Chapter 3 :-

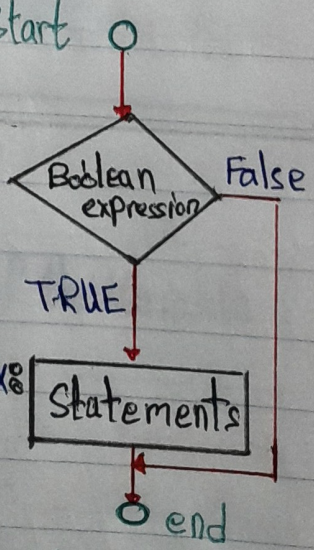
Relational Operator :

1. < : less than
2. <= : Less than or equal to
3. > : greater than
4. >= : greater than or equalto
5. == : equalto
6. != : not equal to

If statements :

Defe is a selection statement which Determine Whether the Sequence that the Program Select and we write it as following Syntax :

```
if (boolean ex.) {        <- expression
    statement(s);
}
```



~~anoth~~ and that it's flow chart :

we can write also as following Syntax :

```
(boolean expression) ? statements ;
```

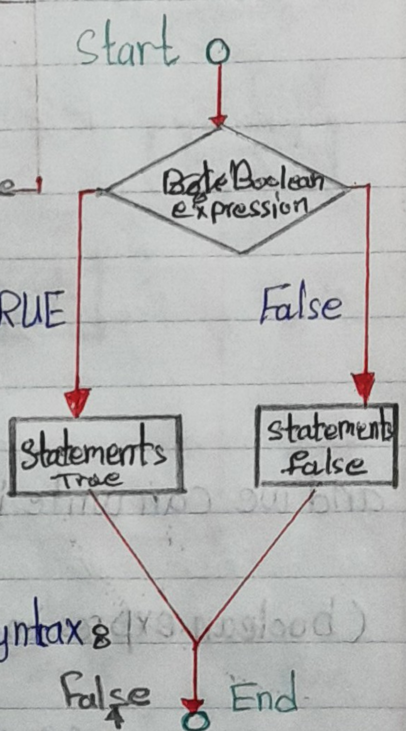
Conditional operator

note you can remove braces when write one statement in if

Two-way if-else Statements:

Def: is a selection Statement which have Two ways to execut your Program and we write as the following Syntax:

```
if (boolean-expre.) {
    Statement(s);      → for true case
}
else {
    Statement(s);      → for false case
}
```



and it's flow chart as Shown
and we can write as following Syntax:

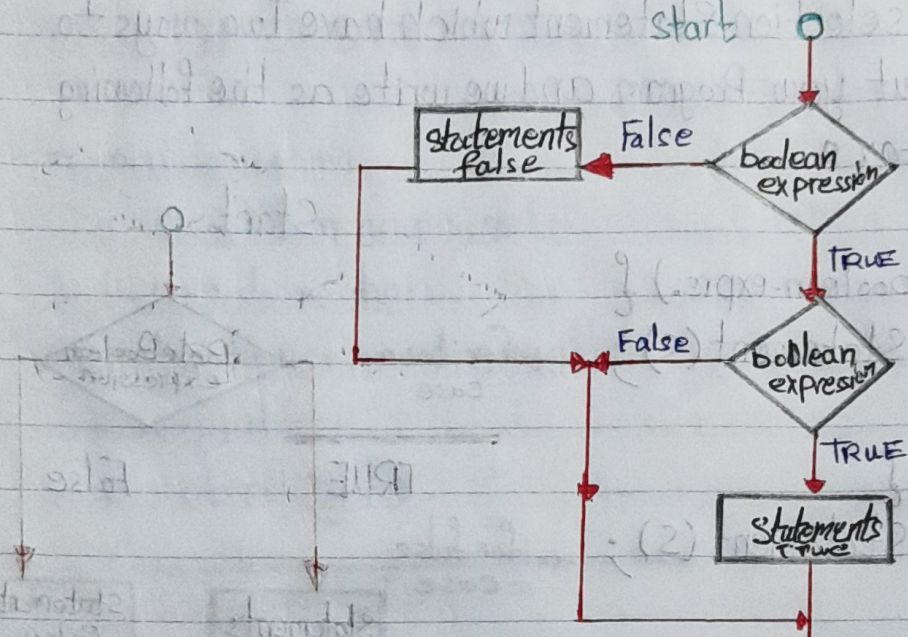
```
                         True          False
(boolean-expression) ? statements : statements;
                     ↓              ↓
                     if             else
```

Nested if and multiway If statements:

Def: The Nested Principle is to make if inside another if and we write it as following Syntax:

```
if (boolean-exp.) {
    if ( boolean-exp. ) {s} ;
```

and it's flow chart as shown



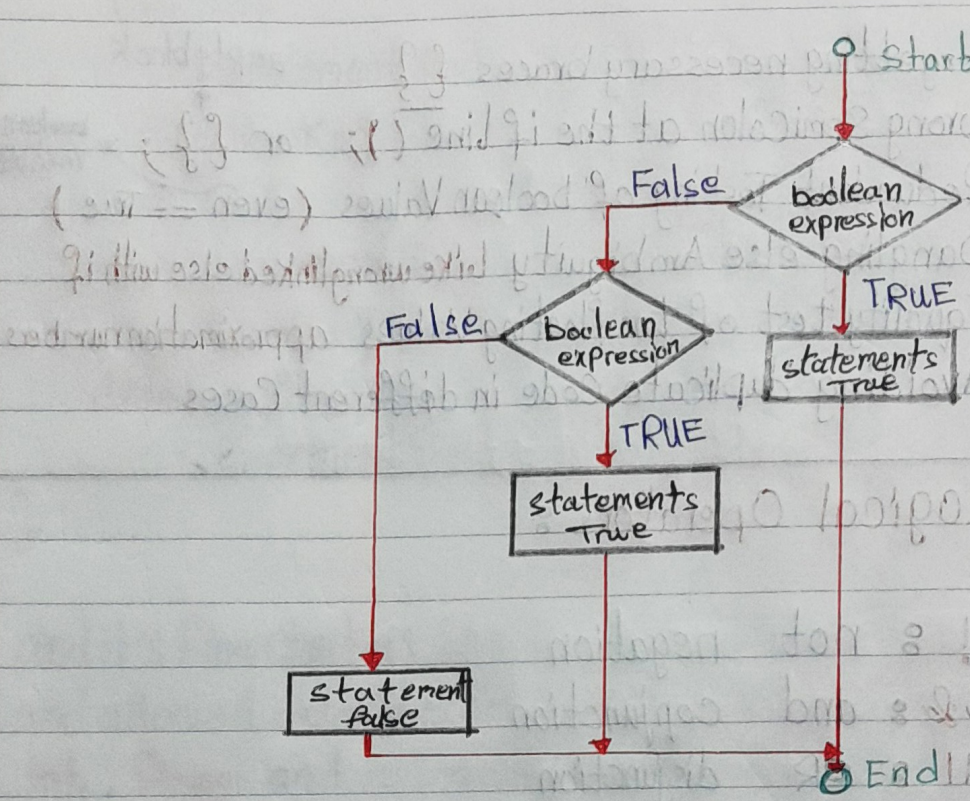
and we can write it as following Syntax: Ternary operator

```
(boolean-expression) ? ((boolean-expresson) ? T : F) :S;
```

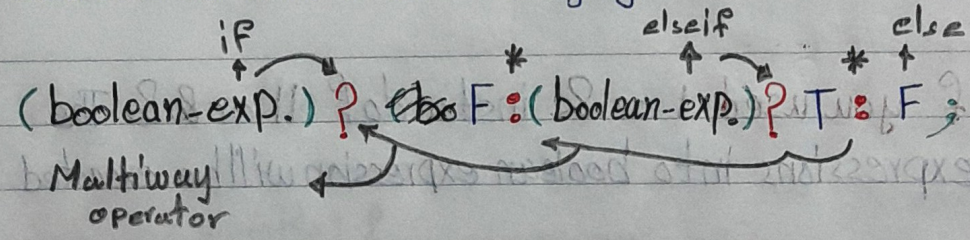
Def: The Multiway if else-if statement is a selection statement which contains more than 2 conditions and we can write it as following Syntax:

```
if (boolean-exp.) {
    statement(s);
} else if (boolean-exp.) {
      statement(s);
} else
      statement(s); }
```

and it's flow chart as shown



and we can write it as following Syntax:



notes: The Difference between The 2 following codes is The else owner nearest if out of braces

```
Block outer if                    Block inner if
if ( boolean ) {                  if ( boolean )
  if ( boolean ) { }                if (boolean) { }
else { } ;                        else ;
```

Common Errors and Pitfalls :

1. forgetting necessary braces {}
2. wrong Semicolon at the if line ( ); or {} ; (empty block; boolean variable)
3. Redundant Testing of boolean Values (even == True)
4. Dangling else Ambiguity Like wrong linked else with if
5. Equality test of two floating Values approximation numbers
6. * Avoiding duplicate code in different Cases

Logical Operators :

1. ! : not negation
2. && : and conjunction
3. || : OR disjunction
4. ^ : exclusive OR exclusion

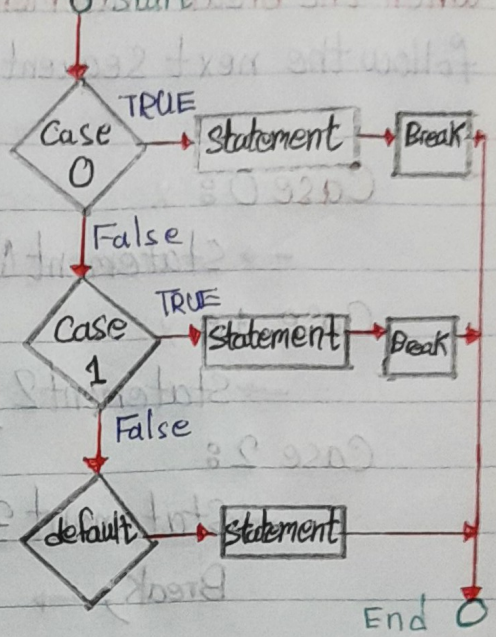
note: if you used | or & instead of (|| or && The) expressions into boolean expression will be executed

---

Switch Statement :

Defs: is a selection statement which more easy than else if Statements that Contains many cases and default Case for abnormall inputs and we can write it as following Syntax :

```
Switch (Variable) {
    Case 0 : {
        Statement 1 ;
        Break ; }
    Case 1 : {
        Statement 2 ;
        Break ; }
    default : {
        Statement 3 ; }
}
```



and it's flowchart as Shown
and allowed datatypes for it are char, byte, short, int, String and it's used in constant Cases not more Than one Variable

note you can use the following Syntax with the similar Cases

```
Switch (Variable) {
    Case 0 :
    Case 1 : {
        Statement 1 ; }
    default :
        Statement 2 ;
}
```

note

when the break statement is removed The program will follow the next sequential

```
Case 0: x
    → statement 1;
Case 1: x
    → statement 2;
Case 2:
    → statement 3; } → if case 2 is true
      Break; →
```

output

: statement 1
statement 2
statement 3

Operator Precedence chart:

1. ~~++~~ i++ and i--
2. +, -, ++i and --i
3. Casting
4. !
5. *, /, %
6. +, - then Relational then ==, !=
7. ∧ then && then ||
8. Augmanted operators and =

Debugging & Steps

1. Executing a single statement at time
2. Tracing into or stepping over a method
3. Setting Break points ◆ intellij ● netbeans
4. Displaying Variables
5. Displaying Call stacks
6. Modifying Variables