清森学校
BEIJING QINGSEN SCHOOL

AP-CSA Boolean Expressions, if statements

YING HUANG

OCT.2022

BOOLEAN EXPRESSIONS

REVIEW - TYPE boolean

- **boolean:** A logical type whose values are true and false.

  It is legal to:

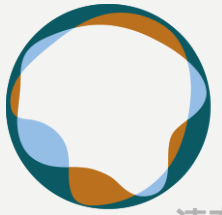
  – create a **boolean** variable

  – pass a **boolean** value as a parameter

  – return a **boolean** value from methods

- Example:

```
int age = 22;

boolean minor = age < 21;

boolean lovesAPCS = true;
```



RELATIONAL EXPRESSIONS

Booleans can also use Relational Operators to store true/false information

Relational Operators			
Operator	**Meaning**	**Example**	**Value**
==	equals	1 + 1 == 2	true
!=	does not equal	3.2 != 2.5	true
<	less than	10 < 5	false
>	greater than	10 > 5	true
<=	less than or equal to	126 <= 100	false
>=	greater than or equal to	5.0 >= 5.0	true

RELATIONAL EXPRESSIONS

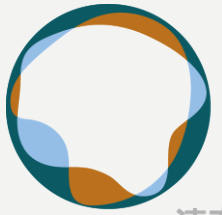
Example (no if):

Write a program that asks the user for their age. Print out if the user is old enough to vote.

Output:

Enter your age: 19

Old enough to vote: true



RELATIONAL EXPRESSIONS

Answer:

```
Scanner input = new Scanner(System.in);
System.out.print("Enter your age: ");
int age = input.nextInt();
boolean oldEnoughVote = age>=18;
System.out.println("Old enough to vote:  "+oldEnoughVote);
```

CODEHS –3.1.6

Write a program that helps the user manage their goals.

Ask for two integers - the first is their goal amount and the second is the actual amount.

Then create three boolean variables, as described below, to help the user determine whether they met their goal.

- The first boolean should be true if the user went over their goal.
- The second boolean should be true if the user was under their goal.
- The third boolean should be true if the user met their goal exactly.

Then print the results (that is, the value of the booleans).

Here is an example:

```
Enter your goal: ↵
45↵
Enter your actual amount: ↵
35↵
Went over goal? false↵
Did not meet goal? true↵
Met goal exactly? false↵
```

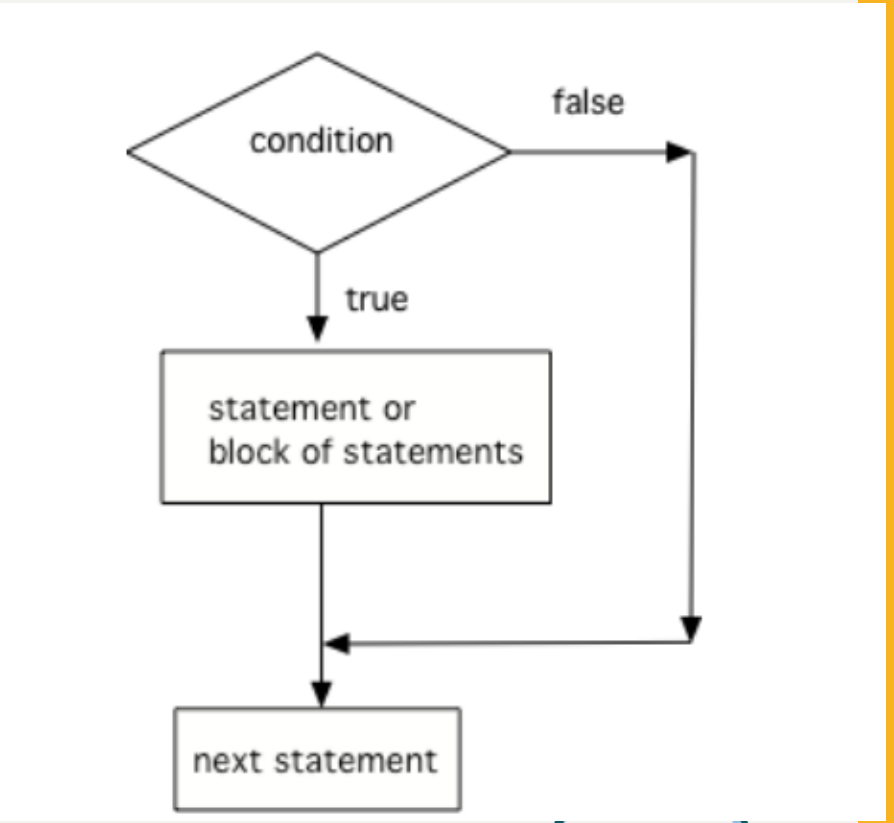
CODEHS –3.1.6 -KEY

```java
Scanner input = new Scanner(System.in);

// Ask for goal amount
System.out.println("Enter your goal: ");
int goal = input.nextInt();

// Ask for actual amount
System.out.println("Enter your actual amount: ");
int actual = input.nextInt();


// Compare numbers by creating three booleans
boolean over = actual > goal;
boolean under = actual < goal;
boolean metGoal = actual == goal;

// Display results as instructed
System.out.println("Went over goal? " + over);
System.out.println("Did not meet goal? " + under);
System.out.println("Met goal exactly? " + metGoal);
```

IF STATEMENT

THE if STATEMENT

- If the condition is true then the next statement or a block of statements will execute.
- If the condition is false then the next statement or block of statements is skipped.



THE if STATEMENT

Syntax:

if(condition is true){

… // statement or a block of statements

}

… //next statement

Code:

```
double gpa = 2.1;
if (gpa >= 2.0) {
    System.out.println("Application accepted.");
}
```

THE if STATEMENT

Example :

Write a program that asks the user for their age. Print out if the user is old enough to vote.

Output1:

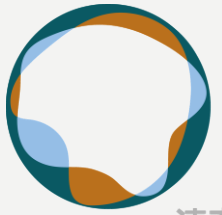
Enter your age: 19

You can vote!

Output2:

Enter your age: 10

Sorry, you're not old enough to vote!



THE if STATEMENT

Answer:

```
Scanner input = new Scanner(System.in);
System.out.print("Enter your age: ");
int age = input.nextInt();
if(age>=18){
    System.out.println("You can vote!");
}
if(age<18){
    System.out.println("Sorry, you're not old enough to vote!");
```

THE if STATEMENT

Answer:

```
Scanner input = new Scanner(System.in);
System.out.print("Enter your age: ");
int age = input.nextInt();
if(age>=18)
    System.out.println("You can vote!");
if(age <18)
    System.out.println("Sorry, you're not old enough to vote!");
```

THE if/if/if STATEMENT

What is the value of grade when the following code executes and score is 93?

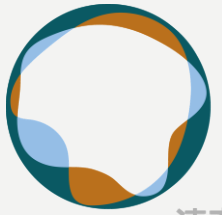
```
if (score >= 90) grade = "A";
if (score >= 80) grade = "B";
if (score >= 70) grade = "C";
if (score >= 60) grade = "D";
else grade = "E";
```

Each of the if statements will be executed.
So grade will be set to A, then B then C and finally D.

THE if/if/if STATEMENT

PRACTICE:

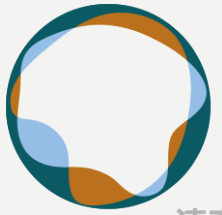
```
int a = 0;
if(a>0)
    a += 1;
    a += 2;

if(a % 3 == 0)
    a += 3;

System.out.println(a);
```



THE if/if/if STATEMENT

PRACTICE:

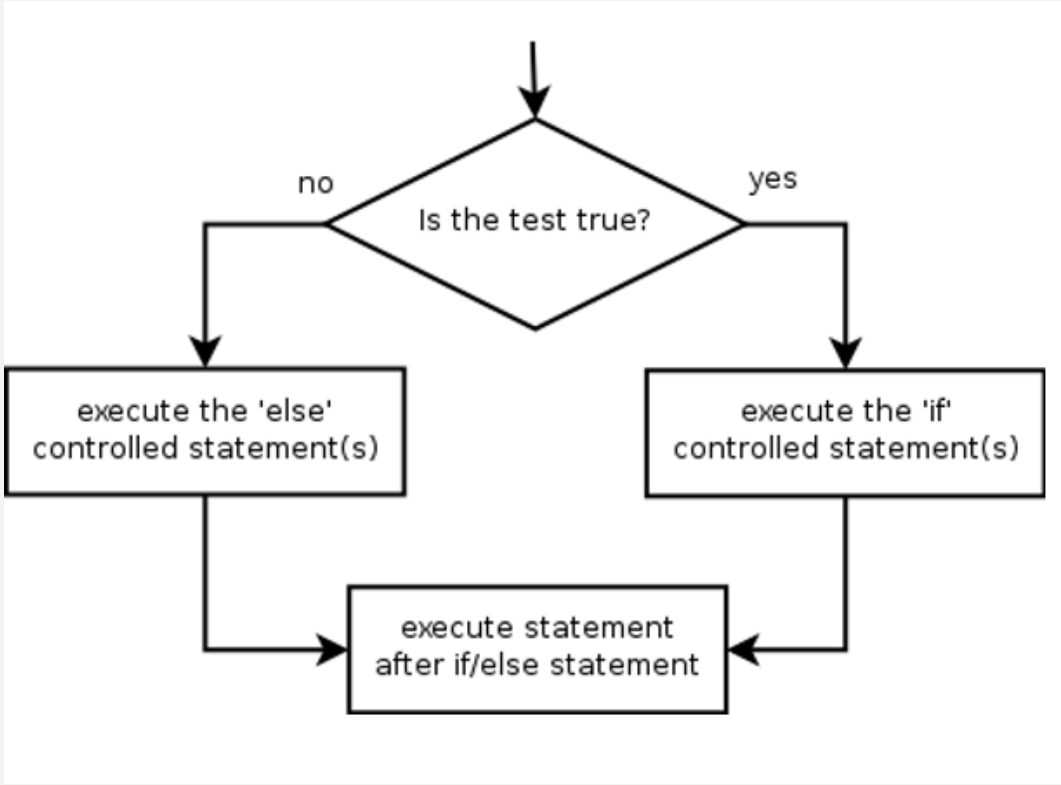
```
int i = 5;
int j = 6;
if(j > i)
    j --;
if(i <= j)
    i ++;
if(i == j)
    j --;
System.out.println(i);
System.out.println(j);
```



清森学校
BEIJING QINGSEN SCHOOL

THE if/else STATEMENT

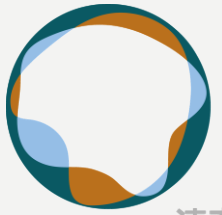
If the initial Boolean expression evaluates to false, else statements execute.



THE if/else STATEMENT

Syntax:

```
if(condition is true){
        … // statement or a block of statements
}
else{
        … // statement or a block of statements
}
… //next statement
```



THE if/else STATEMENT

Code:

```
Scanner input = new Scanner(System.in);
System.out.print("Enter your age: ");
int age = input.nextInt();
if(age>=18){
    System.out.println("You can vote!");
}
else{
    System.out.println("Sorry, you're not old enough to vote!");
}
```

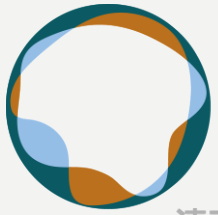
CODEHS – 3.3.5

Currently, Hicham El Guerrouj of Morroco holds the record for running the fastest mile. He ran the mile in 3 minutes 43 seconds, which is 223 seconds.

Write a program that asks the user how fast they can run a mile (in seconds).

Use an if-else to determine if the number is less than the fastest recorded mile time.

- If the user's time is faster than 223 seconds, print "Are you sure? That's a new world record!"
- Otherwise, print, "Right on! Keep running!"



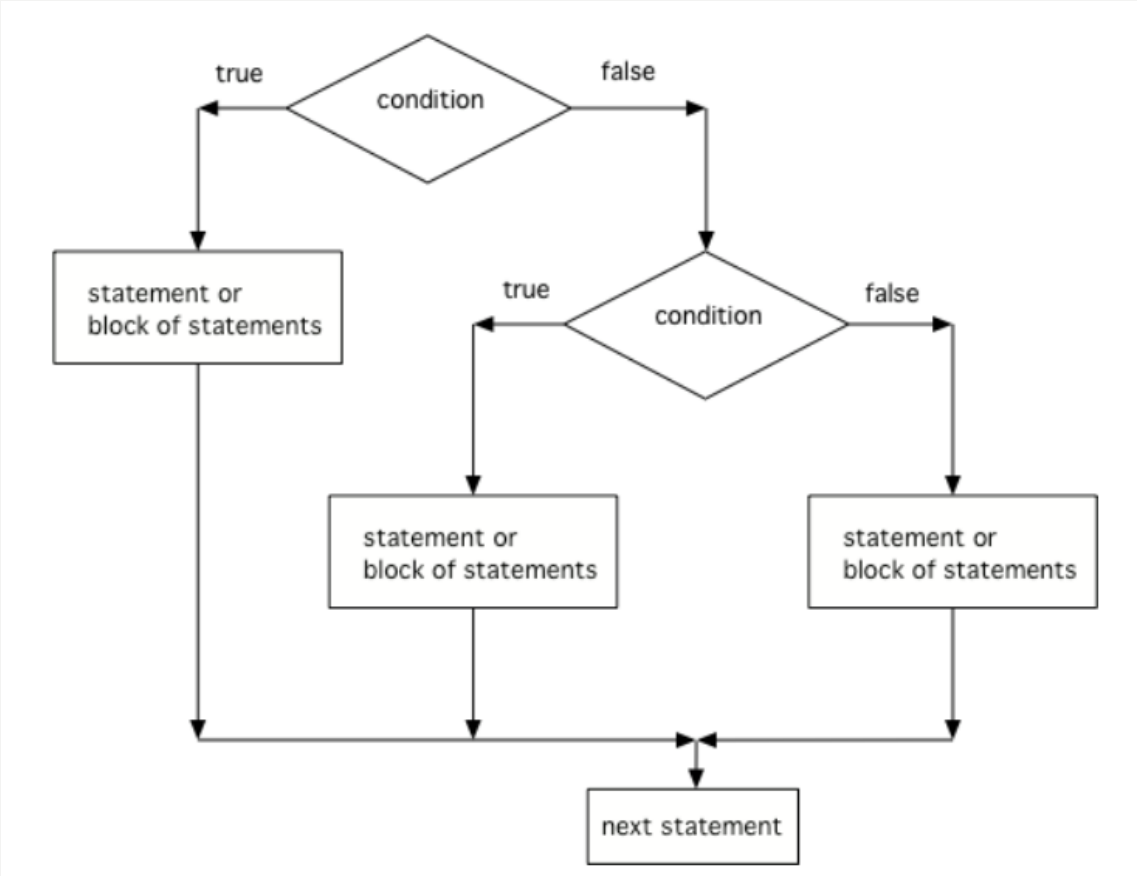
CODEHS – 3.3.5 -KEY

```
Scanner input = new Scanner(System.in);

System.out.println("Enter your mile time in seconds: ");
int num = input.nextInt();

if(num < 223)
{
    System.out.println("Are you sure? That's a new world record!");
}
else
{
    System.out.println("Right on! Keep running!");
}
```

MULTI-BRANCHES

THE if/ else if/ else STATEMENT

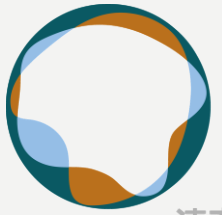
You can even pick between 3 or more possibilities. Just add else if for each possibility after the first if and before the last possibility, the else.



THE if/ else if/ else STATEMENT

Syntax:

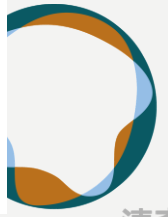
```
if(condition 1 is true){
        … // statement or a block of statements
}
else if (condition 2 is true){
        … // statement or a block of statements
}
else if (condition 3 is true){
        … // statement or a block of statements
}
……
else{
        … // statement or a block of statements
}
… //next statement
```



THE if/ else if/ else STATEMENT

Code:

```
int x = 2;
if (x < 0) {
    System.out.println("x is negative");
}
else if (x == 0) {
    System.out.println("x is 0");
}
else {
    System.out.println("x is positive");
}
System.out.println("after conditional");
```



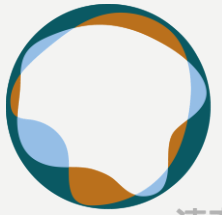
THE if/ else if/ else STATEMENT

PRACTICE

Input your age, and output the information below:

- "You are a minor!" if the age is less than 18.
- "You are a young man!" if the age is more than or equal to 18 and less than 40;
- "You are a middle-aged man!" if the age is more than or equal to 40 and less than 60;
- "You are an old man!" if the age is more than 60;

Then output your age.



THE if/ else if/ else STATEMENT

PRACTICE – KEY

```
System.out.println("Please enter your age:");
int age = io.nextInt();
if(age < 18) {
    System.out.println("You are a minor!");
}
else if(age < 40) {
    System.out.println("You are a young man!");
}
else if(age < 60) {
    System.out.println("You are a middle-aged man!");
}
else {
    System.out.println("You are an old man!");
}
System.out.println("Your age is "+age);
```

CODEHS 3.4.7

Every year, salmon return from the salt water they usually live in to a freshwater river to spawn (lay eggs). Some people like to go and watch the salmon swim upstream to their favorite spawning places, especially at fish farms.

There are several different varities of salmon. Some spawn in the spring and most spawn in the fall.

Your task is to write a program that takes the month of the year as an integer and outputs if it is "Spring spawning season", "Fall spawning season", or "Not spawning season".

The spring spawning season lasts from March to June (month 3 - 6). The fall spawning season lasts from September to November (month 9 - 11).

Remember that in an if-else if statement, the first if statement that evaluates to true is the one whose body is executed. Therefore, you will need to check the months in numerical order.

CODEHS 3.4.7 KEY

```
Scanner input = new Scanner(System.in);

System.out.println("Enter month of year as a number: " );
int month = input.nextInt();

// Strings not necessary, but useful
// since there's a lot of repetition
String notSpawning = "Not spawning season";
String fall = "Fall spawning season";
String spring = "Spring spawning season";


if(month < 3)
{
    System.out.println(notSpawning);
}
else if(month < 7)
{
    System.out.println(spring);
}
else if(month < 9)
{
    System.out.println(notSpawning);
}
else if(month < 12)
{
    System.out.println(fall);
}
else
```

CONDITIONAL STATEMENTS

	If	else if	else
Necessity	Yes, and start a conditional statement	Optional	Optional
Number of Statements	Can only have one	Unlimited	Can only have one

CONDITIONAL STATEMENTS "BOOLEAN ZEN“ - 1

- Students new to *boolean* often test if a result is *true*:

  Suppose **isPrime(n)** returns whether n is prime(a boolean).

```
if (isPrime(57) == true) {    // bad
        ...
}
```

  But this is unnecessary and redundant. Preferred:

```
if (isPrime(57) ){            // good
        ...
}
```

CONDITIONAL STATEMENTS "BOOLEAN ZEN“ - 1

- A similar pattern can be used for a false test:

```
if (isPrime(57) == false) {    // bad
        ...
}
```

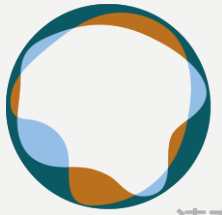
Note: **! is the "not" operator**, which flips the *boolean* value from true to false and false to true.

```
if (! isPrime(57) ){           // good
        ...
}
```

CONDITIONAL STATEMENTS "BOOLEAN ZEN“ - 2

- Methods that return *boolean* often have an if/else that returns true or false:

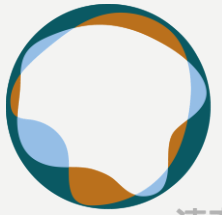
```
public static boolean odd(int n) {
    if (n % 2 != 0) {
        return true;
    } else {
        return false;
    }
}
```



CONDITIONAL STATEMENTS "BOOLEAN ZEN“ - 2

- Methods that return *boolean* often have an if/else that returns true or false:

```
public static boolean odd(int n) {
    if (n % 2 != 0) {
        return true;
    } else {
        return false;
    }
}
```



CONDITIONAL STATEMENTS "BOOLEAN ZEN“ - 3

- We could store the result of the logical test.

```
public static boolean both(int n) {
    boolean test = n % 2 != 0;
    if (test) {   // test == true
        return true;}
    else {      // test == false
        return false; }
}
```

- Notice: Whatever test is, we want to return that.
  - If test is true , we want to return true.
  - If test is false, we want to return false.

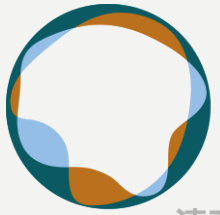
CONDITIONAL STATEMENTS "BOOLEAN ZEN“ - 4

- Observation: The if/else is unnecessary.
  - The variable test stores a *boolean* value;
    its value is exactly what you want to return. So return that!

```
public static boolean odd(int n) {
        boolean test = n % 2 != 0;
        return test; }
```

- An even shorter version:
  - We don't even need the variable test.
    We can just perform the test and return its result in one step.

```
public static boolean odd(int n) {
    return n % 2 != 0;
}
```



HOMEWORK

Codehs:

Unit 2

2.9.6

2.9.7

2.9.8

2.10.6

2.10.7

2.10.8

Unite 3

3.1.6 (no if)

3.1.7(no if)

3.1.8 (no if)

3. 2.6

3. 2. 9

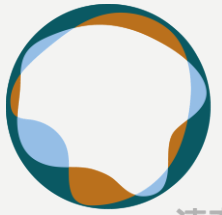
3. 3. 5

3. 3. 4

3. 3. 8

3.4. 7

3.4. 9



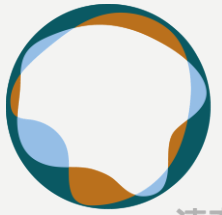
清森学校
BEIJING QINGSEN SCHOOL

REVIEW - PASSWORD

The method passwordStrength() should

print ”Password Strength: Good” if the Password is greater than 8 characters;
print “Password Strength: Bad – add more characters”, if the password is 8 or less characters.

```
public void passwordStrength(String password){

        #statement

}
```



清森学校
BEIJING QINGSEN SCHOOL

REVIEW - PASSWORD

The method passwordStrength() should

print ”Password Strength: Good” if the Password is greater than 8 characters;
print “Password Strength: Bad – add more characters”, if the password is 8 or less characters.

```
public void passwordStrength(String password)
{
    if(password.length() >8)
        System.out.println("Password Strength: Good");
    else
        System.out.println("Password Strength: Bad - add more characters");
}
```

LOGICAL OPERATOR

LOGICAL OPERATOR

Logical operators allow programs to make decisions based on multiple conditions

Operator	Description	Example	Result
`&&`	and	`(2 == 3) && (-1 < 5)`	`false`
`||`	or	`(2 == 3) || (-1 < 5)`	`true`
`!`	not	`!(2 == 3)`	`true`

LOGICAL OPERATOR

Truth table for AND &&

P	Q	P&&Q
true	true	true
true	false	false
false	true	false
false	false	false

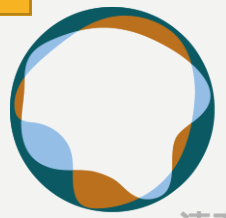
LOGICAL OPERATOR AND &&

Example:

```
(1 > 0) && (3 > 2)
```

```
( 2 >= 1 )  &&  (3 < 2)  &&  (1 > 0)
```

```
( 2 < 1 )  &&  (3 < 2)  &&  (1 > 0)
```



LOGICAL OPERATOR

Truth table for or ||

P	Q	P||Q
true	true	true
true	false	true
false	true	true
false	false	false

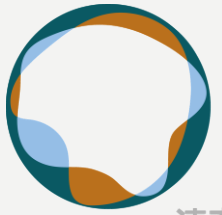
LOGICAL OPERATOR OR ||

Example:

```
(2 > 3) || (1 > 0)
```

```
(2 < 3) || (1 > 0) || (1 > 1)
```

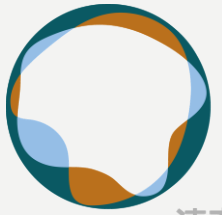
```
(2 < 1) || (1 > 1) || (2 == 0)
```



LOGICAL OPERATOR

Truth table for not ！

P	!P
true	false
false	true



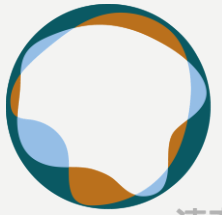
LOGICAL OPERATOR NOT !

Example:

```
! (I > 0)
```

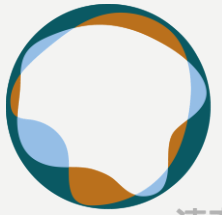
```
! ( 5 <= 5)
```

```
! ( 3/2 != I)
```



ORDER OF OPERATIONS

Precedence	Operator	Operation
highest	**	exponentiation
	-	negation
	*, /, %	multiplication, division, modulo
	+, -	adding, subtraction
	==,!=,<,>,<=,>=	comparisons(relationals)
	!	logical not
	&&	logical and
	||	logical or
lowest	=	assignment



ORDER OF OPERATIONS

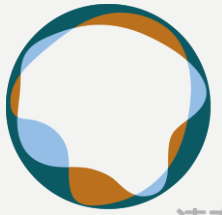
Example:

```
boolean a = 5 * 7 >= 3 + 5 * (7 - 1);
int num = 6;
boolean b = 2<= num && num>=10;
boolean c = 2<= num || num>=10;
int x = 2;
int y = 4;
int z = 5;
boolean d = z > 2 || x > 3 && y < 3 ;
```

ORDER OF OPERATIONS

Practice

```
boolean a =(4 > 3) && (2 != 0);
boolean b = (!(1 == 0)) || (0 > 9);
boolean c = (1 > (7 < 0)) && ((9 == 0) || (1 < 2));
```



ORDER OF OPERATIONS

Practice

```
int x = 42;
int y = 17;
int z = 25;

boolean a = y < x && y <= z;
boolean b = x % 2 == y % 2 || x % 2 == z % 2;
boolean c = x <= y + z && x >= y + z;
boolean d = !(x < y && x < z);
boolean e = (x + y) % 2 == 0 || !((z - y) % 2 == 0);
```

ORDER OF OPERATIONS

Practice

Assume that a and b are integers. The boolean expression (a <= b) ||!(a * b > 0)

will always evaluate to true given that

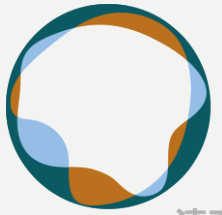
(A) a = b

(B) a > b

(C) a > 0 and b < 0

(D) a = 0 or b = 0

(E) a > b and b < 0



ORDER OF OPERATIONS

Practice

Given that a , b , and c are integers, consider the boolean expression

$$(a < b) \;||\; !((c == a * b) \;\&\&\; (c < a))$$

Which of the following will guarantee that the expression is true ?

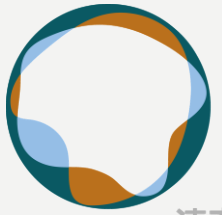
(A) c < a is false .

(B) c < a is true .

(C) a < b is false .

(D) c == a * b is true .

(E) c == a * b is true , and c < a is true .



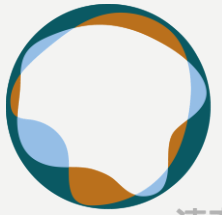
UPGRADE- PASSWORD

The method passwordStrength() should

- print ”Password Strength: Good” if the Password is greater than 8 characters and the password starts with a #

  (Tip: str1.startsWith(str2) return true, if starts with str2)

- print “Password Strength: Bad – add more characters”, if the password is 8 or less characters.



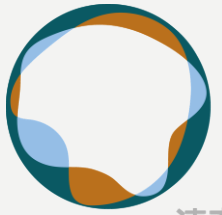
PASSWORD 2

The method passwordStrength() should

- print ”Password Strength: Good” if the Password is greater than 8 characters and the password starts with a #

  (Tip: str1.startsWith(str2) return true, if starts with str2)

- print “Password Strength: Bad – add more characters”, if the password is 8 or less characters.



PASSWORD 2

The method passwordStrength() should

- print ”Password Strength: Good” if the Password is greater than 8 characters and the password starts with a #

  (Tip: str1.startsWith(str2) return true, if starts with str2)

- print “Password Strength: Bad – add more characters”, if the password is 8 or less characters.

```
public void passwordStrength(String password)
{
    if(password.length() >8 && password.startsWith("#"))
        System.out.println("Password Strength: Good");
    else
        System.out.println("Password Strength: Bad - add more characters");
}
```

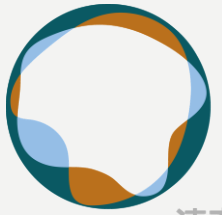
PASSWORD 3

What if we want to leave different message to the user depending on the condition they were missing?

For example:

- If the password is 8 character long, but doesn't start with a #, we want to print: "You need to include a # at the start of your password!"

- If the password is less than 8 character, we want to print: "Make sure your password is 8 characters long!"



PASSWORD 3

For example:

- If the password is 8 character long, but doesn't start with a #, we want to print: "You need to include a # at the start of your password!"
- If the password is less than 8 character, we want to print: "Make sure your password is 8 characters long!"

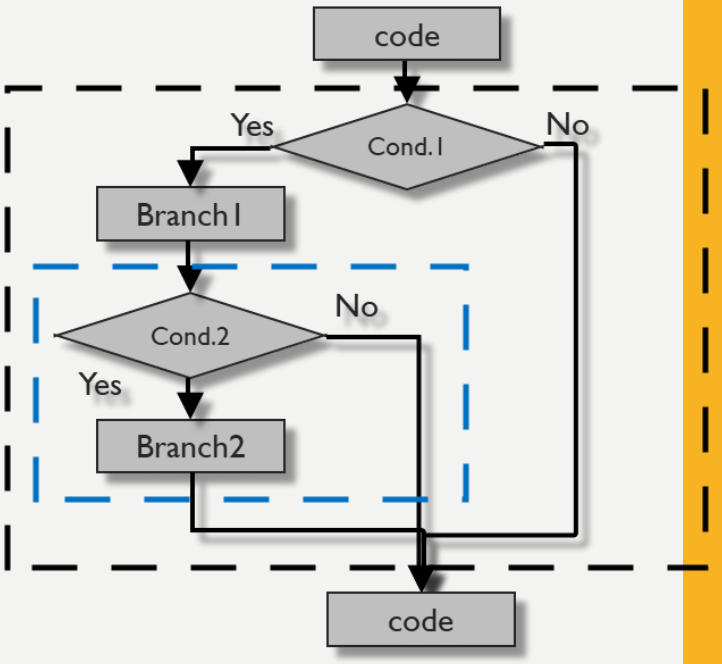
```
public void passwordStrength(String password)
{
    if(password.length() >8){
        if(password.startsWith("#"))
            System.out.println("Password Strength: Good");
    }
    else {
        System.out.println("Make sure your password is 8 characters long!");
    }
}
```

NESTEING IF STATEMENTS

NESTING IF STATEMENTS

Syntax:

```
if(condition 1 is true){
        … // statement or a block of statements
        if (condition 2 is true){
        … // statement or a block of statements
        }
}
else{
        … // statement or a block of statements
}
… //next statement
```



NESTING IF STATEMENTS

Practices

Create a method isHealthy that return true if the user is healthy. A user is healthy if their temperature is lower than 37° , and greater than 35.5.

```
public boolean isHealthy(double temp){
    if(temp > 35.5 % && temp < 37)
        return true;
    else:
        return false;
}
```