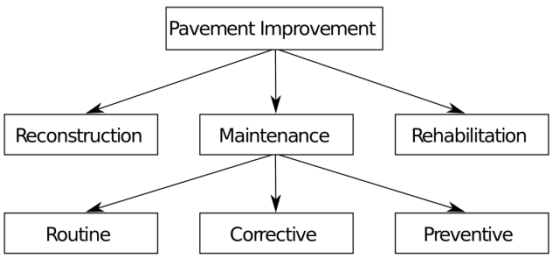
2.1 2.1.2 MEMBEZAKAN MODEL-MODEL PANGKALAN DATA

MODEL PANGKALAN DATA

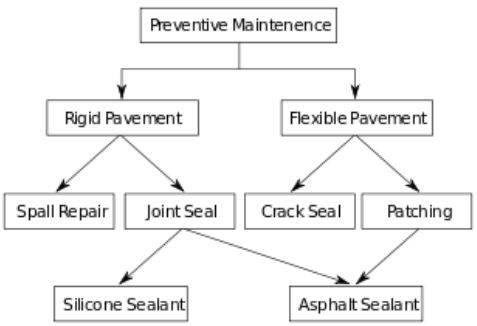
- Penyusunan secara konsepsual suatu pangkalan data dan merupakan satu cara mentakrif dan menggunakan data dalam satu pangkalan data.
- Terdapat 4 model pangkalan data dengan kelebihan masing-masing.



Hierarchical Model



Network Model



Relational Model

Activity Code	Activity Name
23	Patching
24	Overlay
25	Crack Sealing

Key = 24

Activity Code	Date	Route No.
24	01/12/01	I-95
24	02/08/01	I-66

Date	Activity Code	Route No.
01/12/01	24	I-95
01/15/01	23	I-495
02/08/01	24	I-66

Object-Oriented Model

Object 1: Maintenance Report

Date	
Activity Code	
Route No.	
Daily Production	
Equipment Hours	
Labor Hours	

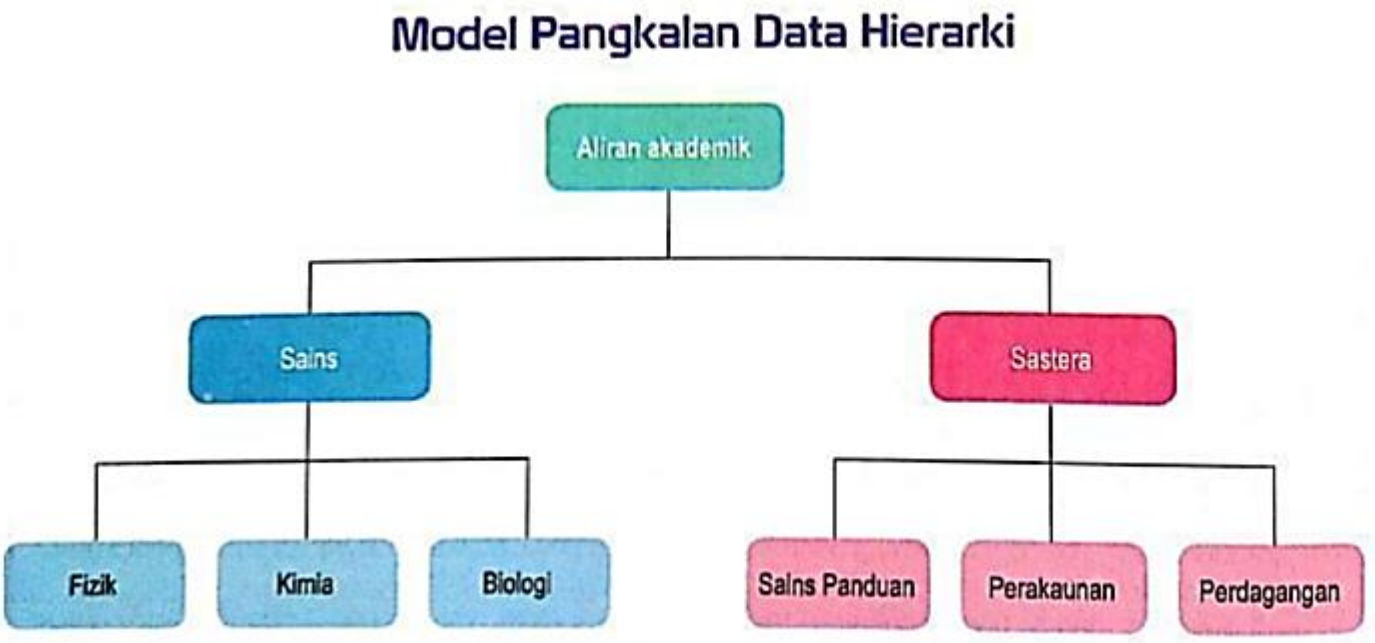
Object 1 Instance

01-12-01
24
I-95
2.5
6.0
6.0

Object 2: Maintenance Activity

Activity Code	
Activity Name	
Production Unit	
Average Daily Production Rate	

MODEL PANGKALAN DATA : HIERARKI



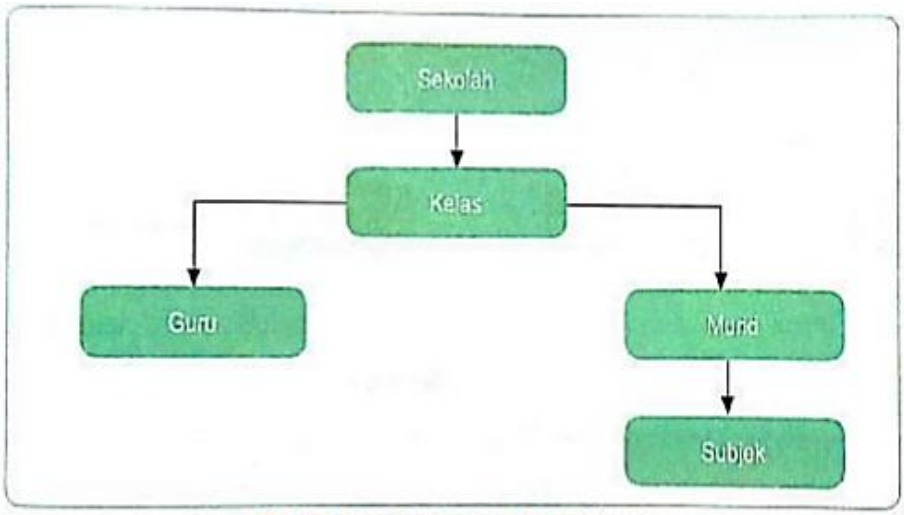
MODEL PANGKALAN DATA HIERARKI

- Model terawal semenjak tahun 1950.
- Sering digunakan dalam sistem maklumat keluaran syarikat IBM.
- Data disusun dalam struktur pokok.
- SESUAI digunakan sekiranya suatu entiti mempunyai hubungan satu induk (parent) dengan satu atau lebih entity anak (child).
- TIDAK menyokong hubungan banyak entiti induk kepada banyak entiti anak.
- Oleh itu, model ini tidak banyak digunakan.

MODEL PANGKALAN DATA : HIERARKI

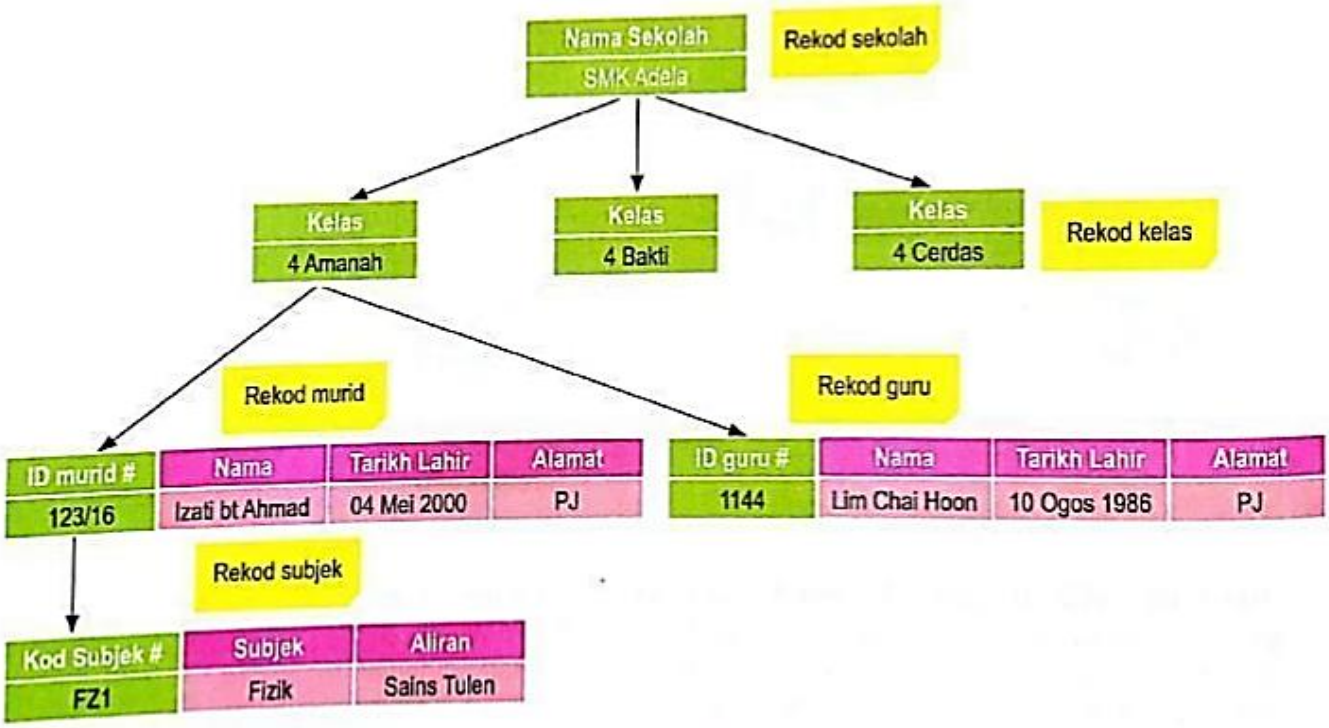
CONTOH

Katakan sebuah pengkalan data berasaskan model pangkalan data hierarki diperlukan untuk menyimpan rekod data entiti-entiti seperti Sekolah, Murid, Kelas, Guru dan Subjek.



Rajah 2.9 Model Pangkalan Data Hierarki

- Entiti Sekolah ialah **INDUK** kepada entiti kelas.
- Entiti Kelas ialah **INDUK** kepada entiti guru dan entiti murid.
- Entiti Murid ialah **INDUK** kepada entiti subjek.
- Entiti Subjek ialah **ANAK** kepada entiti Murid

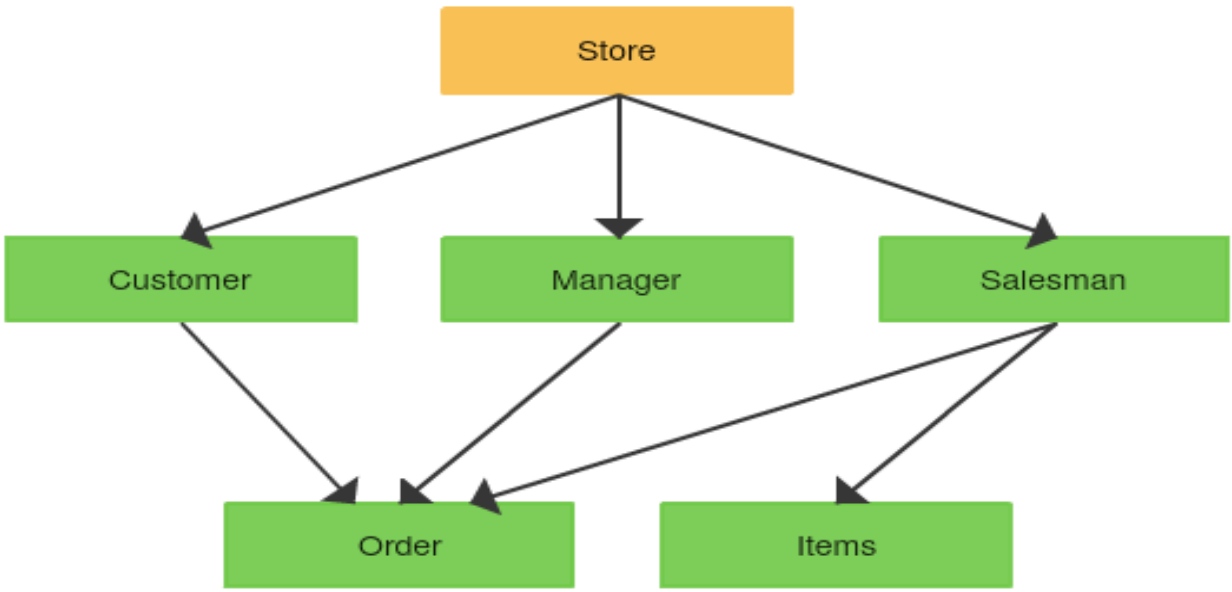


Rajah 2.12 Rekod data dalam model hierarki

MODEL PANGKALAN DATA : RANGKAIAN

MODEL PANGKALAN DATA RANGKAIAN

- Terdiri daripada beberapa jenis rekod dan dihubungkan melalui penunjuk.
- Model ini dapat mengatasi beberapa ketidakbolehtentuan dalam Model Hierarki.

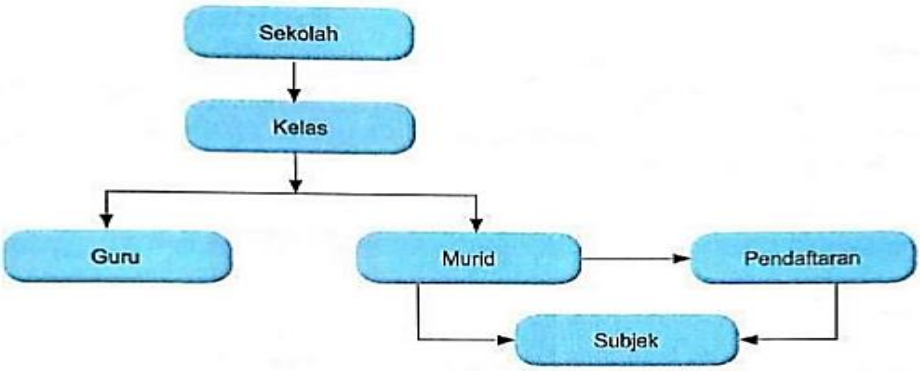


Database design done using network model. In the network model a node can have multiple parent nodes.

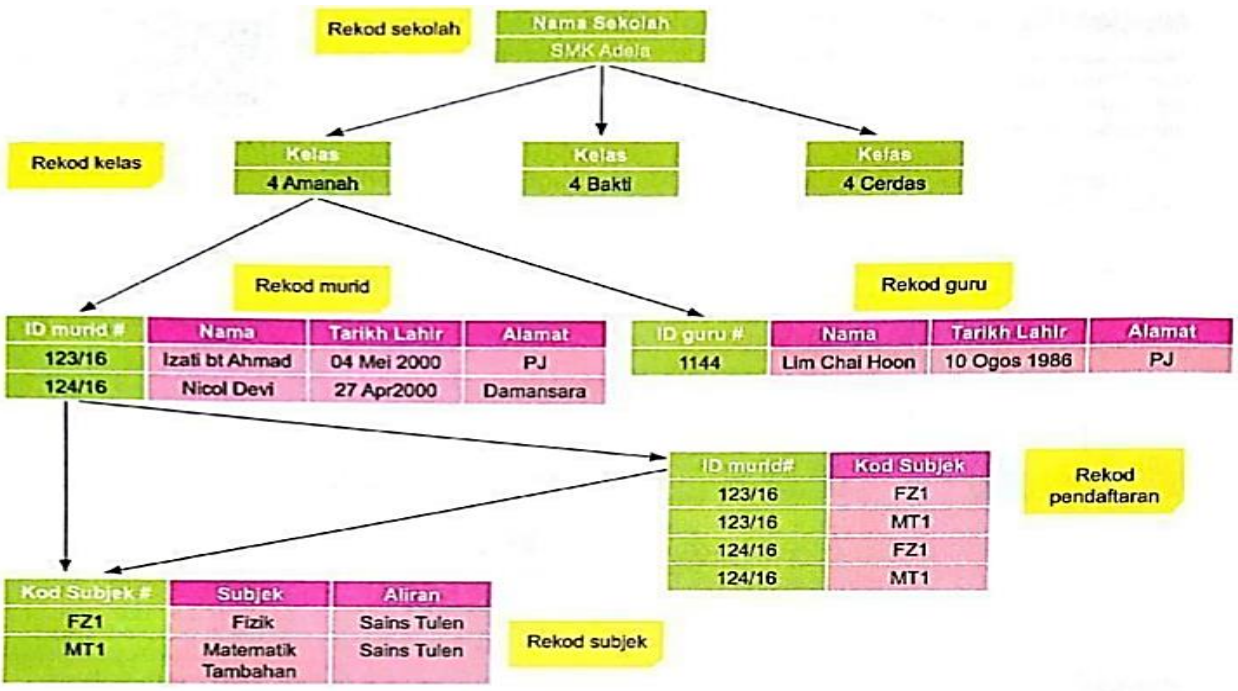
MODEL PANGKALAN DATA : RANGKAIAN

CONTOH

- Entiti Subjek boleh mempunyai 2 **INDUK** iaitu entity Kelas dan entity Murid.
- Pencarian semua Subjek dalam Kelas tertentu boleh dibuat secara terus berbanding dengan Model Hierarki.
- Hubungan Murid dan Subjek adalah hubungan **BANYAK-KE-BANYAK** iaitu setiap murid mungkin mendaftar untuk banyak Subjek dan setiap Subjek mempunyai banyak Murid.



Rajah 2.13 Model pangkalan data rangkaian

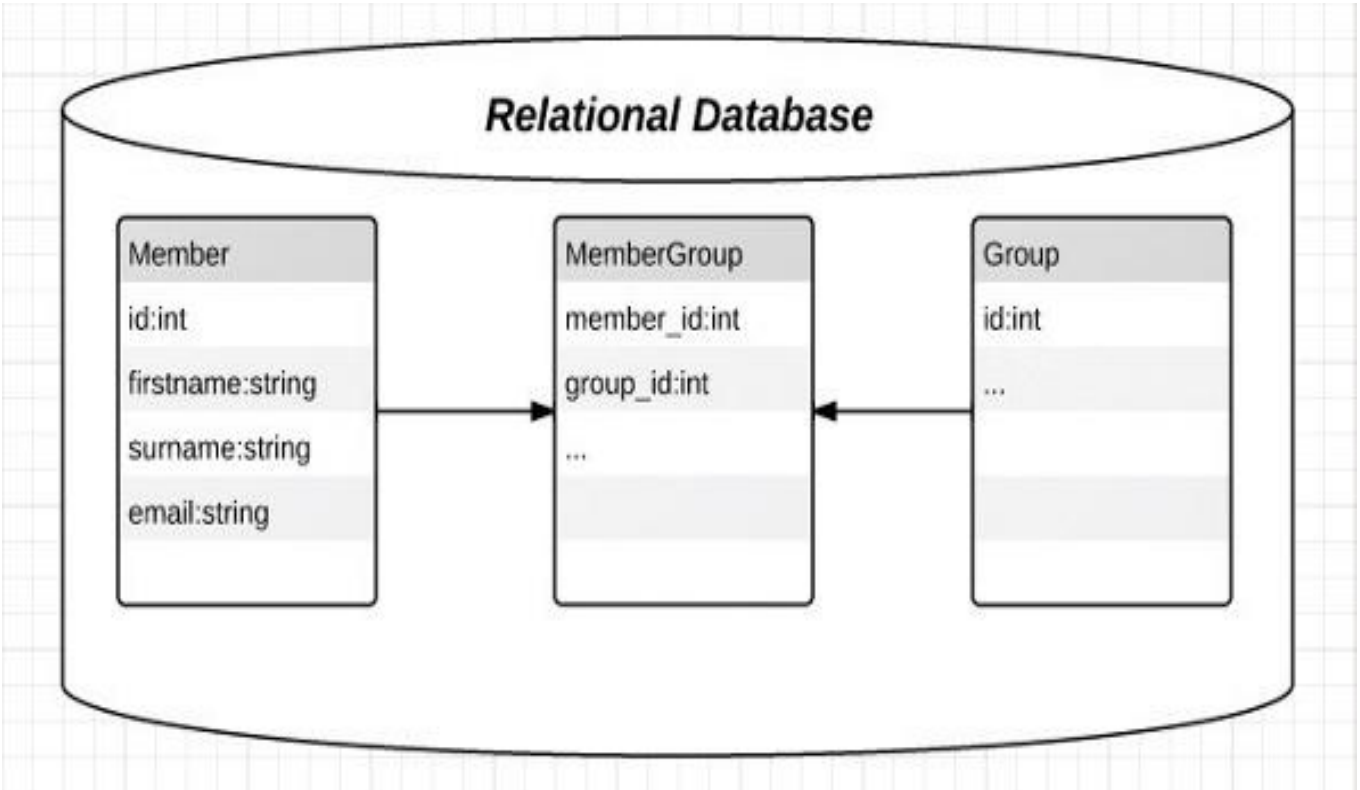


Rajah 2.14 Rekod data model pangkalan data rangkaian

MODEL PANGKALAN DATA : HUBUNGAN

MODEL PANGKALAN DATA HUBUNGAN

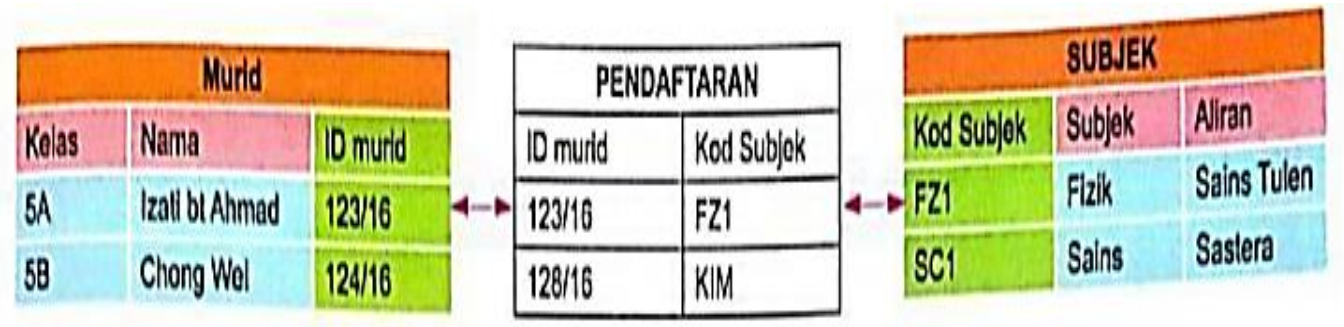
- SERING digunakan kerana mudah dibina, digunakan dan diuruskan dalam dunia sebenar.
- Datanya disusun dalam jadual yang terdiri daripada lajur dan baris.
- KELEBIHAN – Struktur pangkalan data tidak perlu dinyatakan terlebih dahulu.



MODEL PANGKALAN DATA : HUBUNGAN

CONTOH

- Model ini digambarkan dengan menggunakan 3 entiti (jadual) iaitu MURID, PENDAFTARAN dan SUBJEK.
- Atribut ID Murid dari entitI MURID dan Kod subjek dari entiti SUBJEK dijadikan kunci primer.
- Kunci Primer ini akan dihubungkan kepada kunci asing atribut ID Murid dan Kod Subjek di dalam entiti PENDAFTARAN.
- Selepas dihubungkan melalui kunci primer dan kunci asing ini, data berkenaan akan digabungkan.



Murid

Kelas	Nama	ID murid
5A	Izati bt Ahmad	123/16
5B	Chong Wei	124/16

PENDAFTARAN

ID murid	Kod Subjek
123/16	FZ1
128/16	KIM

SUBJEK

Kod Subjek	Subjek	Aliran
FZ1	Fizik	Sains Tulen
SC1	Sains	Sastera

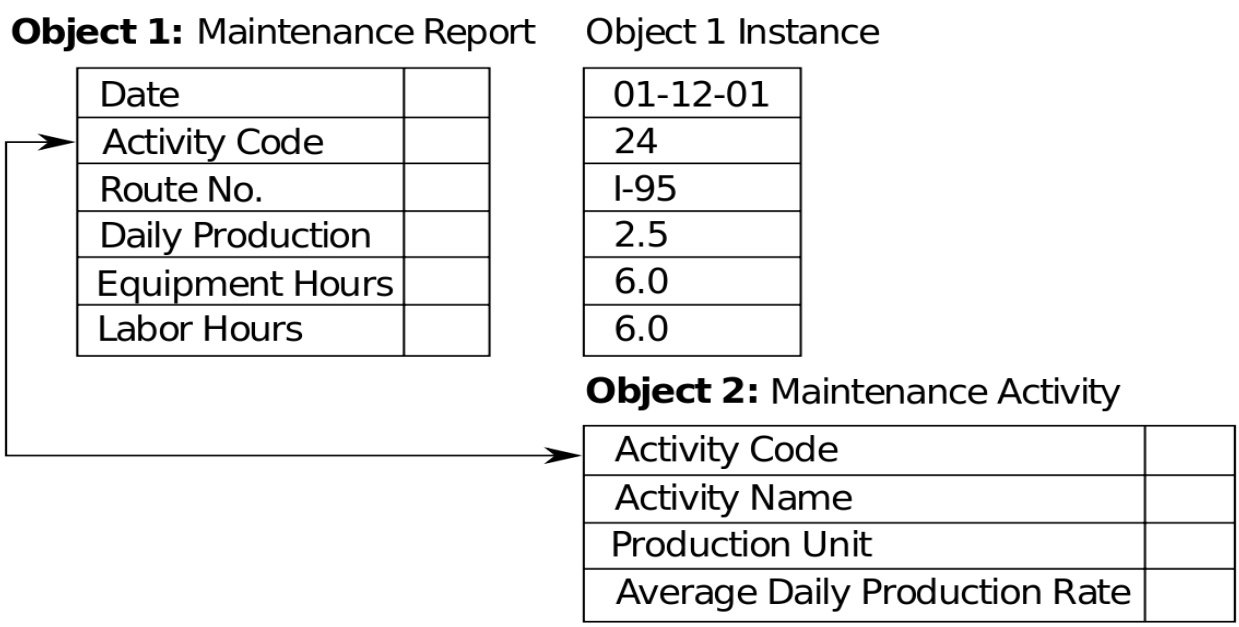
Rajah 2.15 Contoh model pangkalan data hubungan

MODEL PANGKALAN DATA : BERORIENTASIKAN OBJEK

MODEL PANGKALAN DATA BERORIENTASIKAN OBJEK

- Kaedah yang baharu dalam pengurusan data.
- Model ini menyimpan takrifan objek-objek yang boleh digunakan semula oleh perisian.
- Model ini mempunyai ciri yang sama dengan model rangkaian iaitu dapat mewakilkan data untuk hubungan BANYAK-KE-BANYAK

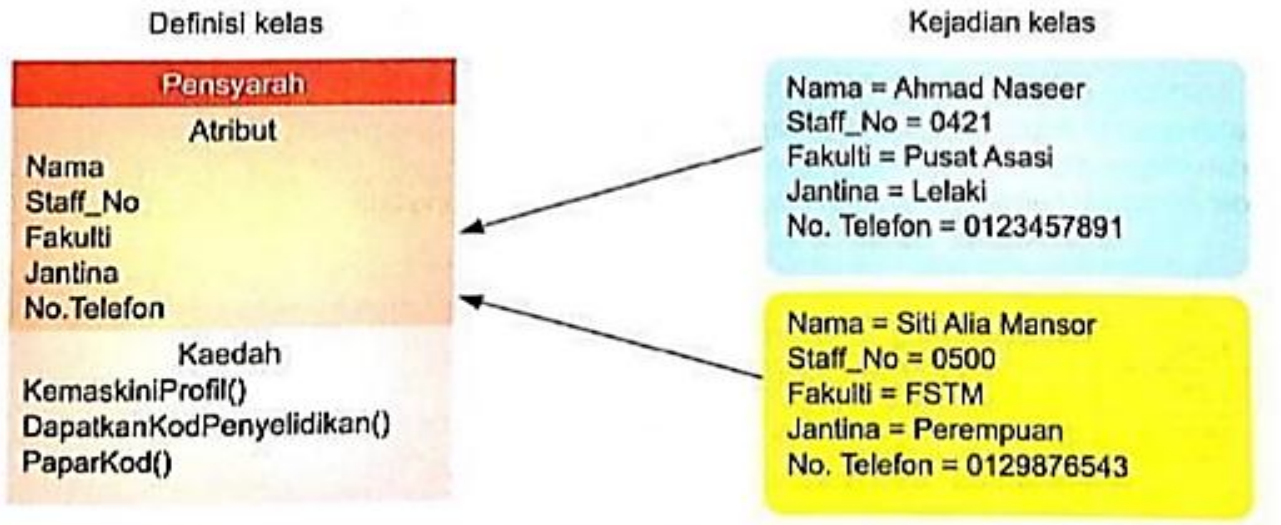
Object-Oriented Model



MODEL PANGKALAN DATA : BERORIENTASIKAN OBJEK

CONTOH

- Rajah 2.16 menggambarkan konsep orientasi objek bagi *kelas* Pensyarah yang mempunyai atribut dan kaedah (*method*) tersendiri.



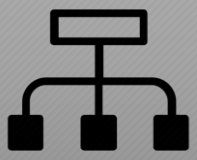
Rajah 2.16 Perwakilan kelas model pangkalan data berorientasikan objek

Kelas (*Class*)	Menggambarkan keadaan entiti objek sebenar.
Atribut (*Attributes*)	Data yang mewakili sifat-sifat objek tersebut.
Kaedah (*Method*)	Menjelaskan kelakuan bagi objek dan juga dikenal sebagai prosedur atau fungsi.

- Objek di dalam *kelas* dikenal sebagai *kejadian kelas* (class instances).
- Setiap *kejadian kelas* mempunyai nilai data tersendiri bagi setiap atribut tetapi masih boleh berkongsi nama atribut dan kaedah yang sama dengan *kejadian kelas* yang lain.

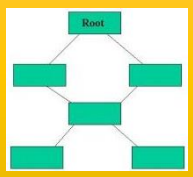
PERBANDINGAN ANTARA MODEL-MODEL PANGKALAN DATA

HIERARKI



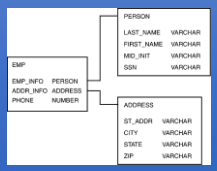
- Model terawal
- Konsep mudah menggunakan struktur pepohon untuk menyusun rekod.
- Tidak sesuai untuk hubungan banyak induk ke banyak anak.

RANGKAIAN



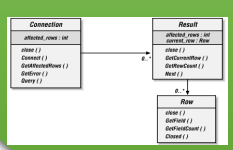
- Untuk mengatasi kelemahan model hierarki.
- Untuk hubungan banyak induk ke banyak anak.
- Sukar untuk menambah hubungan baharu.

HUBUNGAN



- Data disusun di dalam jadual terdiri daripada lajur dan baris.
- Mudah dibina, digunakan dan diuruskan.
- Model yang selalu digunakan.

BERORIENTASIKAN OBJEK

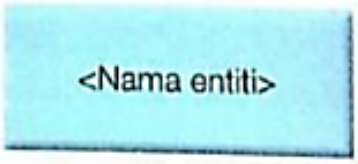


- Kaedah baharu dalam pengurusan data.
- Menyimpan takrifan kelas objek dan hubungan.
- Untuk pangkalan data yang memerlukan hubungan kompleks di antara objek-objek.
- Prestasi pencarian yang terbaik .

2.1 2.1.3 MENGENALPASTI ENTITI, ATRIBUT, SET HUBUNGAN DAN KEKARDINALAN BERDASARKAN KEPERLUAN PENGGUNA

ENTITI

ENTITI



Rajah 2.18 Simbol entiti

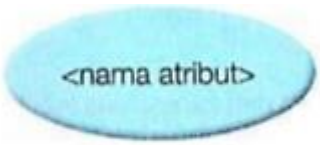
- Suatu objek unik dan boleh dikenal pasti dalam sesuatu persekitaran seperti orang, tempat atau benda.
- Mempunyai data deskriptif yang boleh dikutip dan disimpan.
- Bukan semua objek sesuai dijadikan entity.
- Jika tiada data dapat dikaitkan dengan objek, maka objek tersebut tidak sesuai dijadikan entiti.
- Setiap jadual mewakili satu entity manakala setiap rekod mewakili kejadian satu entiti.
- Nama entiti seharusnya kata nama umum yang bersesuaian.

CONTOH	PENYELESAIAN
Senaraikan entiti untuk pangkalan data dalam persekitaran sekolah.	• Guru • Murid • Mata pelajaran
Diberikan nama murid seperti Arumugam, Boh Leng, Che Siti dan Iking. Lukis simbol entiti yang bersesuaian	Murid

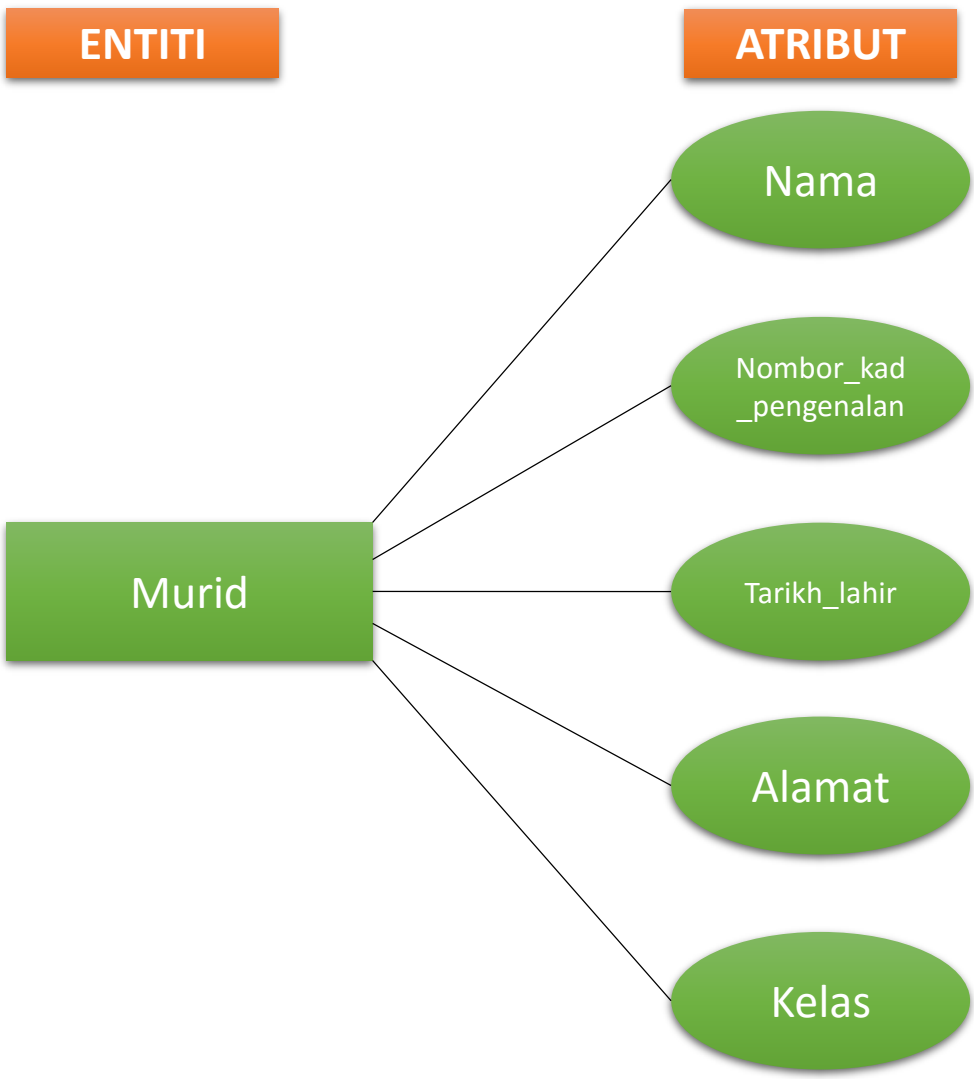
ATRIBUT

ATRIBUT

- Merupakan data deskriptif bagi sesuatu entiti.
- Atribut penting kerana menerangkan ciri-ciri sesuatu entiti.



Rajah 2.19 Simbol atribut



SET HUBUNGAN

SET HUBUNGAN

- Hubungan ialah perkaitan antara entiti.
- Merujuk perbuatan dan diwakili oleh **kata kerja.**
- Melibatkan dua entiti.
- **Entiti pertama (entiti subjek)** – pelaku yang melaksanakan hubungan tersebut ke atas entiti kedua.
- **Entiti kedua (entiti objek)** – menerima kesan daripada hubungan itu.
- **Set Hubungan –** koleksi sepasang entiti subjek-objek yang mempunyai hubungan yang sama.

Sintaks

<entiti subjek > <hubungan> <entiti objek>

SET HUBUNGAN

CONTOH

Diberikan hubungan-hubungan berikut.
- Adam belajar Sains Komputer.
- Ai Ling belajar Geografi.
- Anastasia belajar Sejarah

SOALAN	PENYELESAIAN
a) Kenal pasti entiti dalam hubungan-hubungan di atas.	• Adam, Ai Ling dan Anastasia ialah *kejadian* ataupun *objek* bagi **entiti Murid.** • Sains Komputer, Geografi dan Sejarah ialah *kejadian* ataupun *objek* bagi **entiti Mata pelajaran.**
b) Nyatakan nama hubungan yang sesuai.	• Belajar
c) Lukis gambar rajah hubungan yang bersesuaian.	

CONTOH

Diberikan hubungan-hubungan berikut.
- Adam deposit A0109
- Ali deposit A0150
- Barbara deposit A1011

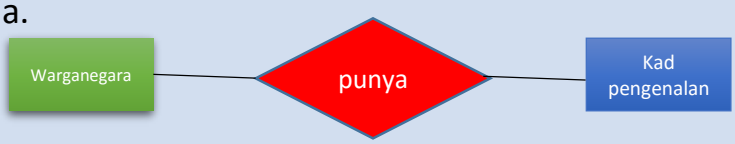
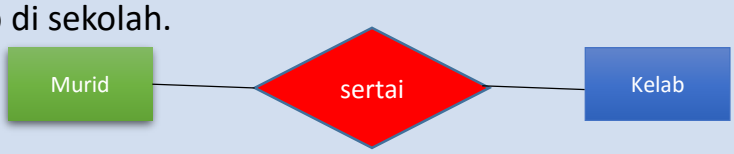
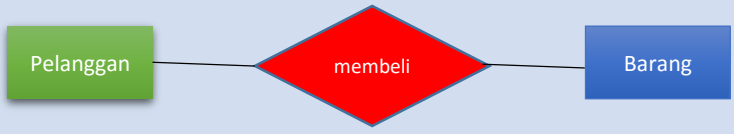
SOALAN	PENYELESAIAN
a) Kenal pasti entiti dalam hubungan-hubungan di atas.	• Adam, Ali dan Barbara ialah *kejadian* ataupun *objek* bagi **entiti Pelanggan Bank.** • A0109, A0150 dan A1011 ialah *kejadian* ataupun *objek* bagi **entiti Akaun.** • **Hubungan** – deposit.
c) Lukis gambar rajah hubungan yang bersesuaian.	

2.1 2.1.3 MENGENALPASTI ENTITI, ATRIBUT, SET HUBUNGAN DAN KEKARDINALAN BERDASARKAN KEPERLUAN PENGGUNA

KEKARDINALAN

KEKARDINALAN

- Merujuk kepada perhubungan antara entiti.
- Kekardinalan menyatakan bilangan entiti yang boleh dihubungkan dengan entiti yang lain melalui set hubungan.
- Dalam pangkalan data, kekardinalan merujuk hubungan di antara rekod-rekod dalam satu jadual kepada rekod-rekod dalam jadual yang lain.

KEKARDINALAN	GAMBAR RAJAH TERHUBUNG
1 : 1 (satu-ke-satu) Hubungan satu entiti dengan satu entiti yang lain.	CONTOH Seorang warganegara mempunyai satu kad pengenalan sahaja. 
1 : M (satu-ke-banyak) Hubungan satu entiti dengan lebih dari satu entiti yang lain.	CONTOH Seorang murid boleh menyertai lebih daripada satu kelab di sekolah. 
M : N (banyak-ke-banyak) Hubungan antara banyak entiti dengan banyak entiti yang lain.	CONTOH Ramai pelanggan bagi sebuah pasar raya membeli pelbagai jenis barangan. 

2.2 REKA BENTUK PANGKALAN DATA HUBUNGAN

Jadual asal

Kod Buku	ID Murid	Nama	No Telefon	Tarikh Pinjam	Tarikh Hantar
IPB124044	125007	ALI BIN AHMAD	053689090	22-Aug-16	12-Sep-16
IPB257868	125007	ALI BIN AHMAD	053689090	22-Aug-16	12-Sep-16
IPB192254	125007	ALI BIN AHMAD	053689090	29-Aug-16	12-Sep-16
IPB051375	125096	FRANCIS JR	036039999	23-Aug-16	13-Sep-16
IPB061045	125888	LIM S W	075555768	23-Aug-16	13-Sep-16

Jadual PEMINJAM

ID Murid	Nama	No Telefon
125007	ALI BIN AHMAD	053689090
125007	ALI BIN AHMAD	053689090
125007	ALI BIN AHMAD	053689090
125096	FRANCIS JR	036039999
125888	LIM S W	075555768

Pangkalan Data

Jadual PINJAMAN

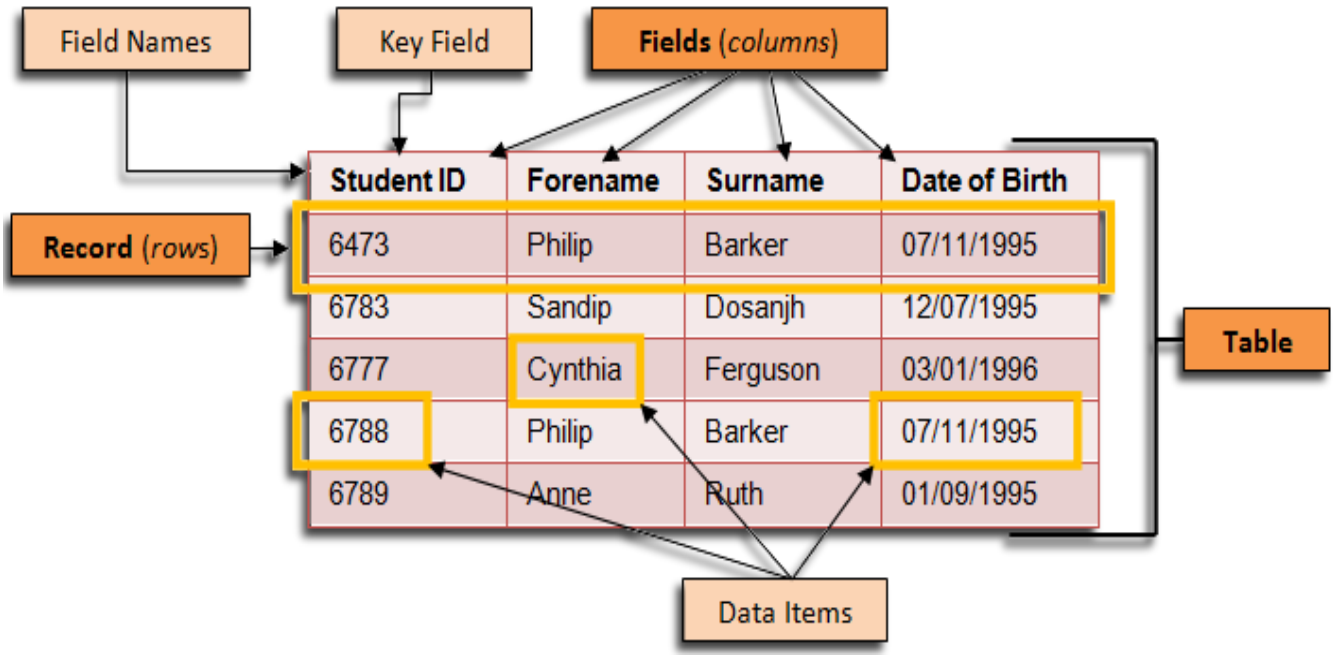
Kod Buku	ID Murid	Tarikh Hantar	Tarikh Pulang
IPB124044	125007	22-Aug-16	12-Sep-16
IPB257868	125007	22-Aug-16	12-Sep-16
IPB192254	125007	29-Aug-16	12-Sep-16
IPB051375	125096	23-Aug-16	13-Sep-16
IPB061045	125888	23-Aug-16	13-Sep-16

Rajah 2.24 Penormalan jadual untuk menghasilkan jadual-jadual hubungan

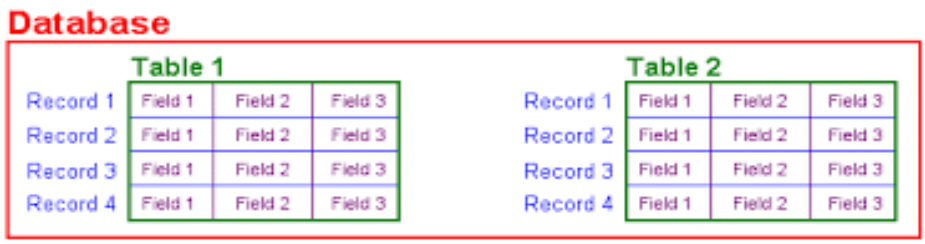
- Jadual asal ini lebar kerana mengandungi banyak lajur atribut.
- Jadual yang lebar cenderung mengandungi baris rekod yang mengulang nilai-nilai atribut yang sama.
- Jadual yang lebar boleh dipecahkan kepada beberapa jadual hubungan yang lebih kecil.
- Jadual-jadual inilah yang akan disimpan dalam pangkalan data.
- Kaedah ini dipanggil penormalan dan digunakan untuk mencegah kelewahan dan kehilangan ketekalan data.

2.2

2.2.1 MENGENALPASTI MEDAN (FIELD), REKOD, JADUAL (TABLE) DAN KEKUNCI BAGI PANGKALAN DATA YANG AKAN DIBANGUNKAN

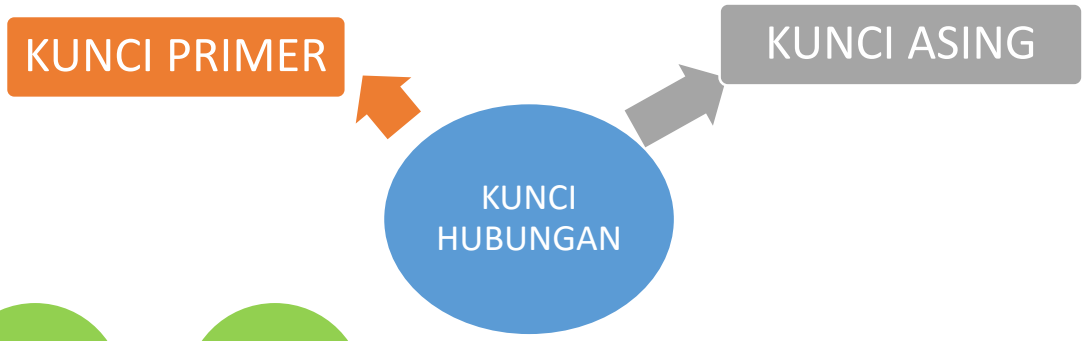


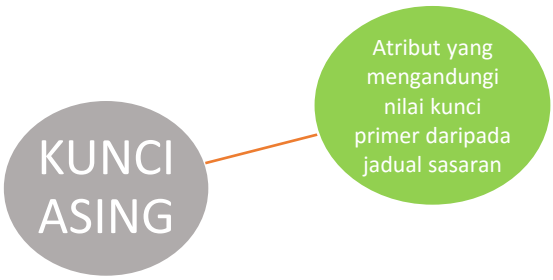
ITEM	KETERANGAN
MEDAN (FIELD)	Medan (*Field*) adalah sebahagian daripada rekod dan mengandungi data untuk subjek rekod.
REKOD	✓ Data disimpan dalam rekod. ✓ Rekod terdiri daripada medan dan mengandungi semua data tentang seseorang, syarikat, atau item tertentu dalam pangkalan data.
JADUAL (TABLE)	✓ Jadual pangkalan data terdiri daripada rekod dan medan yang memegang data. ✓ Setiap jadual dalam pangkalan data memegang data mengenai subjek yang berbeza tetapi berkaitan.
KEKUNCI	Untuk memudahkan pencarian maklumat dalam pangkalan data, maka dalam setiap rekod yang disimpan akan mempunyai satu medan yang dikenali sebagai medan kekunci.



2.2 2.2.2 MENENTUKAN KUNCI PRIMER DAN/ATAU KUNCI ASING YANG SESUAI BAGI SETIAO ENTITI

KUNCI HUBUNGAN – Nama atribut khusus dalam jadual yang digunakan untuk mengekalkan integriti data hubungan.





Jadual PINJAMAN

Kunci primer | Kunci asing

Kod Buku	ID Murid	Tarikh Hantar	Tarikh Pulang
IPB124044	125007	22-Aug-16	12-Sep-16
IPB257868	125007	22-Aug-16	12-Sep-16
IPB192254	125007	29-Aug-16	12-Sep-16
IPB051375	125096	23-Aug-16	13-Sep-16
IPB061045	125888	23-Aug-16	13-Sep-16

Jadual PEMINJAM

Kunci primer

ID Murid	Nama	No Telefon
125007	ALI BIN AHMAD	053689090
125096	FRANCIS JR	036039999
125888	LIM S W	075555768

Rajah 2.25 Hubungan antara kunci primer dan kunci asing yang menggunakan contoh jadual hubungan PINJAMAN dan PEMINJAM

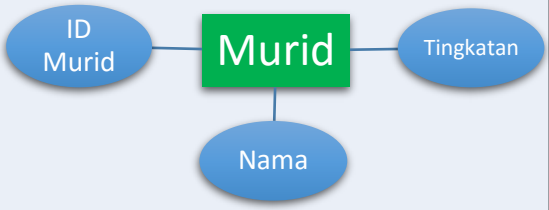
KEPENTINGAN KUNCI PRIMER

- Pastikan rekod tidak bertindih
- Beri identiti unik bagi setiap rekod
- Jadikan Data lebih utuh
- Jimat ruang stor komputer
- Mudahkan proses carian dan capaian rekod

2.2 2.2.3 MENGHASILKAN GAMBAR RAJAH PERHUBUNGAN ENTITI (ERD) UNTUK MEMODELKAN DATA HUBUNGAN

GAMBAR RAJAH PERHUBUNGAN (ERD)

- Merupakan teknik grafik untuk memodelkan data.
- Model yang dihasilkan adalah pada tahap konsep dan tidak terikat kepada mana-mana Sistem Pengurusan Pangkalan Data (*Database Management System* – DBMS).
- ERD mewakili persekitaran dalam struktur entiti, hubungan entiti dan atribut entiti.

<table>
<tr><th>KOMPONEN</th><th>SIMBOL</th><th colspan="2">CONTOH</th></tr>
<tr><td>ENTITI
• Sesuatu yang mempunyai data untuk disimpan.
• Biasanya dilabelkan dengan kata nama.
• Boleh terdiri daripada elemen-elemen persekitaran (orang,objek, tempat,konsep dan kejadian).</td><td>Nama Entiti</td><td colspan="2"><table><tr><th>JENIS ENTITI</th><th>CONTOH</th></tr><tr><td>Orang</td><td>Guru, Murid, Doktor</td></tr><tr><td>Tempat</td><td>Negara, negeri, daerah, bandar, desa</td></tr><tr><td>Objek</td><td>Produk, kenderaan, peralatan, bangunan</td></tr><tr><td>Peristiwa</td><td>Pendaftaran, permohonan, rayuan, pertanyaan, transaksi</td></tr><tr><td>Konsep</td><td>Akaun, kursus</td></tr></table></td></tr>
<tr><td>HUBUNGAN
• Perkaitan yang wujud antara 2 entiti.</td><td>Kata Kerja</td><td colspan="2">• CONTOH : mendaftar, mempunyai, mengisi, meminjam
 
</td></tr>
<tr><td>ATRIBUT
• Ciri atau sifat entiti.
• Setiap set entiti terdiri daripada beberapa atribut.</td><td>Nama</td><td colspan="2"> 
</td></tr>
</table>

2.2

2.2.3 MENGHASILKAN GAMBAR RAJAH PERHUBUNGAN ENTITI (ERD) UNTUK MEMODELKAN DATA HUBUNGAN

Langkah-langkah Melukis ERD

LANGKAH-LANGKAH MELUKIS ERD		CONTOH
		" Murid mengambil peperiksaan "
LANGKAH 1	Kenal pasti data yang diperlukan oleh sistem dari persekitaran pangkalan data.	
LANGKAH 2	Kenal pasti kumpulan untuk data yang berkait secara logikal. (Entiti)	• Murid • Peperiksaan
LANGKAH 3	Kenalpasti perkaitan antara dua entiti untuk semua entiti	• Mengambil
		

2.2

2.2.4 MEREKA BENTUK PANGKALAN DATA HUBUNGAN DENGAN MENUKAR ERD KE SKEMA HUBUNGAN

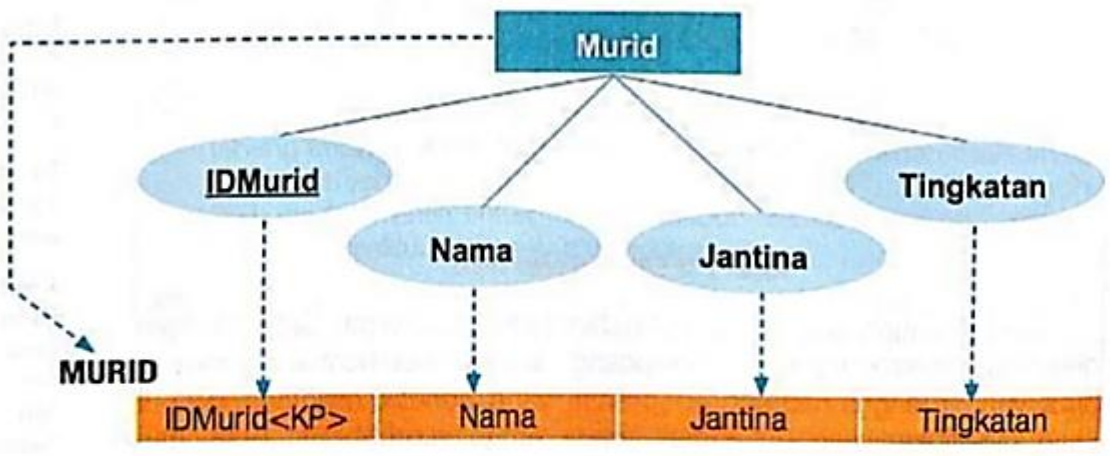
PERTIMBANGAN DALAM REKA BENTUK PANGKALAN DATA

- Mengandungi data yang lengkap dan diperlukan sahaja
- Hendaklah memudahkan penyimpanan, pencarian dan pengemaskinian
- Menggunakan model yang sesuai

Format Set Hubungan ERD

JENIS	KETERANGAN	
PERNYATAAN TEKS	FORMAT	NAMA ENTITI (Atribut 1 <KP>, Atribut 2, …Atribut N)
	CONTOH	Murid (IDMurid<KP>, Nama, Jantina, Tingkatan)
PERNYATAAN GRAFIK	FORMAT	NAMA ENTITI Atribut 1 <KP> | Atribut 2 | Atribut … | Atribut N
	CONTOH	MURID IDMurid<KP> | Nama | Jantina | Tingkatan

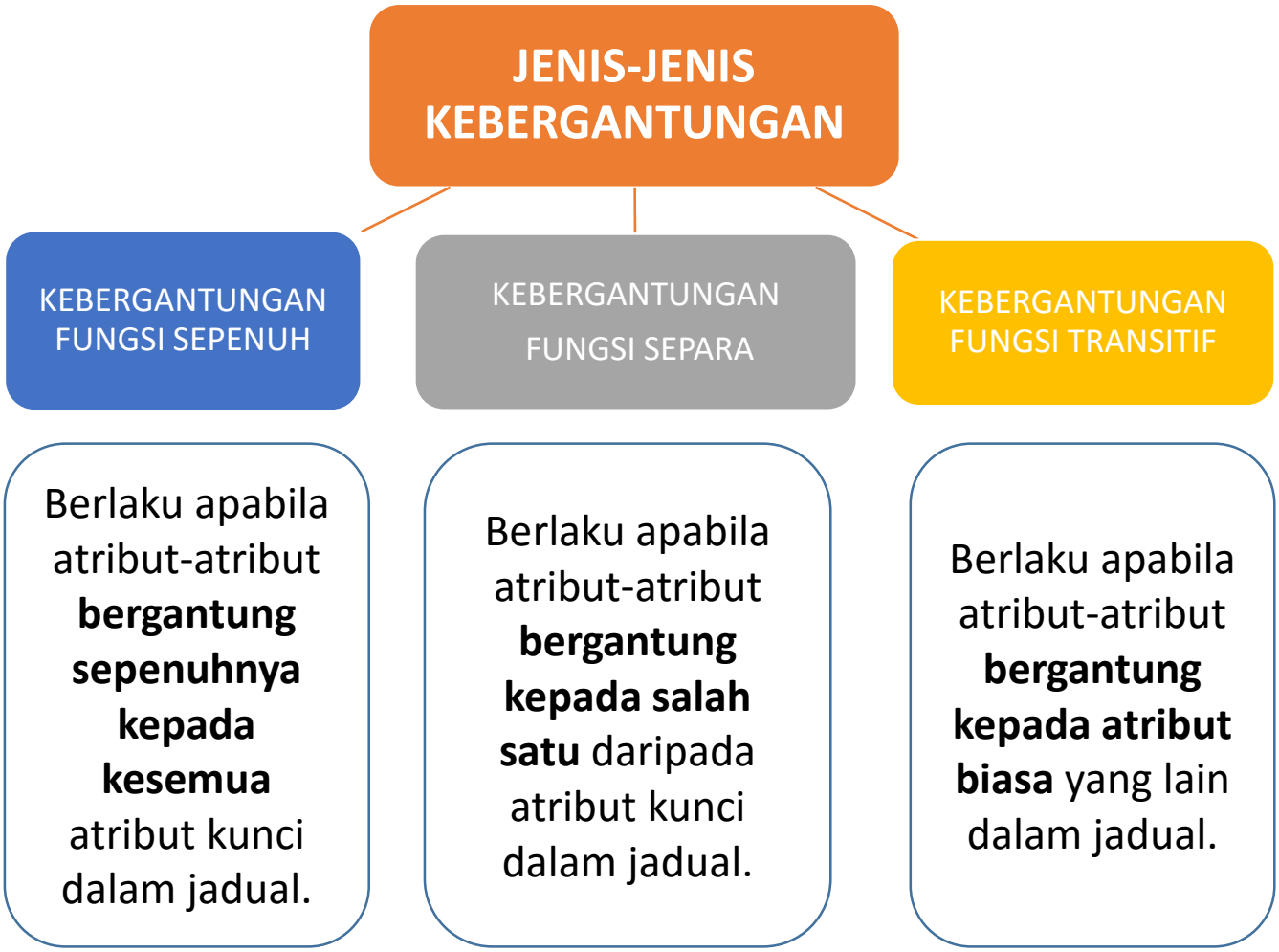
CATATAN :
- Kunci Primer ditandakan sebagai <KP>
- Kunci Asing ditandakan sebagai <KA>



Rajah 2.33 Menukar ERD entiti murid kepada set hubungan

2.2

2.2.5 MENJELASKAN KONSEP KEBERGANTUNGAN FUNGSI SEPENUH, SEPARA DAN TRANSITIF



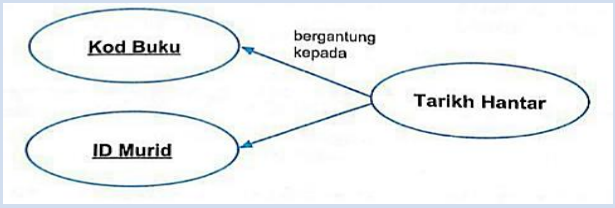
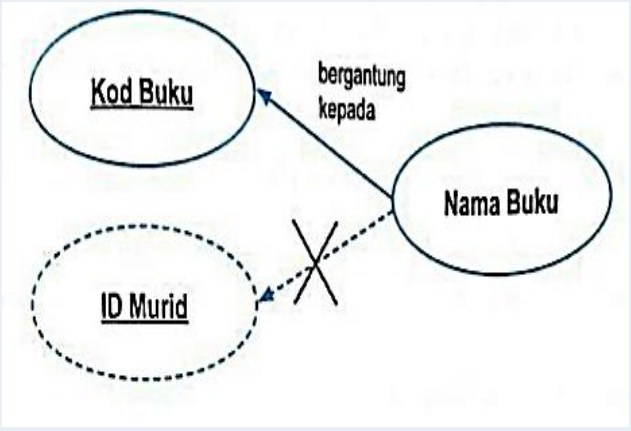
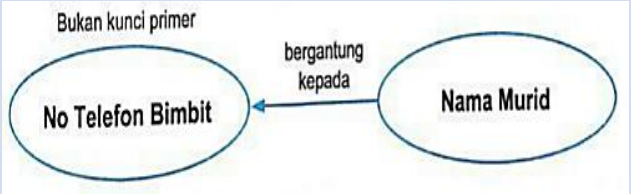
2.2 2.2.5 MENJELASKAN KONSEP KEBERGANTUNGAN FUNGSI SEPENUH, SEPARA DAN TRANSITIF

CONTOH

Kunci primer | Kunci primer

Kod Buku	Nama Buku	Pengarang	ID Murid	Nama Murid	No Telefon Bimbit	Tarikh Pinjam	Tarikh Hantar
IPB124044	Java Programming	Maruyama et al	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB257868	C Programming	M. A. Bakar	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB192254	Computer Graphics	Shirley Jr	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB051375	Software Engineering	Roger Estain	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016
IPB051325	Information Systems, an Introduction	Farah et al	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016

Rajah 2.34 Jadual BUKU PINJAMAN

JENIS KEBERGANTUNGAN	CONTOH	
KEBERGANTUNGAN FUNGSI SEPENUH		Sekiranya tiada salah satu kunci primer, Tarikh Hantar buku tidak dapat ditentukan.
KEBERGANTUNGAN FUNGSI SEPARA		Atribut Nama Buku bergantung kepada Kod Buku sahaja, bukan kedua-dua Kod Buku dan ID Murid. Jadi Nama Buku mempunyai kebergantungan fungsi seprara kepada Kod Buku
KEBERGANTUNGAN FUNGSI TRANSITIF		Tidak melibatkan kunci primer.

2.2 2.2.6 MELAKSANAKAN PROSES PENORMALAN DATA SEHINGGA 3NF TERHADAP SKEMA HUBUNGAN

- Jadual baharu biasanya dalam bentuk tidak ternormal atau **0NF**.
- Jadual 0NF boleh berfungsi, tetapi akan menyebabkan penyimpanan data lewah yang banyak dan boleh menjejaskan integriti data apabila kecuaian berlaku sewaktu kemas kini.
- Oleh itu, sesuatu jadual perlu disemak terlebih dahulu sebelum dilaksanakan dalam pangkalan data.

Jadual 0NF mempunyai banyak duplikasi nilai atribut

Buku			Murid			Tarikh Pinjam	Tarikh Hantar
Kod Buku	Nama Buku	Pengarang	ID Murid	Nama Murid	No Telefon Bimbit		
IPB124044	Java Programming	Maruyama et al	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB257868	C Programming	M. A. Bakar	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB192254	Computer Graphics	Shirley Jr	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB051375	Software Engineering	Roger Estain	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016
IPB051325	Information Systems, an Introduction	Farah et al	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016

Rajah 2.38 Jadual PINJAMAN BUKU dalam bentuk 0NF

PENORMALAN

- Merupakan satu kaedah menganalisis jadual-jadual berasaskan atrikut kunci dan kebergantungan fungsi dengan tujuan mengurangkan duplikasi data dalam pangkalan data.
- Penormalan biasanya memecahkan jadual 0NF kepada 2 atau lebih jadual-jadual hubungan yang sudah ternormal.
- Penormalan dilakukan secara sistematik dan berperingkat.
- 3 bentuk – 1NF, 2NF, 3NF
- Penormalan dibuat sehingga jadual mencapai peringkat 2NF atau 3NF

2.2 2.2.6 MELAKSANAKAN PROSES PENORMALAN DATA SEHINGGA 3NF TERHADAP SKEMA HUBUNGAN

CONTOH

Jadual asal (ONF): BUKU PINJAMAN

Buku			Murid			Tarikh Pinjam	Tarikh Hantar
Kod Buku	Nama Buku	Pengarang	ID Murid	Nama Murid	No Telefon Bimbit		
IPB124044	Java Programming	Maruyama et al	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB257868	C Programming	M. A. Bakar	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB192254	Computer Graphics	Shirley Jr	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB051375	Software Engineering	Roger Estain	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016
IPB051325	Information Systems, an Introduction	Farah et al	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016

Penormalan

Jadual PINJAMAN

Kod Buku	ID Murid	Tarikh Pinjam	Tarikh Hantar
IPB124044	125007	27-Sep-2016	10-Okt-2016
IPB257868	125007	27-Sep-2016	10-Okt-2016
IPB192254	125007	27-Sep-2016	10-Okt-2016
IPB051375	125096	28-Sep-2016	11-Okt-2016
IPB051325	125096	28-Sep-2016	11-Okt-2016

Jadual BUKU

Kod Buku	Nama Buku	Pengarang
IPB124044	Java Programming	Maruyama et al
IPB257868	C Programming	M. A. Bakar
IPB192254	Computer Graphics	Shirley Jr
IPB051375	Software Engineering	Roger Estain
IPB051325	Information Systems, an Introduction	Farah et al

Jadual MURID

ID Murid	Nama Murid
125007	Harris bin Aman
125096	Francis Embong

Jadual TELEFON

Nama Murid	No Telefon Bimbit
Harris bin Aman	0176677889
Francis Embong	0121122335

Rajah 2.39 Normalisasi jadual 0NF BUKU PINJAMAN kepada jadual-jadual hubungan 3NF PINJAMAN, BUKU, MURID dan TELEFON

2.2 2.2.6 MELAKSANAKAN PROSES PENORMALAN DATA SEHINGGA 3NF TERHADAP SKEMA HUBUNGAN

TUKARKAN SKEMA PERHUBUNGAN ONF KEPADA 1NF

- Objektif penukaran adalah untuk memastikan lajur jadual adalah atomik dan mempunyai kunci primer.
- Langkah pertama ialah memastikan keatomikan data-data dengan menggunakan satu lajur untuk setiap satu atribut.

Kunci primer

Kod Buku	Nama Buku	Pengarang	ID Murid	Nama Murid	No Telefon Bimbit	Tarikh Pinjam	Tarikh Hantar
IPB124044	Java Programming	Maruyama et al	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB257868	C Programming	M. A. Bakar	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB192254	Computer Graphics	Shirley Jr	125007	Harris bin Aman	0176677889	27-Sep-2016	10-Okt-2016
IPB051375	Software Engineering	Roger Esta'n	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016
IPB051325	Information Systems, an Introduction	Farah et al	125096	Francis Embong	0121122335	28-Sep-2016	11-Okt-2016

Rajah 2.40 Jadual PINJAMAN BUKU dalam bentuk 1NF

Berdasarkan jadual 1NF, hasilkan skema hubungan

PENYELESAIAN

BUKU PINJAMAN (Kod Buku <KP>, Nama Buku, Pengarang, ID Murid<KP>, Nama Murid, No Telefon Bimbit, Tarikh Pinjam, Tarikh Hantar)

2.2 2.2.6 MELAKSANAKAN PROSES PENORMALAN DATA SEHINGGA 3NF TERHADAP SKEMA HUBUNGAN

TUKARKAN SKEMA PERHUBUNGAN 1NF KEPADA 2NF

- Objektif penukaran 1NF kepada 2NF adalah untuk menghapuskan kebergantungan fungsi separa.
- Kenal pasti kumpulan-kumpulan data berulang dan pecahkan kepada jadual-jadual berasingan yang dipanggil jadual hubungan.
- Kaji skema 1NF dengan mencari kebergantungan antara atribut-atribut bukan kunci dengan atribut kunci primer.
- Kebergantungan fungsi separa berlaku apabila atribut biasa bergantung kepada salah satu atribut kunci primer sahaja.
- Kenal pasti kumpulan atribut tersebut dan asingkan sebagai skema hubungan yang baharu.

CONTOH

Jadual 1NF
BUKU PINJAMAN (Kod Buku <KP>, Nama Buku, Pengarang, ID Murid<KP>, Nama Murid, No Telefon Bimbit, Tarikh Pinjam, Tarikh Hantar)

Didapati ID Murid, Nama Murid dan No Telefon Bimbit mempunyai data berulang. Gunakan kurungan untuk menandakan kumpulan data berulang.
BUKU PINJAMAN (Kod Buku <KP>, Nama Buku, Pengarang, **(**ID Murid<KP>, Nama Murid, No Telefon Bimbit**)**, Tarikh Pinjam, Tarikh Hantar)

Asingkan atribut-atribut bagi kumpulan data berulang dan berikan nama entiti Murid.
BUKU PINJAMAN (Kod Buku <KP>, Nama Buku, Pengarang, **()**, Tarikh Pinjam, Tarikh Hantar) MURID (ID Murid<KP>, Nama Murid, No Telefon Bimbit)

Tukarkan nama jadual BUKU PINJAMAN kepada PINJAMAN. Atribut ID Murid dikekalkan, jadikan Kunci Asing
PINJAMAN (Kod Buku <KP>, Nama Buku, Pengarang, **ID Murid <KP><KA>**, Tarikh Pinjam, Tarikh Hantar) MURID (ID Murid<KP>, Nama Murid, No Telefon Bimbit)

2.2 2.2.6 MELAKSANAKAN PROSES PENORMALAN DATA SEHINGGA 3NF TERHADAP SKEMA HUBUNGAN

TUKARKAN SKEMA PERHUBUNGAN 1NF KEPADA 2NF

Kedua-dua jadual PINJAMAN dan MURID masih dalam 1NF selagi mengandungi kebergantungan fungsi separa.

PINJAMAN (Kod Buku <KP>, Nama Buku, Pengarang, **ID Murid <KP><KA>**, Tarikh Pinjam, Tarikh Hantar)

MURID (ID Murid<KP>, Nama Murid, No Telefon Bimbit)

- Dalam jadual PINJAMAN – Tarikh Pinjam dan Tarikh Hantar bergantung penuh kepada kedua-dua kunci primer Kod Buku dan ID Buku **(Kebergantungan Fungsi Sepenuh**).
- Atribut Nama Buku dan Pengarang bergantung kepada kunci primer Kod Buku sahaja.
- Oleh itu, jadual PINJAMAN mempunyai kebergantungan kunci separa di antara Nama Buku dan Pengarang dengan Kod Buku.
- Oleh itu, kumpulan atribut data dengan kebergantungan fungsi separa diasingkan sebagai skema hubungan baharu – entiti BUKU.

PINJAMAN (**Kod Buku <KP><KA>,** ID Murid <KP><KA>, Tarikh Pinjam, Tarikh Hantar)

BUKU (Kod Buku <KP>, Nama Buku, Pengarang)

Semak kedua-dua jadual untuk kewujudan lain-lain kebergantungan fungsi separa.
Jika tiada , maka jadual sudah menjadi jadual 2NF

HASIL PERNORMALAN 2NF

PINJAMAN (Kod Buku <KP><KA>, ID Murid <KP><KA>, Tarikh Pinjam, Tarikh Hantar)

BUKU (Kod Buku <KP>, Nama Buku, Pengarang

MURID (ID Murid<KP>, Nama Murid, No Telefon Bimbit)

2.2

2.2.6 MELAKSANAKAN PROSES PENORMALAN DATA SEHINGGA 3NF TERHADAP SKEMA HUBUNGAN

TUKARKAN SKEMA PERHUBUNGAN 2NF KEPADA 3NF

- Objektif penukaran adalah untuk menghapuskan kebergantungan fungsi transitif.
- Pada kebiasaannya, pernormalan sehingga tahap 2NF sudah memadai.
- Tahap 3NF cuma perlu dalam situasi di mana terdapat kebergantungan fungsi transitif di antara atribut dalam sesetengah jadual.
- Kebergantungan ini tersembunyi kerana wujud di antara atribut-atribut biasa, tidak melibatkan kunci primer.
- Biasanya dapat dikenal pasti daripada pengalaman penggunaan data-data.

CONTOH

Kaji skema hubungan jadual MURID untuk mencari kebergantungan transitif
MURID (ID Murid<KP>, Nama Murid, No Telefon Bimbit)

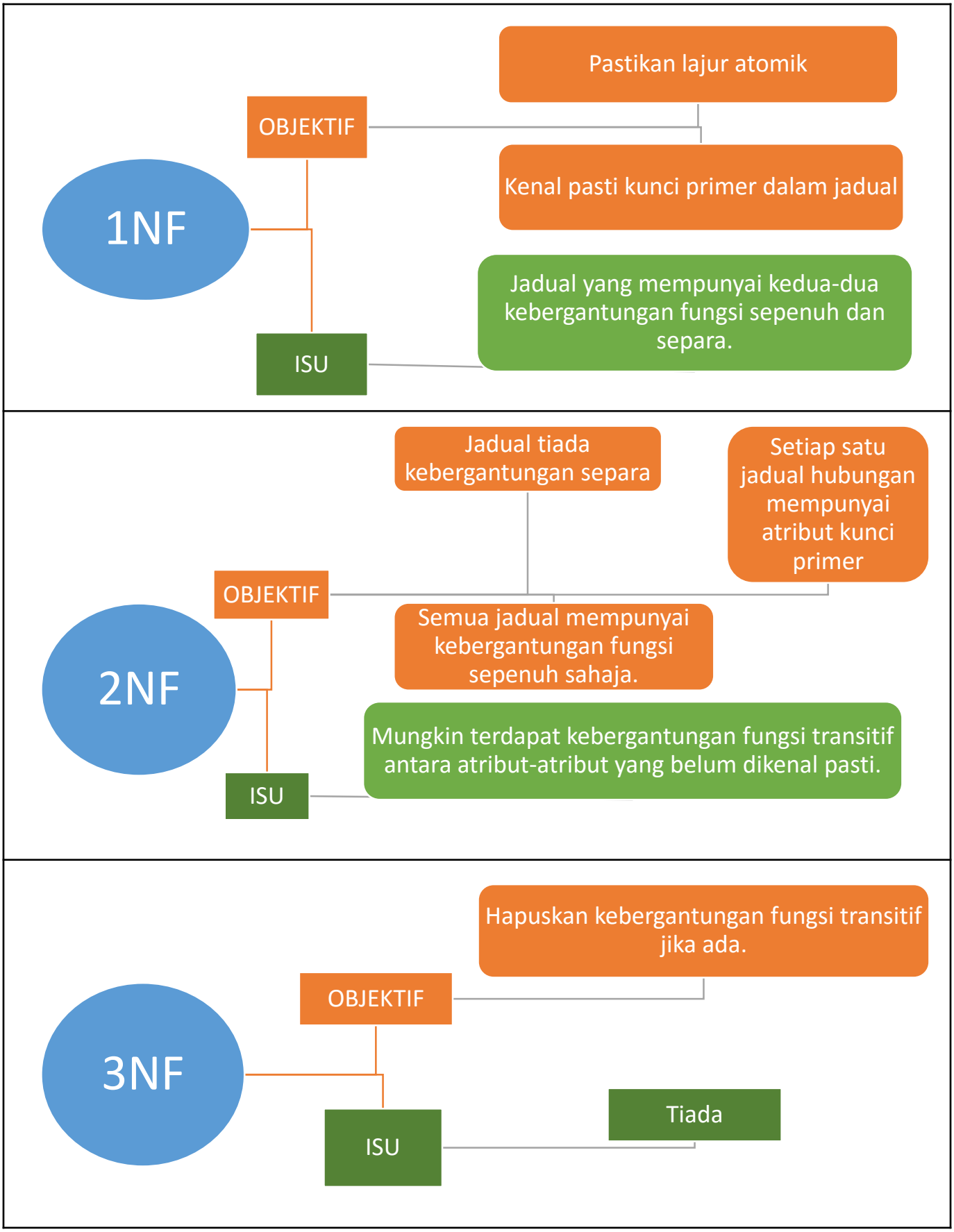
- Atribut Nama Murid bergantung kepada No Telefon Bimbit walaupun No Telefon Bimbit bukan kunci primer.
- Asingkan fungsi transitif tersebut dengan menghasilkan skema hubungan TELEFON.

MURID (ID Murid<KP>, Nama Murid, **No Telefon Bimbit <KA>**)

TELEFON (**No Telefon Bimbit <KP>** , Nama Murid)

Oleh sebab skema jadual PINJAMAN dan BUKU tidak menghasilkan kebergantungan baharu, maka jadual-jadual tersebut tidak mempunyai bentuk 3NF.

2.2 2.2.6 MELAKSANAKAN PROSES PENORMALAN DATA SEHINGGA 3NF TERHADAP SKEMA HUBUNGAN



2.3 PEMBANGUNAN PANGKALAN DATA HUBUNGAN

2.3.1	MEMBINA JADUAL BERPANDUKAN SKEMA HUBUNGAN MENGGUNAKAN PERISIAN PANGKALAN DATA HUBUNGAN
2.3.2	MENCIPTA BORANG YANG BERKAITAN DENGAN KANDUNGAN JADUAL
2.3.3	MEMASUKKAN DATA DALAM JADUAL MENGGUNAKAN BORANG
2.3.4	MENGHASILKAN *QUERY* UNTUK MENDAPATKAN SEMULA MAKLUMAT YANG DIPERLUKAN
2.3.5	MENJANA LAPORAN BERDASARKAN HASIL *QUERY*
2.3.6	MENGHASILKAN SATU SISTEM MAKLUMAT MUDAH MELALUI MAKRO MENGGUNAKAN MENU (SWITCHBOARD)
2.3.7	MENDOKUMENTASIKAN HASIL KERJA

~Rujuk Buku Teks~

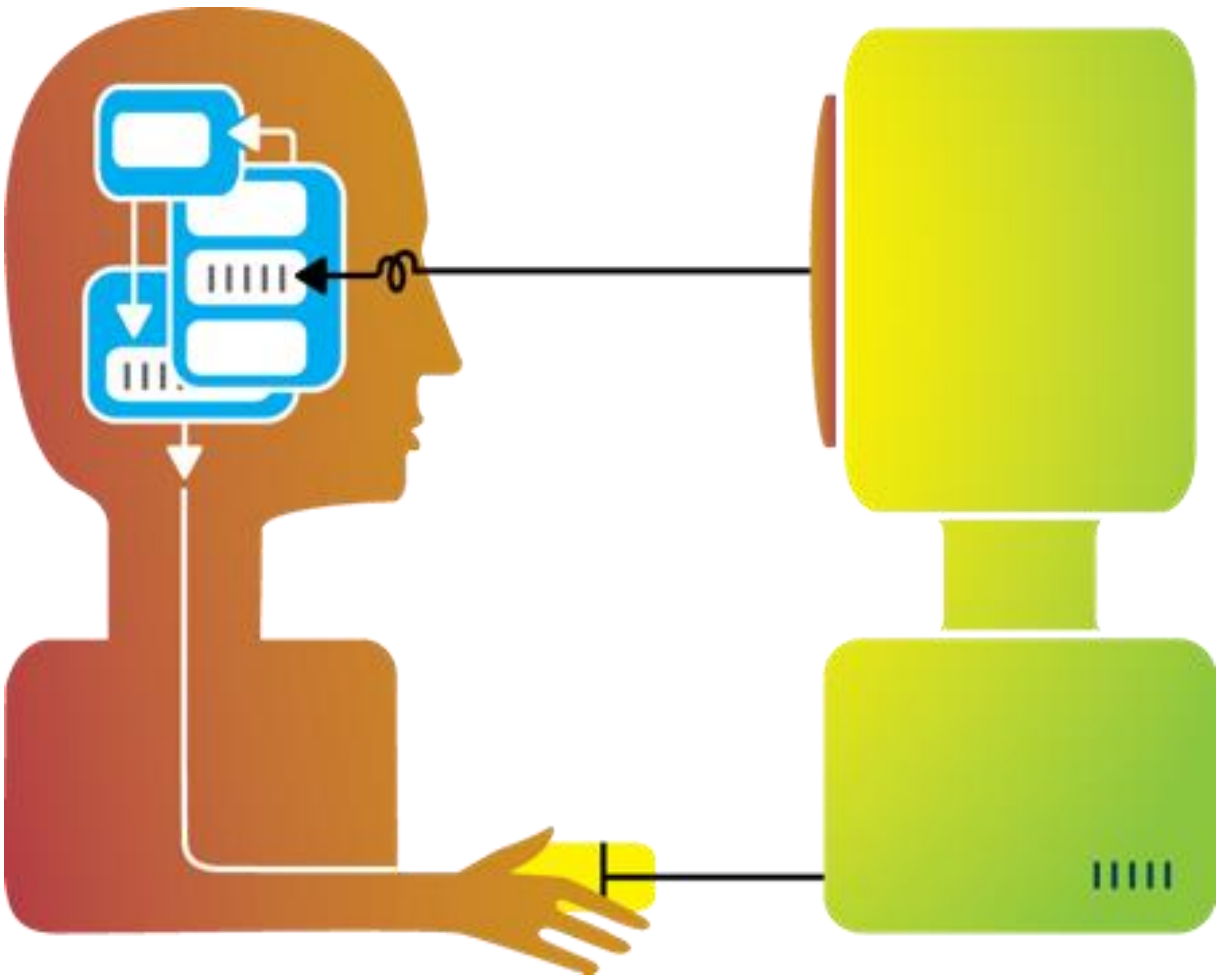
2.4 PEMBANGUNAN SISTEM PANGKALAN DATA

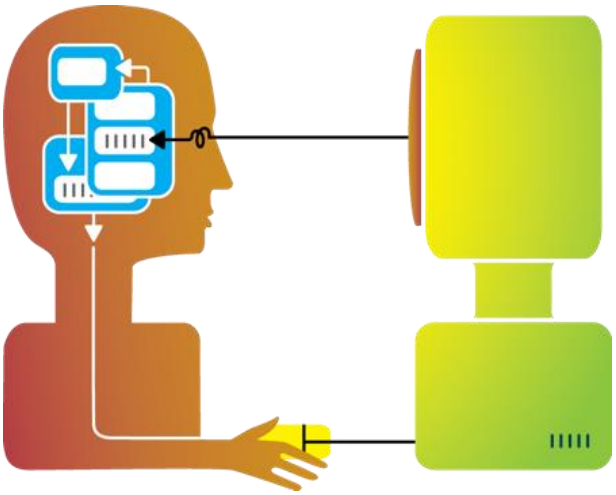
2.4.1	MENGHASILKAN SEBUAH PANGKALAN DATA YANG TERNORMAL
2.4.2	MEMBANGUNKAN SISTEM PANGKALAN DATA DENGAN ANTARA MUKA BERGRAFIK MENGGUNAKAN PERISIAN PEMBANGUNAN SISTEM PANGKALAN DATA MENGIKUT SDLC UNTUK MENYELESAIKAN MASALAH

~Rujuk Buku Teks~

3.0

INTERAKSI MANUSIA KOMPUTER





3.0 INTERAKSI MANUSIA KOMPUTER

- 3.1 Rekabentuk Interaksi
- 3.2 Paparan dan Reka Bentuk Skrin

3.1 REKA BENTUK INTERAKSI

REKA BENTUK INTERAKSI

- Merupakan satu aspek penting dan perlu dititikberatkan semasa membangunkan produk atau aplikasi.
- Reka bentuk yang menarik dan mudah digunakan akan membuatkan pengguna selesa dan seronok menggunakan produk tersebut.
- REKA BENTUK – susun atur teks, gambar, butang dan menu dalam sesuatu produk.
- INTERAKSI – tindakan manusia semasa menggunakan produk tersebut.
- REKA BENTUK INTERAKSI – membenarkan pengguna untuk berkomunikasi dan berinteraksi dengan produk atau aplikasi.

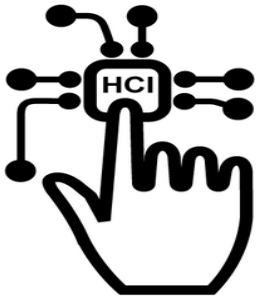
3.1

PRINSIP REKA BENTUK INTERAKSI

PRINSP	KETERANGAN
KONSISTENSI *(Consistency)*	• Semua elemen perlu kekal pada kedudukan yang sama supaya pengguna akan berasa selesa semasa menggunakan aplikasi. • Jika dialihkan – fokus pengguna terganggu. • Reka bentuk perlu konsisten dari segi persembahan dan fungsi pada semua antara muka.
KEBOLEHAN MEMBUAT PEMERHATIAN *(Perceivability)*	• Penunjuk atau butang interaksi perlulah mudah dikenal pasti. • Pengalaman pengguna terbaik adalah apabila pengguna boleh menggunakan aplikasi dengan selesa dan tanpa ragu-ragu.
BOLEH DIPELAJARI *(Learnability)*	• Reka bentuk interaksi sepatutnya mudah dipelajari dan diingati.
KEBOLEHAN UNTUK MENJANGKA *(Predictability)*	• Reka bentuk interaksi yang baik dan berkesan dapat membuatkan pengguna membuat jangkaan perkara yang akan berlaku dalam aliran proses aplikasi tersebut sebelum pengguna menggunakan aplikasi tersebut.
MAKLUM BALAS *(Feedback)*	• Maklum balas boleh memberikan gambaran sebenar reka bentuk interaksi dan maklumat mengenai hasil reka bentuk interkasi tersebut. • Pereka bentuk menggunakan maklum balas untuk melakukan penambahbaikan pada reka bentuk interaksi yang telah dihasilkan. • Sediakan maklum balas sekiranya pengguna memerlukannya. • Kegagalan untuk memberikan maklum balas yang dikehendaki boleh menyebabkan proses pengulangan yang tidak sepatutnya terhadap sebarang tindakan, kesalahan dan ralat.

3.1 3.1.1 MENGKAJI KEPERLUAN INTERAKSI ANTARA MANUSIA DAN KOMPUTER

KENAPA INTERAKSI ANTARA MANUSIA DAN KOMPUTER DIPERLUKAN



- MENDAPAT PERMINTAAN DALAM PASARAN
- MENINGKATKAN PRODUKTIVITI
- MENGURANGKAN KOS SELEPAS JUALAN
- MENGURANGKAN KOS PEMBANGUNAN
- MENGEMBANGKAN AKTIVITI DAN MENAMBAHKAN PENGALAMAN MANUSIA
- PENGKOMPUTERAN SOSIAL

3.1

3.1.2 MENILAI PRODUK INTERAKTIF BERDASARKAN TUJUAN DAN PRINSIP ASAS REKA BENTUK INTERAKTIF

ANTARA KAEDAH MENILAI PRODUK

- Temu bual
- Pemerhatian
- Soal selidik
- Perbincangan ahli kumpulan
- Refleksi

Jadual 3.1 Contoh Borang Soal Selidik untuk Menilai Kebolehgunaan Produk

Nama Aplikasi / Laman sesawang/ Sistem / Produk yang dinilai *Facebook*
Kategori Aplikasi / Laman sesawang / Sistem / Produk *Social Networking*
Platform iOS / *Windows* / PC / *Android* ...
Versi Aplikasi / Laman sesawang / Sistem / Produk (*jika ada)
Fungsi Utama Aplikasi / Laman sesawang / Sistem / Produk
Tarikh

Prinsip Asas	Penilaian	Tandakan (√) jika Ya dan (X) jika Tidak	
		Ya (√)	Tidak (X)
Konsisten	Adakah butang navigasi sentiasa berada di sebelah kiri?		
	Adakah laman web ini kerap menambah butang navigasi yang baharu?		
	Jika tetapan bahasa ditukar,adakah semua butang navigasi bertukar mengikut bahasa yang dipilih?		
Kebolehan membuat pemerhatian	Butang-butang navigasi diletakkan dalam satu kumpulan pada paparan aplikasi.		
	Nama dan ikon yang digunakan pada butang amat mudah untuk difahami.		
	Saya masih boleh menggunakan *Facebook* walaupun menu diletakkan di sebelah atas.		
	Saya tidak dapat memastikan akaun *Facebook* itu adalah pemilik akaun sebenar atau tidak terutamanya yang melibatkan akaun selebriti.		
Boleh dipelajari	Saya menggunakan kesemua butang navigasi yang disediakan setiap kali saya menggunakan aplikasi ini.		
	Saya mudah memahami aplikasi ini dan tidak perlu diajar banyak kali untuk menggunakannya.		
	Tetapan aplikasi ini membenarkan saya mengubahnya mengikut kehendak saya.		
	Saya memahami semua fungsi butang dan menu yang ditunjukkan dalam aplikasi ini.		
	Semakin hari saya semakin cekap menggunakan aplikasi ini.		
Kebolehan untuk menjangka	Apabila menekan apa-apa butang, saya tahu apa yang akan dipaparkan selepas itu.		
	Aplikasi ini membawa saya ke halaman yang tepat mengikut kehendak saya.		
	Sesetengah butang navigasi membuat saya tertanya-tanya, apakah yang akan dipaparkan jika saya menekan butang itu.		
	Kadang-kadang saya berasa marah dengan aplikasi ini kerana iklan yang tidak diperlukan dipaparkan juga.		
Maklum balas	Aplikasi ini membuat carian apabila saya memasukkan nama pada ruang ***search***.		
	Aplikasi ini menyenaraikan tempat yang ada berhampiran dengan saya sewaktu menekan butang ***nearby places***.		
	Saya tidak boleh melihat gambar rakan yang belum menerima permohonan saya untuk menjadi rakannya.		
	Sesetengah navigasi mengambil masa yang lama untuk menghasilkan paparan.		

Soalan tamat.
Sekian, terima kasih

3.2 PAPARAN DAN REKA BENTUK SKRIN

PAPARAN DAN REKA BENTUK SKRIN

- Memainkan peranan yang penting dalam membangunkan sesebuah program atau perisian.
- Reka bentuk yang mudah, lengkap dan mesra pengguna perlu dititikberatkan.

Proses Reka Bentuk Interaksi

Mengenal pasti Keperluan Interaksi → Membangunkan Reka Bentuk Alternatif → Membina Prototaip Interaksi → Membuat Penilaian Reka Bentuk Interaksi Produk

3.2 3.2.1 MENGAPLIKASI PROSES REKA BENTUK INTERAKSI DALAM ATUR CARA YANG DIBANGUNKAN

MENGENAL PASTI KEPERLUAN INTERAKSI

- Keperluan produk dan mengetahui sebab sesuatu produk itu dibina perlu di kenal pasti.
- Fikirkan mengenai produk alternatif yang akan dibina dan apa yang anda inginkan produk itu lakukan untuk anda.
- Kumpulkan maklum balas daripada pengguna sasaran mengenai fungsi produk yang mereka inginkan.

MEMBANGUNKAN REKA BENTUK ALTERNATIF

- **REKA BENTUK ALTERNATIF** – lakaran beberapa reka bentuk yang akan dicadangkan kepada pereka bentuk yang dihasilkan melalui gambaran idea pereka bentuk itu sendiri, produk yang sedia ada, hasil tinjauan maklum balas dan sebagainya.
- Wujudkan sekurang-kurangnya 2 reka bentuk alternatif.

MEMBINA PROTOTAIP INTERAKSI

- Setelah reka bentuk altenatif dengan lakaran papan cerita siap, kedua-duanya akan diedarkan kepada pengguna untuk dinilai.
- Pengguna akan memberi komen terhadap reka bentuk alternatif tersebut.
- Hasil dapatan akan digunakan untuk menghasilkan reka bentuk prototaip yang lebih baik.

MEMBUAT PENILAIAN KE ATAS REKA BENTUK

- Setelah menghasilkan prototaip reka bentuk, sekali lagi prototaip yang telah ditambah baik akan dinilai oleh pengguna.
- Penilaian ini merupakan penilaian akhir di mana produk telah diimplementasikan dengan menggunakan perisian komputer.
- Intrumen penilaian perlu dibina setelah paparan dan reka bentuk dihasilkan.
- Inrstrumen penilaian mestilah menepati kriteria yang diperlukan.

3.2 3.2.1 MENGAPLIKASI PROSES REKA BENTUK INTERAKSI DALAM ATUR CARA YANG DIBANGUNKAN

CONTOH

Tajuk Projek	**Program Mengira Dua Nombor**
Objektif Projek	Membolehkan murid darjah tiga mengira dua nombor dengan pantas secara automatik
Sasaran	Murid darjah tiga
Kategori pengguna	Guru/Murid

Tandakan (✓) sekiranya jawapan anda Ya dan tandakan (✗) sekiranya jawapan anda Tidak.

Kategori	Kriteria	Ya (✓)	Tidak (✗)
Reka Bentuk Skrin	1. Reka bentuk skrin mudah dan ringkas.		
	2. Pemilihan warna dan ikon yang bersesuaian dan menarik.		
	3. Sistem navigasi yang disediakan mudah dikenal pasti dan mesra pengguna.		
	4. Saiz paparan adalah sesuai.		
	5. Tidak mengandungi kesalahan ejaan.		
	6. Tidak menyebabkan murid sesat dalam penerokaan.		
	7. Laras bahasa yang digunakan mudah difahami.		
	8. Tiada gangguan teknikal semasa menggunakan program ini.		
Interaktiviti pengguna	1. Murid boleh mengawal sepenuhnya butang kawalan operasi matematik.		
	2. Butang kawalan operasi matematik adalah mesra pengguna.		
	3. Butang kawalan operasi menepati setiap pengiraan matematik.		
	4. Butang "Keluar" berfungsi dengan baik.		
	5. Butang "Reset" berfungsi dengan baik.		

Rajah 3.8 Contoh Instrumen Penilaian Kuantitatif Prototaip "Program Mengira Dua Nombor"

3.2 3.2.2 MENGHASILKAN PROTOTAIP PAPARAN DAN REKA BENTUK SKRIN

~Rujuk Buku Teks~

3.2 3.2.3 MEMBINCANGKAN LAPORAN DARIPADA PENILAIAN KUANTITATIF TERHADAP PAPARAN DAN REKA BENTUK SKRIN

PENILAIAN KUANTITATIF

- Penilaian yang dilakukan untuk mengukur keberkesanan sesuatu produk secara statistik.
- Boleh diukur dan dinyatakan dalam bentuk nombor atau peratusan mengikui kesesuaian gaya persembahan.
- Boleh dianalisis daripada data yang diperoleh daripada jumlah pengguna dan tahap kepuasan mereka seperti yang ditentukan oleh kaji selidik atau temu bual.

CONTOH

Jadual 3.5 Penilaian Kuantitatif terhadap paparan dan reka bentuk skrin yang diperolehi daripada maklum balas 25 orang pengguna

No	Item	Jumlah		PERATUS (%)	
		Ya (✓)	Tidak (✗)	Ya (✓)	Tidak (✗)
Reka Bentuk Skrin	1. Reka bentuk skrin mudah dan ringkas.	23	2	92	8
	2. Pemilihan warna dan ikon adalah bersesuaian dan menarik.	15	10	60	40
	3. Sistem navigasi yang disediakan mudah dikenal pasti dan mesra pengguna.	23	2	92	8
	4. Saiz paparan adalah sesuai.	24	1	96	4
	5. Tidak mengandungi kesalahan ejaan.	25	0	100	0
	6. Tidak menyebabkan murid sesat dalam penerokaan.	25	0	100	0
	7. Laras bahasa yang digunakan mudah difahami.	25	0	100	0
	8. Tiada gangguan teknikal semasa saya menggunakan program ini.	24	1	96	4
Interaktiviti pengguna	1. Murid boleh mengawal sepenuhnya butang kawalan operasi matematik.	25	0	100	0
	2. Butang kawalan operasi matematik mesra pengguna.	25	0	100	0
	3. Butang kawalan operasi menepati setiap pengiraan matematik.	25	0	100	0
	4. Butang "Keluar" berfungsi dengan baik.	25	0	100	0
	5. Butang "Reset" berfungsi dengan baik.	25	0	100	0
	Purata			95.1	4.9

Rajah 3.16 Carta pai bagi maklum balas pengguna terhadap paparan dan reka bentuk skrin Program Mengira Dua Nombor