DAYANG NORHAYATI ABANG JAWAWI
ROSBI MAMAT

UTM
UNIVERSITI TEKNOLOGI MALAYSIA

Bab 3: Operator dan Ungkapan Aritmetik

© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.1 Pengenalan

- Operator
  - Data yang disimpan di dalam ingatan boleh diolah dengan menggunakan **operator** (cth : operator umpukan "=" dan operator darab "*").
  - **Operator** digunakan untuk menggabungkan kendalian atau operan menjadi ungkapan.

- Ungkapan Aritmetik
  - merupakan satu kombinasi pemalar, pemboleh ubah dan operator yang digunakan untuk membuat pengiraan. Contoh ungkapan dengan tiga pemboleh ubah adalah seperti berikut:

ungkapan	`gaji + elaun - hutang`

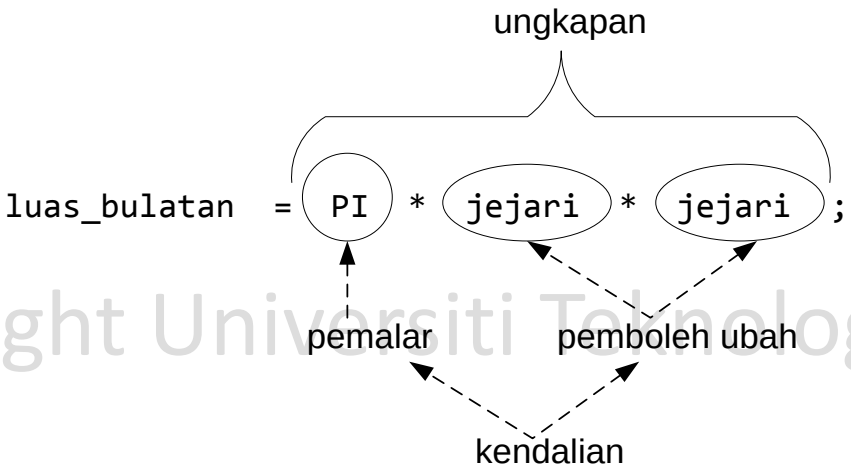
3.1 Pengenalan

- Ungkapan yang ditamatkan dengan ” ; ” menjadi kenyataan.

Kenyataan	`pendapatan = gaji + elaun - hutang;`

- C mempunyai 45 operator untuk mengolah pelbagai jenis data di dalam bahasa pengaturcaraan C. Operator ini terbahagi kepada tiga kategori iaitu **operator aritmetik**, **operator hubungan** dan **operator logik**.

- Contoh satu kenyataan dan ungkapan aritmetik:-



© Copyright Universiti Teknologi Malaysia

3.2 Operator Aritmetik

- Operator aritmetik dalam C terbahagi kepada 2:-

  - **Operator aritmetik sesatu** : digunakan dengan satu kendalian sahaja.

  - **Operator perduaan** : digunakan dengan dua kendalian dan satu operator yang terletak di antara dua kendalian.

© Copyright Universiti Teknologi Malaysia

3.2.1 Operator Aritmetik Sesatu

- Terdapat sembilan operator utama dalam kategori sesatu. Dibawah menunjukkan operator aritmetik sesatu dan fungsinya.

Operator	Fungsi operator
+	Positif
-	Negative
++	Tokok
--	Susut
~	Salingan atau pelengkap satu
sizeof	Saiz sesuatu pemboleh ubah dalam bait
!	Tak
&	Alamat bagi sesuatu pemboleh ubah
*	Kandungan sesuatu alamat

© Copyright Universiti Teknologi Malaysia

3.2.1 Operator Aritmetik Sesatu

- Contoh berikut menunjukkan dua pemboleh ubah diisytiharkan iaitu integer dan uinteger dengan saiz dua bait.

```
int integer; unsigned int uinteger;

uinteger = 64;

integer = +uinteger;          /* integer = 64 uinteger = 64 */

integer = -uinteger;    /* integer = -64 uinteger = 64 */

integer = ++uinteger;          /* integer = 65 uinteger = 65 */

integer = --uinteger;    /* integer = 64 uinteger = 64 */
```

© Copyright Universiti Teknologi Malaysia

3.2.1 Operator Aritmetik Sesatu

- Contoh operator tokok dan susut di atas dilakukan sebelum kendalian, ini dinamakan **operator awalan** yang mana nilai ditokok atau disusut dahulu sebelum umpukan ke pemboleh ubah integer dibuat.

- Jika operator tokok dan susut dikenakan selepas satu pemboleh ubah, **operator akhiran**, nilai asal akan diumpukkan ke pemboleh ubah dan selepas itu barulah nilai pemboleh ubah tersebut ditokok atau disusut.

© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.2.1 Operator Aritmetik Sesatu

- Contoh atur cara Membezakan operasi operator tokok akhiran dan awalan

```
1:  #include <stdio.h>
2:
3:  int main () {
4:                  int bil, bilCampur, bilTolak;
5:                  bil = 24;
6:                  bilCampur = bil++;          bilTolak = ++bil;
7:                  printf( "bil = %d, bilCampur = %d, iTolak = %d", bil,
8:                          bilCampur, bilTolak);
9:                  return 0;
10: }
```

- Paparan perbezaan nilai bil, bilCampur dan bilTolak

```
bil = 26, bilCampur = 24, bilTolak = 26
```

3.2.2 Operator Aritmetik Perduaan

- **Operator perduaan** beroperasi pada dua kendalian dan ia mengembalikan nilai hasil manipulasi.

- Operator aritmetik perduaan:-

Operator	Fungsi
+	Tambah
-	Tolak
*	Darab
/	Bahagi
%	Baki bahagi atau modulus

© Copyright Universiti Teknologi Malaysia

3.2.2 Operator Aritmetik Perduaan

❖ **Contoh** Perubahan nilai pemboleh ubah dengan umpukan ungkapan artimetik perduaan:

```
int x, y, z;
x = 10; y = 17;        /* x diumpukkan dengan nilai 10 dan
                          y diumpukkan dengan nilai 17 */
z = x + y;             /* z diumpukkan dengan nilai 27 */
y = y - x;             /* y diumpukkan dengan nilai 7 */
x = y * z;             /* x diumpukkan dengan nilai 189 */
z = x / 20;            /* z diumpukkan dengan nilai 9 */
y = z % x;             /* y diumpukkan dengan nilai 9 */
```

© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.2.2 Operator Aritmetik Perduaan

- **Contoh** Perubahan nilai pemboleh ubah dengan umpukan ungkapan artimetik perduaan(SAMBUNGAN):

Umpukan 1

```
x =10; y = 17;
```

x	y	z
10	17	?

Umpukan 2

```
z = x + y;
```

x	y	z
10	17	27

Umpukan 3

```
y = y - x;
```

x	y	z
10	7	27

Umpukan 4

```
x = y * z;
```

x	y	z
189	7	27

Umpukan 5

```
z = x / 20;
```

x	y	z
189	7	9

Umpukan 6

```
y = z % x;
```

x	y	z
189	9	9

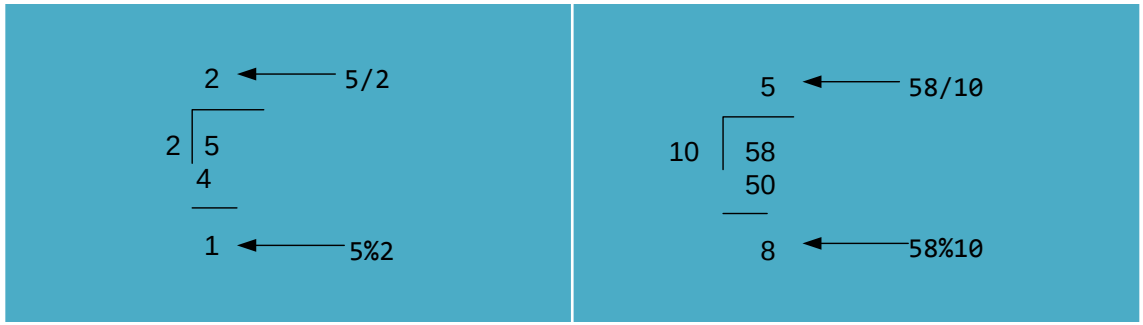
© Copyright Universiti Teknologi Malaysia

3.2.2 Operator Aritmetik Perduaan

- Operator perduaan modulus ini memberi nilai baki bagi pembahagian dua nombor dan ianya hanya boleh digunakan bersama kendalian berjenis integer sahaja.

- Kedua-dua operator perduaan bahagi dan modulus melibakan operasi pembahagian dua nombor.

- Hubungan operator perduaan bahagi dan modulus:



© Copyright Universiti Teknologi Malaysia

3.3 Ungkapan dengan Pelbagai Operator

- Setakat ini, ungkapan aritmetik mudah digunakan dengan satu operator pada satu kenyataan, seperti berikut:

```
x= -y;  x = y + z;
```

- Ungkapan aritmetik perduaan yang kompleks mempunyai bilangan operator melebihi satu dan bilangan kendalian yang melebihi dua. Format aritmetik perduaan yang kompleks adalah seperti berikut:

$kendalian_1$ $operator_1$ $kendalian_2$ $operator_2$ $kendalian_3$....$operator_{n-1}$ $kendalian_n$

- Contoh pelbagai operator dalam satu ungkapan adalah seperti berikut:

```
x= -y + z * 7 % 2 /++z;
```

© Copyright Universiti Teknologi Malaysia

3.3 Ungkapan dengan Pelbagai Operator

- Peraturan untuk menilai ungkapan ini adalah berdasarkan kepada peraturan algebra seperti berikut:

  - **Peraturan kurungan**. Kesemua sub-ungkapan di dalam kurungan perlu dinilai dahulu, bermula dengan sub-ungkapan dalam ke luar untuk kurungan bersarang.
  - **Peraturan keutamaan operator**. Peraturan untuk menilai operator ungkapan pada satu ungkapan yang sama. Keutamaan operator sesatu dan perduaan ditunjukkan di jadual:-

Keutamaan	Operator	Jenis	Kesekutuan
1 (tinggi)	- + -- ++ ! sizeof	sesatu	kanan ke kiri
2	* / %	perduaan	kiri ke kanan
3(rendah)	+ -	perduaan	kiri ke kanan

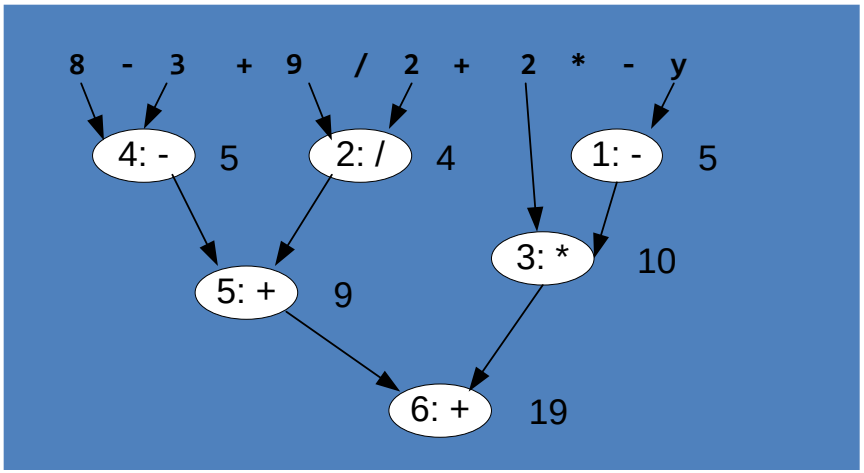
  - **Peraturan kesekutuan**. Jika terdapat lebih dari satu operator pada satu ungkapan yang mempunyai aras keutamaan yang sama, maka peraturan ini digunakan untuk menilai operator pada ungkapan sama ada dari operator paling kiri ke kanan atau sebaliknya seperti di Jadual diatas.

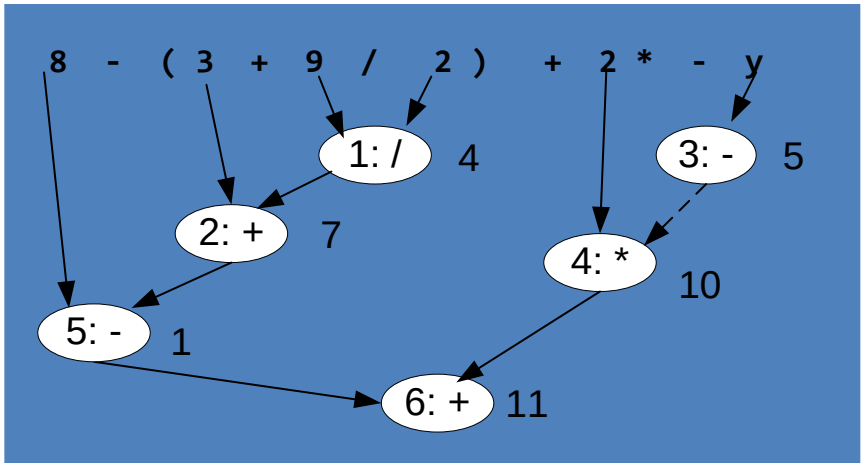
© Copyright Universiti Teknologi Malaysia

3.3 Ungkapan dengan Pelbagai Operator

- Contoh dua ungkapan yang melibatkan kendalian dan operator yang sama tetapi dinilai dengan keutamaan berbeza.

```
int z, y=-5;
z= 8 - 3 + 9 / 2 + 2 * - y;
z= 8 - (3 + 9 / 2) + 2 * - y;
```





© Copyright Universiti Teknologi Malaysia

3.3 Ungkapan dengan Pelbagai Operator

- Mengetahui peraturan keutamaan untuk menilai ungkapan aritmetik membantu pemahaman bagaimana C memproses kenyataan dengan ungkapan aritmetik.

- Tetapi untuk menghasilkan kod atur cara yang mudah dibaca penggunaan kurungan amat digalakkan.

  Contoh ungkapan **`x*y*z+y/b-c`**

  lebih mudah difahami jika ditulis **`x*((y*z)+y)/(b-c)`**

- Contoh formula matematik dengan ungkapan aritmetik

Formula	Ungkapan C
$b^2 - 4ac$	(b * b) – (4 * a * c)
$\frac{a+b}{c+d}$	(a + b) / (c + d)
$\frac{1}{y+\frac{y}{x}}$	1/ (y + (y / x))

3.4 Kenyataan Umpukan

- Kenyataan umpukan menggunakan simbol =. Pemboleh ubah yang perlu diumpuk nilai mestilah berada di sebelah kiri operator umpukan.

- Kenyataan umpukan seperti berikut akan memberi ralat sintaks kerana pemboleh ubahnya berada di sebelah kanan operator umpukan.

```
24 = i;
```

- Operator umpukan ini juga boleh digunakan dalam 2 cara:
  - umpukan berganda
  - umpukan operator majmuk

© Copyright Universiti Teknologi Malaysia

3.4.1 Umpukan Berganda

- Terdapat lebih dari satu operator umpukan dalam sesuatu kenyataan umpukan berganda.

- Contoh 1:

```
int a = b = c = d = e = 250;
```

- Contoh 2 di slide seterusnya

© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.4.1 Umpukan Berganda

- Contoh 2:

```
1:  #include <stdio.h>
2:
3:  int main () {
4:          int b = 2, nombor = 0, jum = 0, purata = 3;
5:          nombor = b = 10 + (jum=10-purata);
6:          printf("%d %d %d %d", b , nombor, jum, purata);
7:
8:          return 0;
9:  }
```

Output:

```
17 17 7 3
```

© Copyright Universiti Teknologi Malaysia

3.4.1 Umpukan Berganda

- Langkah menunjukkan perubahan nilai pemboleh ubah dengan umpukan ungkapan berganda:

Baris 4	Baris 5
Langkah 1: `b = 2` b: 2, nombor: ?, jum: ?, purata: ?	Langkah 1: `jum=10-purata` b: 2, nombor: 0, jum: 7, purata: 3
Langkah 2: `nombor = 0` b: 2, nombor: 0, jum: ?, purata: ?	Langkah 2: `b = 10 + (jum=10-purata)` b: 17, nombor: 0, jum: 7, purata: 3
Langkah 3: `jum = 0` b: 2, nombor: 0, jum: 0, purata: ?	Langkah 3: `nombor = b = 10 + (jum=10-purata)` b: 17, nombor: 17, jum: 7, purata: 3
Langkah 4: `purata = 3` b: 2, nombor: 0, jum: 0, purata: 3	

© Copyright Universiti Teknologi Malaysia

3.4.2 Umpukan Operator Majmuk

- **Operator umpukan majmuk** menggabungkan operator perduaan dengan operator umpukan mudah untuk memudahkan kenyataan sintaks perduaan berikut:

  `pemboleh_ubah = pemboleh_ubah operator ungkapan;`

  kepada:

  `pembolehubah operator= ungkapan;`

- Gabungan beberapa operator yang dibenarkan adalah seperti berikut:

  - Umpukan tambah dengan symbol +=
  - Umpukan tolak dengan symbol -=
  - Umpukan darab dengan symbol *=
  - Umpukan bahagi dengan symbol /=
  - Umpukan tambah dengan symbol %=

© Copyright Universiti Teknologi Malaysia

3.4.2 Umpukan Operator Majmuk

- Contoh penggunaan kenyaatan umpukan majmuk dan kenyataan dengan ungkapan operator perduaan yang sama:

Ungkapan operator majmuk	Ungkapan operator perduaan
`masa -= 1;`	`masa = masa -1;`
`bil += 1;`	`bil = bil + 1;`
`jum_ti += ti;`	`jum_ti = jum_ti + ti;`
`n *= x + 1;`	`n = n * (x + 1);`

- Penilaian ungkapan pelbagai operator yang melibatkan operator umpukan menggunakan peraturan keutamaan operator seperti di Jadual

Keutamaan	Operator	Jenis	Kesekutuan
1 (tinggi)	- + -- ++ ! sizeof	sesatu	kanan ke kiri
2	* / %	perduaan	kiri ke kanan
3	+ -	perduaan	kiri ke kanan
4 (rendah)	= += -= *= /= %=	umpukan	kanan ke kiri

© Copyright Universiti Teknologi Malaysia

3.5 Jenis Data dengan Ungkapan Aritmetik

- Semua pemboleh ubah atau pemalar yang digunakan pada atur cara C perlu diisytihar jenis datanya. Pemboleh ubah, pemalar atau malar digunakan pada ungkapan aritmetik akan dinilai dan diberi jenis data tertentu.

- Jika kesemua kendalian di dalam satu ungkapan aritmetik adalah berjenis integer, maka hasil bagi ungkapan aritmetik ini adalah juga berjenis integer. Tetapi, jika salah satu daripada kendalian adalah nombor perpuluhan, maka hasil ungkapan aritmetik adalah berjenis titik apung.

- Contoh:

```
5.0/2.0  hasil 2.5
5.0/2    hasil 2.5
5/2.0    hasil 2.5
5/2      hasil  2
```

© Copyright Universiti Teknologi Malaysia

3.5 Jenis Data dengan Ungkapan Aritmetik

- Jenis data pada pemboleh ubah paling kiri operator umpukan dan jenis hasil penilaian ungkapan akan menentukan nilai yang akan diumpukkan.

- Contoh:

```
int i1=3, i2=2, hasil1, hasil2;
double d=2.0, hasil3, hasil4;
hasil1 = i1/i2;      /* 3/2   */
hasil2 = i1/d;       /* 3/2.0 */
hasil3 = i1/i2;      /* 3/2   */
hasil4 = i1/d;       /* 3/2.0 */
```

- Gambaran kandungan pemboleh ubah hasil1, hasil2, hasil3 dan hasil4:

hasil1	hasil2	hasil3	hasil4
1	1	1.0	1.5

© Copyright Universiti Teknologi Malaysia

3.5.1 Operator Kas

- Terdapat situasi dalam satu ungkapan aritmetik yang melibat operasi beberapa pemboleh ubah jenis data yang berbeza, di mana pengatur cara perlu mengarahkan jenis data sesuatu pemboleh ubah ditukar.

- Pertukaran jenis data boleh diarahkan menggunakan **operator kas**. Format sintaks operator kas ialah seperti berikut:

```
(jenis_data) kendalian_atau_ungkapan
```

- Contoh kenyataan yang akan mengahasilkan ralat sintaks:

```
double a=3.0, b=2.0, hasil;
hasil = a % b; /*Ralat Sintaks!!!*/
```

- Contoh bebas ralat

```
hasil = (int)a % (int)b; /*Bebas Ralat*/
```

© Copyright Universiti Teknologi Malaysia

3.5.1 Operator Kas

- Dalam operasi umpukan bahagi, salah satu kendalian perlulah berjenis titik apung untuk menghasilkan nilai dengan titik perpuluhan.

- Contoh berikut menunjukkan bagaimana operator kas digunakan untuk menghasilkan nilai purata yang lebih tepat.

```
int jum_markah = 456, bil_pelajar = 5;
double purata_markah1, purata_markah2;
purata_markah1 = jum_markah/bil_pelajar;
purata_markah2 = (double) jum_markah / bil_pelajar;
```

© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.6 Operator Hubungan dan Logik

- **Operator hubungan** digunakan untuk melakukan perbandingan dua ungkapan.

- Ungkapan menggunakan operator hubungan dinamakan **predikat mudah**. Ungkapan predikat mudah ini digunakan dalam sesuatu kenyataan untuk membuat keputusan.

- Operator hubungan C:

Operator	Penerangan
<	kurang daripada
<=	kurang daripada atau sama dengan
>	lebih daripada
>=	lebih daripada atau sama dengan
==	sama dengan
!=	tidak sama dengan

3.6 Operator Hubungan dan Logik

- Hasil dari operator hubungan ialah samada BENAR (TRUE) atau PALSU (FALSE). Dlm C, PALSU diwakili oleh nombor/nilai sifar (0) dan BENAR oleh sebarang nombor bukan sifar(biasanya 1).

- Contoh:

kenyataan : `y = (4 == 5)` akan memberikan y nilai sifar

kenyataan : `y = (x <= z)` memberikan y nilai bukan sifar jika x lebih kecil atau sama dgn z memberikan y nilai sifar jika x lebih besar dari z

© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.6 Operator Hubungan dan Logik

- Jika ungkapan operator hubungan digunakan bersama pelbagai operator lain, **peraturan keutamaan** perlu dipatuhi.

- Peraturan keutamaan operator dengan operator ungkapan :

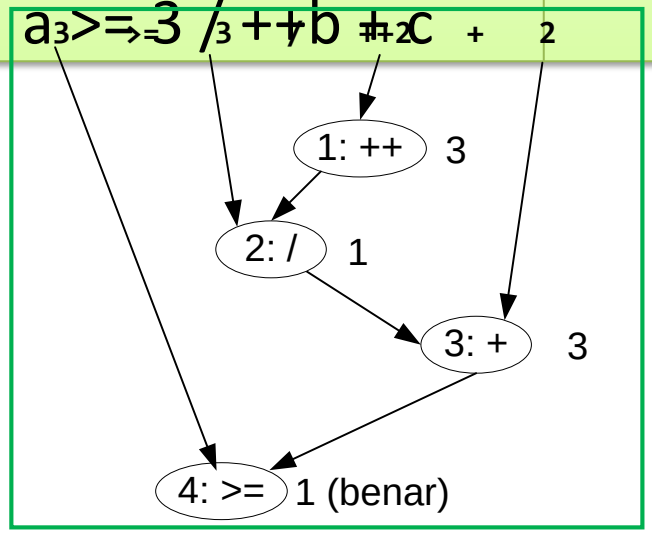
Keutamaan	Operator	Jenis	Kesekutuan
1 (tinggi)	`- + -- ++ ! sizeof (jenis)`	sesatu	kanan ke kiri
2	`* / %`	perduaan	kiri ke kanan
3	`+ -`	perduaan	kiri ke kanan
4	`< <= > >=`	hubungan	kiri ke kanan
5	== !=	hubungan	kiri ke kanan
6	&&	logik	kiri ke kanan
7	||	logik	kiri ke kanan
8 (rendah)	= += -= *= /= %=	umpukan	kanan ke kiri

© Copyright Universiti Teknologi Malaysia

3.6 Operator Hubungan dan Logik

❖ Contoh penggunaan operator hubungan :

Integer			Prediket mudah	Hasil
a	b	C		
1	5	2	-a + b > c	1 (benar)
20	4	5	a != b * c	0 (palsu)
3	2	2	a >= 3 / ++b + c	1 (benar)



© Copyright Universiti Teknologi Malaysia

3.6 Operator Hubungan dan Logik

- Predikat kompleks boleh ditulis dengan menggunakan predikat mudah dan operator logik.

- Operator logik digunakan untuk menggabungkan beberapa ungkapan yang mengandungi operator hubungan.

- Operator logik:

Operator	Penerangan
&&	Dan
||	Atau
!	Tak

© Copyright Universiti Teknologi Malaysia

3.6 Operator Hubungan dan Logik

- Operator && dan || adalah operator perduaan yang memerlukan sekurang-kurang dua kendalian atau ungkapan untuk mendapatkan satu ungkapan logik. Manakala operator ! merupakan operator hanya memerlukan satu.

- Penilaian ungkapan logik boleh dibuat dengan menggunakan jadual kebenaran logik.

- Jadual kebenaran bagi operator && :

ungkapan1	ungkapan2	ungkapan1 && ungkapan2
benar	benar	benar
benar	palsu	palsu
palsu	benar	palsu
palsu	palsu	palsu

© Copyright Universiti Teknologi Malaysia

3.6 Operator Hubungan dan Logik

❖ Jadual kebenaran bagi operator || :

ungkapan1	ungkapan2	ungkapan1 || ungkapan2
benar	benar	benar
benar	palsu	benar
palsu	benar	benar
palsu	palsu	palsu

❖ Jadual kebenaran bagi operator ! :

ungkapan1	!ungkapan1
benar	palsu
palsu	benar

3.6 Operator Hubungan dan Logik

- Ketiga-tiga jadual ini boleh diringkaskan dengan kenyataan jika ungkapan1 dan ungkapan2 adalah ungkapan-ungkapan yang mengandungi operator hubungan mudah seperti **`x < 10, z == 0`**, maka:

ungkapan1 || ungkapan2	adalah benar jika dan hanya jika kedua-dua ungkapan adalah benar
!ungkapan1	adalah benar jika satu atau kedua-dua ungkapan adalah benar
ungkapan1 && ungkapan2	adalah benar jika ungkapan1 adalah palsu dan sebaliknya

© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.6 Operator Hubungan dan Logik

- Aturan kesatuan operator logik dinilai dari kiri ke kanan dan keutamaan operator logik **Tak** dan **Dan** adalah rendah.

- Penggunaan **kurungan** adalah penting untuk menentukan aturan ungkapan dinilai.

- Contoh:

`(6 < 2) && (3==3)`	adalah palsu
`! ( (6 < 2) && (3==3) )`	adalah benar
`j != 0 && (x/10 < 3)`	adalah benar jika j bukan sifar dan x lebih kecil dari 30
`!(12 != 12) || ((2 > 1) && 5 == 2))`	adalah benar

© Copyright Universiti Teknologi Malaysia

3.7 Fungsi Pratakrif Perpustakaan Matematik

- Setakat ini ungkapan matematik ditulis dengan menggunakan operator matematik yang mudah seperti +, -, *, / dan %. Tetapi ini tidak cukup untuk menyelesaikan masalah matematik yang lebih rumit seperti:

$$c = \sqrt{a^2 + b^2}$$

$$\cos\alpha + \cos\beta = 2\cos\tfrac{1}{2}(\alpha+\beta)\cos\tfrac{1}{2}(\alpha-\beta)$$

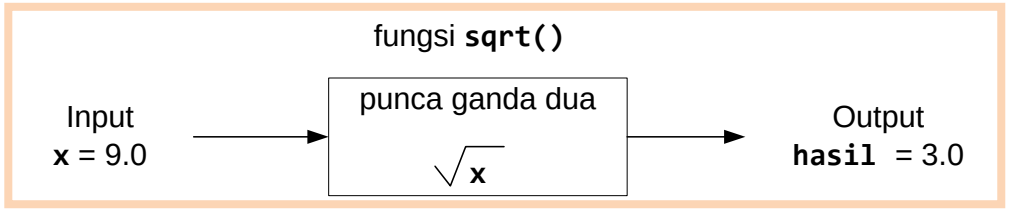
- **Fungsi pratakrif** - Fungsi yang disediakan di dalam perpustakaan piawai C.
- Fungsi ini boleh digunakan dengan memasukkan arahan prapemproses fail kepala.
- Contohnya, untuk menggunakan fungsi `sqrt()` dari perpustakaan math maka arahan prapemproses dengan arahan rangkum berikut diperlukan dalam atur cara.

```
#include <math.h>
```

3.7 Fungsi Pratakrif Perpustakaan Matematik

- Penggunaan fungsi ini adalah seperti operasi kotak hitam yang mana pengguna tidak perlu mengetahui bagaimana fungsi dilaksanakan tetapi hanya perlu mengetahui input dan output fungsi.

- Contoh kotak hitam fungsi sqrt():



© Copyright Universiti Teknologi Malaysia

© Copyright Universiti Teknologi Malaysia

3.7 Fungsi Pratakrif Perpustakaan Matematik

❖ Contoh fungsi matematik yang disediakan di dalam perpustakaan C:

Fungsi	Perpustakaan Piawai	Tujuan dan Contoh	Argumen	Hasil
abs(x)	stdlib.h	$\|x\|$ abs(-10) hasil 10	int	int
exp(x)	math.h	e^x exp (-1.0) hasil 2.71828	double	double
log(x)	math.h	$\log_e x$ log(2.71828) hasil 1.0	double	double
pow(x,y)	math.h	x^y pow (2.0, 3.0) hasil 8	double, double	double
cos(x)	math.h	$\cos x$ cos(45.5) hasil 0.05306	double	double
sqrt(x)	math.h	$\sqrt{x}$ sqrt(9.0) hasil 3.0	double	double

© Copyright Universiti Teknologi Malaysia

3.7 Fungsi Pratakrif Perpustakaan Matematik

- Contoh penggunaan fungsi matematik piawai:

```
1:  #include <stdio.h>
2:  #include <math.h>
3:
4:  int main () {
5:        double sisi_a, sisi_b, sisi_c;
6:        sisi_a = 5.2;
7:        sisi_b = 2.5;
8:
9:        //kira sisi c
10:       sisi_c   =   sqrt((pow(sisi_a,   2.0))   +   (pow(sisi_b,
11: 2.0)));
12:
13:       printf("Panjang hipotenus:  %f", sisi_c);
14:
15:       return 0;
    }
```

- Output:

```
Panjang hipotenus:  5.769749
```

© Copyright Universiti Teknologi Malaysia