CMAS 12A 4-13-10

Chap: statements & control flow

(1) sequential: Default
execute stmts in order

(2) Conditional (Branching)
test a logical condition(s),
if true go one way, if
false another

(3) Iterative (looping)
repeat some statement(s) until
some logical cond. becomes true
or false.

Relational & Logical Operators

Relational OP	name	EX
<	less than	5<10
>	greater than	5>10
==	equals	5==10
<=	less than or eq.	5<=10
>=	greater than or eq.	5>=10
!=	not equal	5!=10

Logical OP	name	EX.
&&	and	(1<2)&&(3==4)
||	or	(1<2)||(3==4)
!	not	!(3==4)

Truth Tables:

let a, b Be boolean variables

a	b	a && b	a || b
F	F	F	F
F	T	F	T
T	F	F	T
T	T	T	T

Binary

a	!a
F	T
T	F

unary

Note: || is inclusive or i.e.

a or b or possibly both are true.

Another meaning of "or" in English is exclusive or:

a or b, but not both.

Ex. ?

$$!((a \&\& !b) || (!a \&\& b))$$

Other notation:

English	Java	Math	Circuits
a and b	a && b	$a \wedge b$	$a \cdot b$
a or b	a || b	$a \vee b$	$a + b$
not a	!a	$\neg a$	$\overline{a}$

Conditional stmts:

- if
- if - else

} Discuss now

- switch (Discuss later)

```
if (condition)
    stmt1;   <-- True Branch
stmt2;
  :
```

or

```
if (condition){
    stmt1;
     :
    stmtK;
}
stmt(K+1);
  :
```

Defn: Compound stmt: any list of stmts within braces {.. }.

Ex.

```
int a = 6, b = 5, temp;
if (a > b) {
    temp = a;
    a = b;
    b = temp;
}
System.out.println(a + " " + b);
```

a	b	temp
6	5	-
6	5	6
5	5	6
(5)	(6)	6

if with else clause:

```
if (cond)
    stmt1; ← True branch
else
    stmt2; ← False branch
stmt3;
  :
```

or

```
if (cond) {
    Stmt1;
     :
    Stmt K;
} else {
    Stmt(K+1);
     :
    Stmtl;
}
Stmtl;
```

True Branch

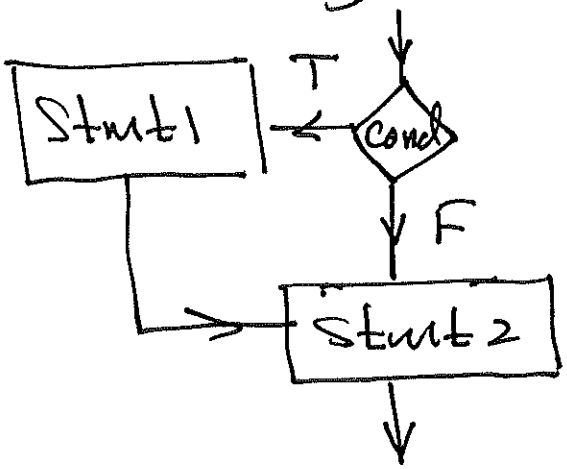
False Branch

Ex.

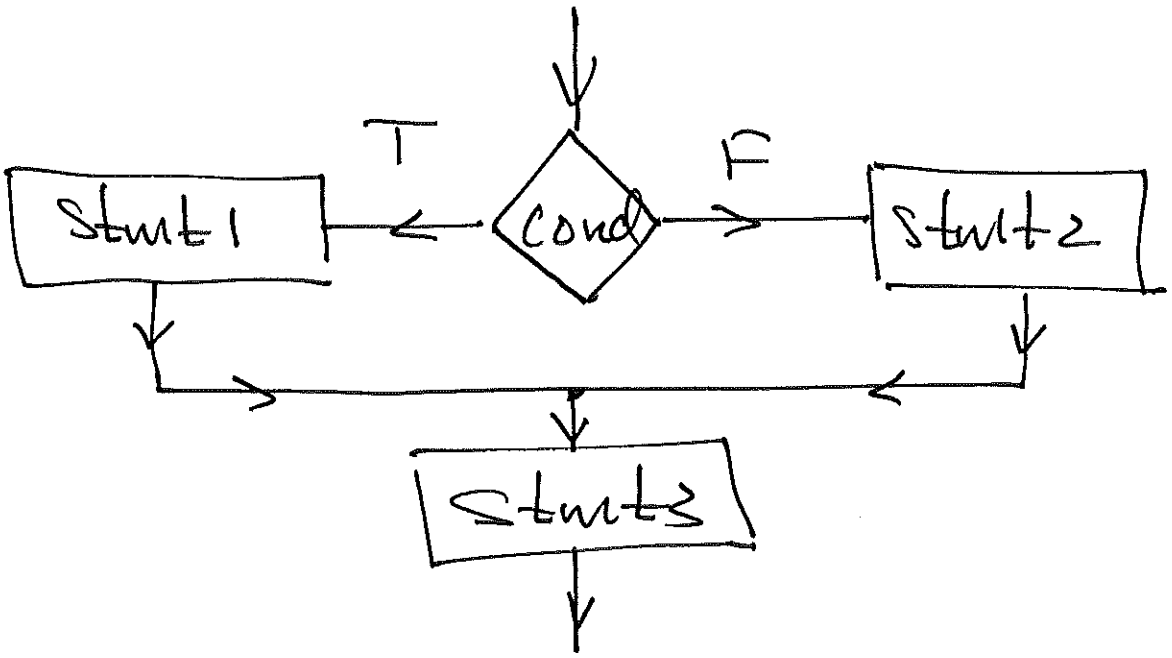
```
int a=5, b=6;
if (a < b)
    System.out.println(a + " " + b);
else
    System.out.println(b + " " + a);
```

Control flow Diagrams:

- if



if - else :



Ex. logical error :

```
if (cond)
    stmt1;
    stmt2;
stmt3;
```

← executed unconditionally

Ex. Syntax error

```
if (cond)
    stmt1;
    stmt2;
else
    stmt3;
stmt4;
```

Ex. logical error

```
if (cond);
    stmt1; <-- unconditioned
stmt2;
```

Ex. style OK if stmt1, stmt2 short

```
if (cond) stmt1;
else stmt2;
```

Ex. OK if really short

```
if (cond) S1; else S2;
```

more style:

Ex.

```
if (cond1)
    if (cond2)
        S1;
    else
        S2;
S3;
```

True

Ex.

```
if (cond1)
    [S1;

else
    [if (cond2)
        [S2;
    else
        [if (cond3)
            [S3;
        else
            [S4;
```

Better style:

Ex.

```
if (cond 1)
    s1;
else if (cond 2)
    s2;
else if (cond 3)
    s3;
else
    s4;
```