Writing Conditions (Class Notes)

16/05/2024

Author: Subhasis Das

Conditions are statements which return true or false, Conditions are written using relational and logical operators, Sometimes, Arithmetical Operators are also used. Conditions look like : a>b, a<b, a>b && a<b , a!=b, a==b, a+b==1

Arithmetic Operators: + , - , * , / (division) , % (modulo)

Precedence of Arithmetic Operators: * / and % have same priority, after them: + and -

relational Operators : > (greater than)
< (less than)
>= (greater than or equal to)
<= (less than or equal to)
== (equals to)
!= (not equal to)

Logical Operators: ! (Logical NOT) ** Reverses meaning of a condition
&& (Logical AND) ** Returns true if both conditions are true
|| (Logical OR) ** Returns true if at least one condition is true

condition1 && condition2 **:** this combined condition returns true if condition1 and condition2 both are true

condition1 || condition2 : this combined condition returns true if at least one of the conditions is true or both are true.

Precedence of logical operators: ! (NOT) && (AND) || (OR)
**precedence means order of execution (like BODMAS in Mathematics)

Lets see a few conditions….

To check whether a number A is greater than B or not.
A > B : if this condition returns true, we can say A is greater than B

To check whether a number A is smaller than B or not.
A < B : if this condition returns true, we can say A is smaller than B

To Check if A and B are equal
A==B : if this condition returns true, we can say A and B are equal

To check if A and B are unequal
A != B : if this condition is true, we can say A is not equal to B

To check if A is greater than or equal to B
A>=B: if this condition is true, we can say A is greater than or equal to B

To check if A is smaller than or equal to B
A<=B: if this condition is true, we can say A is smaller than or equal to B

Examples:

To check if A is greater than 5:	A>5
To check if B is less than 1 :	B<1
To check if C is equal to 2:	C==2
To check if sum of a and b is equal to 10:	a+b == 10
to check if k is not equal to 3:	k != 3
to check if n is greater than or equal to 10:	n >= 10
to check if p is smaller than or equal to k:	p<=k

Checking Multiple Conditions:

***** We need logical operators to check multiple conditions:***

To check if a number N is between 10 and 20 (excluding 10 and 20):
N>10 && N<20

To check if a number N is between 10 and 20 (including 10 and 20)
N>=10 && N<=20

To check if N is 10 or 20 (here N needs to be exactly 10 or 20)
N== 10 || N==20

To check if N is not 10 and not 20
N != 10 && N != 20

to check if N is anything except between 10 and 20
! (N>=10 && N<=20) can also be written as N<10 || N>20

Divisibility Check:

*****A is divisible by B if we get 0 as remainder after dividing A by B. to get remainder we can use modulo (%)***

To check if A is divisible by 2: A%2 == 0 (it means that we are getting 0 as remainder after dividing A by 2)

To check if N is divisible by 5: N % 5 ==0

to check if N is divisible by i : N %i==0

To check if A is divisible by B: A%B ==0

to check if N is a multiple of 3: N%3==0

The statements N is multiple of 3 and N is divisible by 3 are same

** Getting the last digit of a number

To get the last digit of a number N , we need to divide N with 10 and take the remainder , the remainder is the last digit. k= N% 10; here k stores the last digit of N

Examples 12 %10 is 2
34 % 10 is 4
100 % 10 is 0
45% 10 is 5
78 % 10 is 8
99 % 10 is 9

If we write **k=123 % 10;** k will store 3 as 3 is the last digit of 123

To check if a number N ends with 5 : N % 10 == 5
To check if a number N ends with 7: N%10 ==7
To check if a number N ends with P : N%10 ==P

***** we use these conditional statements in conditional constructs (if -else if -else) and in conditional operator (ternary)***

A few examples of how to use conditional statements (**Although we will learn conditional construct in next chapter**)

Ternary Syntax: *(condition) ? if true : if false;*

double k= (5<4) ? 1 : 0; here k will store 0 as the condition is false

String s= (7> 2) ? "HI" : "Hello"; here s will store "HI" as the condition is true

int p= (2==4) ? 5 : 25; here p will store 25 as the condition is false

If-else syntax:

if(condition) {

Codes inside this block execute if condition is true

}**else**{

Codes inside this block execute if condition is false
}

Example:

```
if(n > 0){
System.out.println("Positive Number");
}else{
System.out.println("Negative Number");
}
```

*** here, if n is greater than 0 "Positive Number" is printed otherwise it prints "Negative Number"*

Conversion from if else to ternary:

if-else:

```
int p;

if(a>b){
p=1;
}else{
p=2;
}
```

Using ternary:

```
int p= (a>b) ? 1 : 2 ;
```

Chained Ternary Example:

Find greatest between a,b and c: `int max= (a>b && a>c) ? a : (b>c) ? b : c;`

Modular Arithmetic (Basic)

1. To get last k digits of n: $n \% 10^k$
2. To remove last k digits: $n / 10^k$
3. To get first k digits: $n / 10^{c-k}$ [c = count of digits]

Conditional Constructs

1. ***Ternary (Already discussed in this document)***

2. ***only if, if-else, if-else if- else ladder***

if statement:

Syntax: if(condition) {
//code
————
}
here, if the condition is true only then the block of code will get executed.

Observe that I have used {} (Curly braces) to make the code block. Conditions can also be written without curly braces, for example.
if(condition) code;
** It should be noted that in this case only the first statement after the condition is dependent on the condition. which means only the first statement after the condition is part of the code block associated with the condition.

if -else statement:
Syntax: if(condition){
//code
————-
}else{
//code
————-
}
Here, if the condition is true- the if block is executed otherwise the else block is executed

if-else if-else statement:
Syntax: if(condition){
//code
————-
}else if(condition){
//code
————-
}else{
//code
}
** You can add as many **else if(condition)** as you want
** else block is optional

3. **switch-case:**
Here, a variable is passed to the switch as parameter and it checks for equality with case values.
Syntax:

```
switch(choice){
case 1:     //Code
            break;
case 2:     //Code
            break;
case 3:
            //Code
            break;
……
default:
}
```

** default case is optional in switch-case

** **break** is keyword in java, it is used to terminate a construct in java
** **System.exit(0)** is used to terminate whole program
** Absence of break causes fall through in java

Examples:

```
switch(a){
case 1: System.out.println("One");
        break;
case 2: System.out.println("Two");
        break;
case 3: System.out.println("Three");
        break;
case 4: System.out.println("Four");
        break;
default: System.out.println("Invalid");
}
```

Outputs based on value of a:

a=1 : One

a=2: Two

a=3: Three

a=4: Four

a=10: Invalid

Example with ***Fall through:***

```
switch(a){
case 1: System.out.println("One");
        break;
case 2: System.out.println("Two");

case 3: System.out.println("Three");

case 4: System.out.println("Four");
        break;
default: System.out.println("Invalid");
}
```

Outputs based on value of a:

a=1 : One

a=2: Two
Three
Four

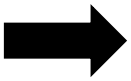
a=3: Three
Four

a=4: Four

a=10: Invalid

Logical OR using switch case with fall through

```
If(a==1){
System.out.println("One");
}else if(a==2 || a==3){
System.out.println("Two or Three");
}else{
System.out.println("None");
}
```



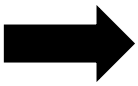
```
switch(a){
case 1: System.out.println("One");
        break;

case 2:
case 3: System.out.println("Two or Three");
        break;

default: System.out.println("None");
}
```

Logical AND using nested switch-case

```
If(a==1 && b==2){
    System.out.println(“Hello”);
}else{
    System.out.println(“Bye”);
}
```



```
switch (a) {
   case 1:
         switch (b) {
               case 2:
                     System.out.println("Hello");
                     break;
                default:
                     System.out.println("Bye");
         }
         break;
   default:
      System.out.println("Bye");
}
```

Check if n is even or odd using switch case:

```
switch(n%2){
    case 0:  System.out.println(“Even”);
             break;
    default: System.out.println(“Odd”);
}
```

Check if n ends with 5 or not:

```
switch(n%10){
    case 5:  System.out.println(“Ends with 5”);
             break;
    default: System.out.println(“Doesn’t end with 5”);
}
```

Exercise:

1. Find second smallest using ternary
2. Check if a number is a multiple of 5 or not using switch-case
3. Check if a number ends with 5 and is a multiple of 7 or not (Hint: using nested switch-case)
4. Check if a number ends with 2, 3 or 5 using switch case
5. Convert the following if-else construct to switch-case:

```
If(a==1){
    K=0;
}else if( a==2 || a==3 ){
    K=1;
}else if( a==4 && a==5 ){
    K=2;
}else{
    K=100;
}
```