CONTROL STATEMENT

- To change the execution order of program
- As the method of controlling the execution order
  - Conditional Statement : **if** St., **switch** St.
  - Repeat Statement : **for** St., **while** St., **do-while** St.
  - Branch Statemnt : **break** St., **continue** St., **return** St.

CONDITIONAL STATEMENT - IF STATEMENT

- FORM OF IF STATEMENT
  - IF (<CONDITIONAL EXPRESSION>) <STATEMENT>
  - IF (< CONDITIONAL EXPRESSION >) <STATEMENT1> ELSE <STATEMENT2>

```
if (a < 0)  a = a++;
if (a > b)  m = a;  else m = b;
```

UE OR FALSE)

CONDITIONAL STATEMENT - IF STATEMENT

- NESTE

```
if (<cond. expr.>)
        if (<cond. expr.>)
              // . . .
              <statement>
```

```
if (<cond. expr.1>)          <statement 1>
else if (<cond. expr.2>)  < statement 2>
          …
else if (<cond. expr. n>)  < statement n>
else  < statement>
```

CONDITIONAL STATEMENT - SWITCH STATEMENT

- FORM OF SWITCH STATEMENT

```
switch ( <expr.> )  {
        case <const. expr. 1> : <statement 1>
        case < const. expr. 2> : < statement 2>
                :
        case < const. expr. n> : < statement n>
        default : < statement>
  }
```

REPEAT STATEMENT - FOR STATEMENT

- REPEAT THE SEQUENCE OF STATEMENT AS MANY AS DEFINED.
- FORM OF FOR STATEMENT

- **FOR (<EXPR. 1> ; < EXPR. 2> ; < EXPR. 3>)**

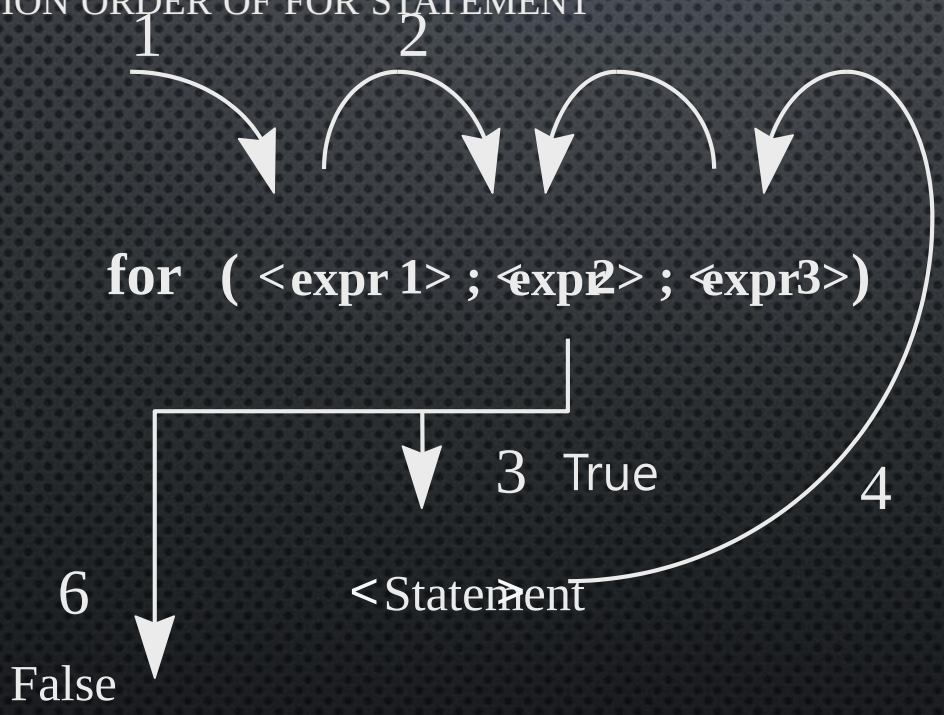
  <STATEMENT>

  - <EXPR. 1> : INITIALIZE THE CONTROL VARIABLE
  - <EXPR. 2> : CHECK THE CONTROL VARIABLE

```
s = 0;
for (i=1; i<=N; ++i)    // sum from 1 to N : i increment
        s += i;
```

REPEAT STATEMENT - FOR STATEMENT

- EXECUTION ORDER OF FOR STATEMENT



REPEAT STATEMENT - FOR STATEMENT

- INF

```
for ( ; ; )
    <statement>
```

  - TO STOP THE LOOP : BREAK STATEMENT, RETURN STATEMENT

- NESTED FOR STATEMENT
  - MULTI DIMENSIONAL ARRAY

```
for (i=0; i<N; ++i)
    for (j=0; j<M; ++j)
        matrix[i][j] = 0;
```

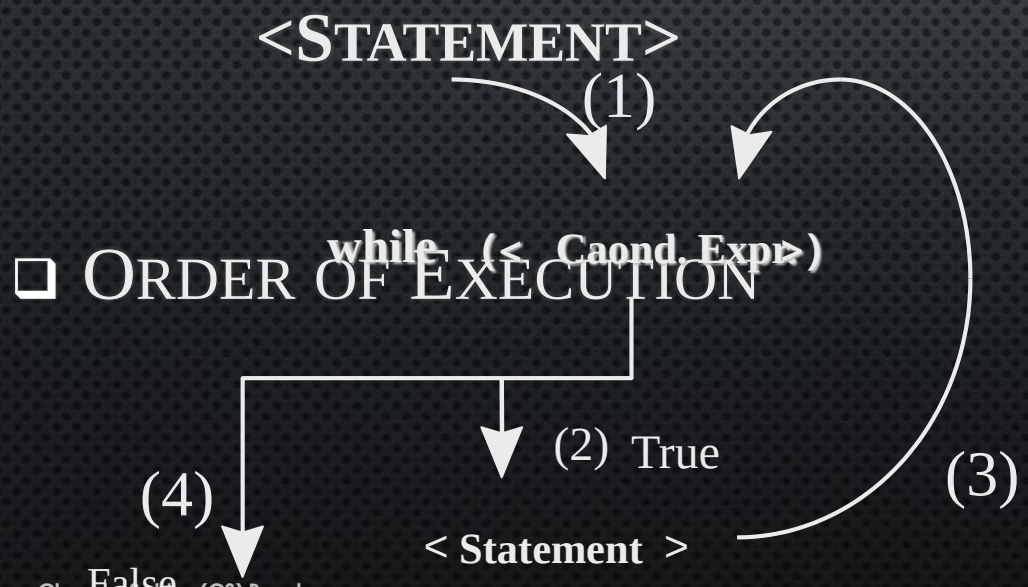
REPEAT STATEMENT - WHILE STATEMENT

- FORM OF WHILE STATE

WHILE (COND. EXPR.

```
  i = 1;  s = 0;
while (i <= N) {// summation from 1 to N
        s += i;  ++i;
  }
```

- ORDER OF EXECUTION



REPEAT STATEMENT - WHILE STATEMENT

- (…EMENT ANI

```
for (i = 0; i < N; ++i)
      s += i;
```

```
i = 0;
while (i < N) {
      s += i;
      ++i;
}
```

REPEAT STATEMENT – DO - WHILE STATEMENT

- After executing the repeating statements, then check the conditional expression
- Form of do-while statement

```
DO
    <STATEMENT>
WHILE (<CONDITIONAL EXPRESSION>);
```

Although the conditional expression
is false, execute the statement
one time above at least

BRANCH STATEMENT - BREAK STATEMENT

- TO MOVE CONTROL TO THE OUT OF THE BLOCK
- FROM OF BREAK STATEMENT

B

```
int i = 1;
while (true) {
    if (i == 3)
        break;
    System.out.println("This is a " + i + " iteration");
    ++i;
}
```

BRANCH STATEMENT – CONTINUE STATEMENT

- To move control to the start of next repeatation
- From of continue statement

- **continue [Label] ;**

- W

```
for (i=0; i<=5; ++i) {
        if (i % 2 == 0)
                continue;
        System.out.println("This is a " + i + " iteration");
}
```