

**JavaScript 1**  
**alert("Lektion 10");**

Utbildare: Mahmud Al Hakim

**NACKADEMIN**

# Lektionstillfällets mål

- **Mål med lektionen**
  - Arbeta med selektioner
  - Strukturdiagram (Flowcharts)
  - if–satsen
  - Jämförelseoperatorer
  - Logiska operatorer
  - Switch–satsen
- **Arbetsmetod**
  - Teori och praktik varvas under lektionen

# Kort summering av föregående lektion

## Föregående lektion:

- Vi har jobbat med ett antal inbyggda globala JS-Objekt
  1. String
  2. Number
  3. Math
  4. Date

Tips: Standard built-in objects

[https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\\_Objects](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects)

# Repetition: String Methods med Bob Tabor

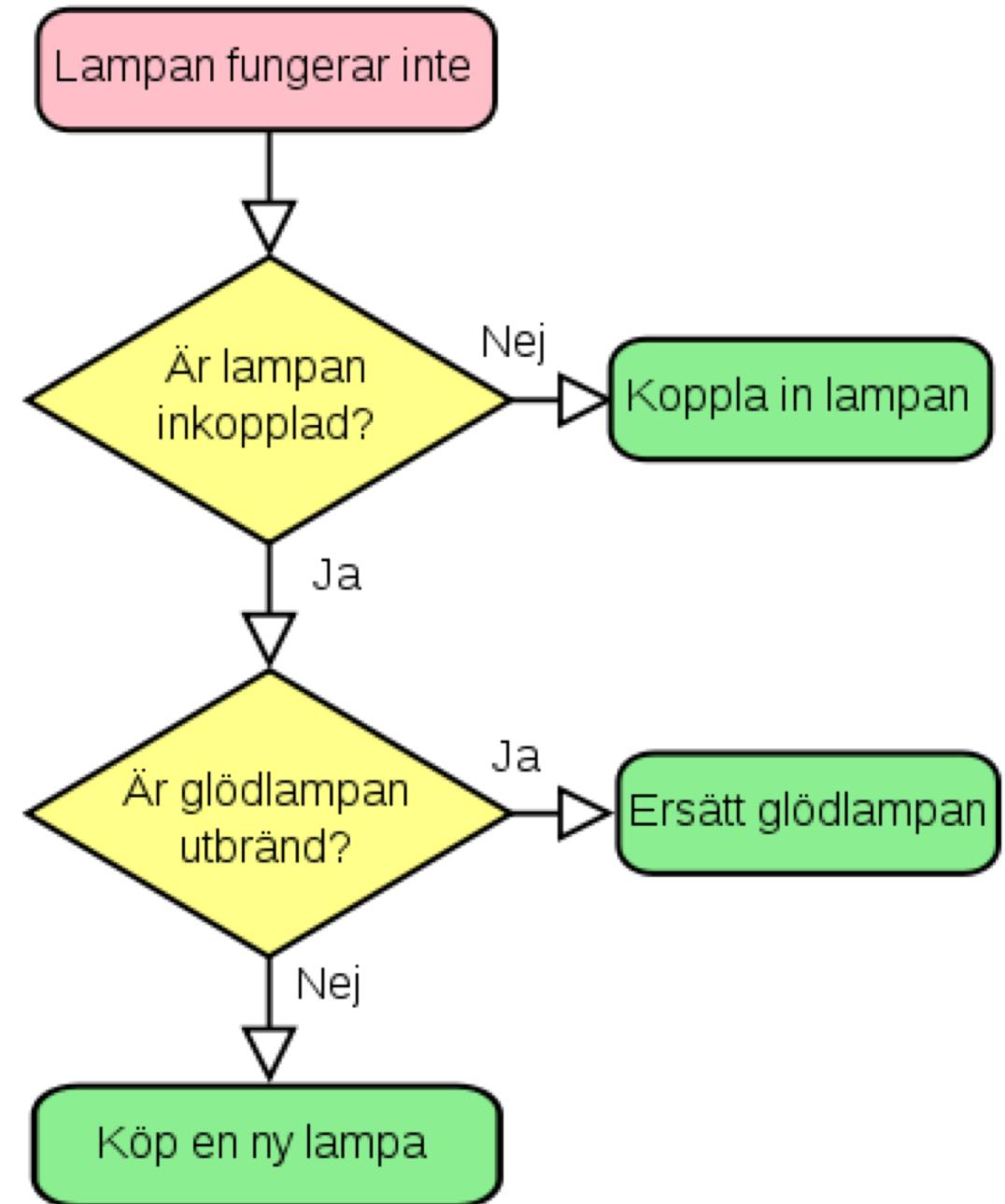
[https://mva.microsoft.com/en-us/training-courses/javascript-fundamentals-for-absolute-beginners-14194?l=1cJ3SPVxE\\_8300115881](https://mva.microsoft.com/en-us/training-courses/javascript-fundamentals-for-absolute-beginners-14194?l=1cJ3SPVxE_8300115881)

The screenshot shows a video player interface. The main content area has a blue background with white text. On the left, it says "String Methods". Below that, "Bob Tabor" and a link "https://devu.com" are listed. In the bottom right corner of the content area, there is a small illustration of a person standing on a yellow sphere, surrounded by icons representing video, audio, and other media. The bottom of the screen features a dark navigation bar with a play button, a timestamp "0:01 / 8:48", and several control icons.



# Vad är en algoritm?

- En algoritm är en beskrivning av hur ett visst problem shall lösas.
- Algoritmer kan uttryckas i **pseudokod** (naturligt språk) eller **flödesschema/flowchart** (strukturdiagram).
- Programmering går ut på att beskriva algoritmer så att de kan exekveras av maskiner och dessutom förstås av mänsklor.



# Strukturdiagram/Flödesschema (Flowcharts)

- Ett flödesschema är en grafisk beskrivning av en algoritm eller en process.
- Dess grafiska struktur kan bestå av ovaler, rektanglar och romber som binds samman av pilar för att visa ordningen i flödesschemat.
- Innehållet i de grafiska figurerna består av korta beskrivningar, som till exempel instruktioner och villkor.
- Flödesscheman används till att analysera, designa, dokumentera eller hantera en process eller ett program inom olika områden.
- <https://sv.wikipedia.org/wiki/Fl%C3%B6desschema>

# Pseudokod – Övning

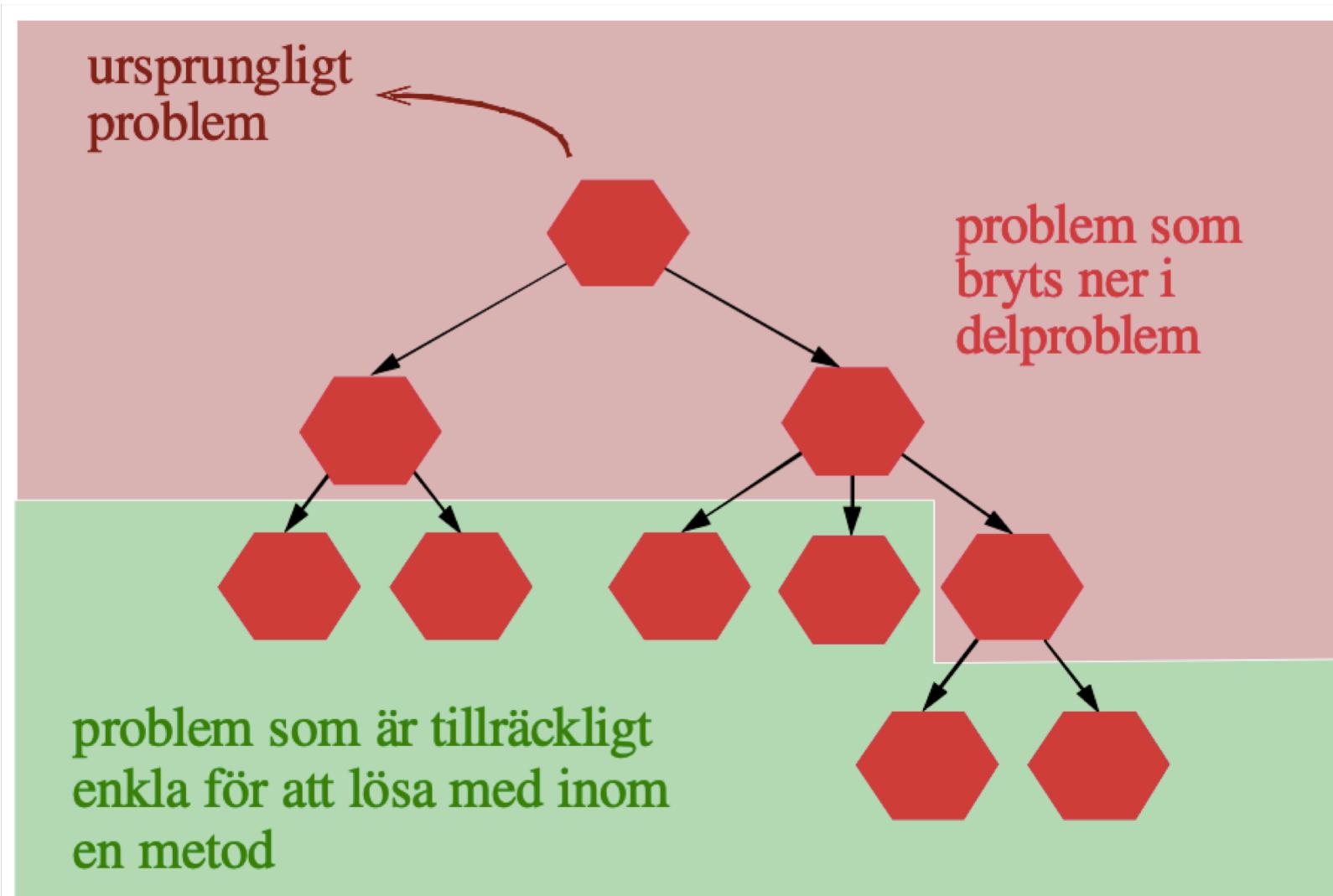
- **Beskriv i pseudokod en algoritm som läser in 100 tal och som beräknar och skriver ut medelvärdet av de inlästa talen.**
1. Använd en variabel, *summa*, som sätts till 0 och en räknare som sätts till 1.
  2. Upprepa följande så länge som räknaren är mindre än eller lika med 100:
    1. Läs in ett tal.
    2. Öka summa med det inlästa talet.
    3. Öka räknaren med ett.
  3. Beräkna uttrycket  $\text{summa}/100$  och skriv ut det.

while a number is greater than zero,  
do the following:  
Add current number to the running total.  
once the number is greater than zero  
Pseudocode

# Top-Down Design

- Top-down design innebär att vi betraktar det ursprungliga problemet på en hög abstraktionsnivå och bryter ner det ursprungliga problemet i ett antal delproblem.
- Varje delproblem betraktas sedan som ett separat problem, varvid fler aspekter på problemet beaktas, dvs vi arbetar med problemet på en lägre abstraktionsnivå än vi gjorde med det ursprungliga problemet.
- Om nödvändigt bryts delproblemen ner i mindre och mer detaljerade delproblem.
- Denna process upprepas till man har delproblem som är enkla att överblicka och lösa.

Med top-down design blir det möjligt att lösa det ursprungliga problemet steg för steg istället för att direkt göra en fullständig lösning



# Kontrollstrukturer

- En algoritm består av ett antal operationer och anvisningar om i vilken ordning operationerna skall utföras.
- En algoritms komponenter kallas för kontrollstrukturer.

## **1. Sekvens**

En följd av satser som utförs en i taget i den ordning de skrivits.

## **2. Selektion (val)**

Utför olika satser i olika situationer.

Innebär att ett alternativ bland två eller flera skall väljas.

## **3. Iteration (repetition, upprepning)**

Satser som upprepas ett visst antal .

# Selektioner

- Precis som namnet antyder handlar det om val.
- Programmet kan ta olika vägar beroende på olika villkor.
- Vanliga selektionssatser
  - 1) if-satsen och if...else
  - 2) switch-satsen

# if-satsen

```
if(villkor){  
    // En eller flera satser  
}
```

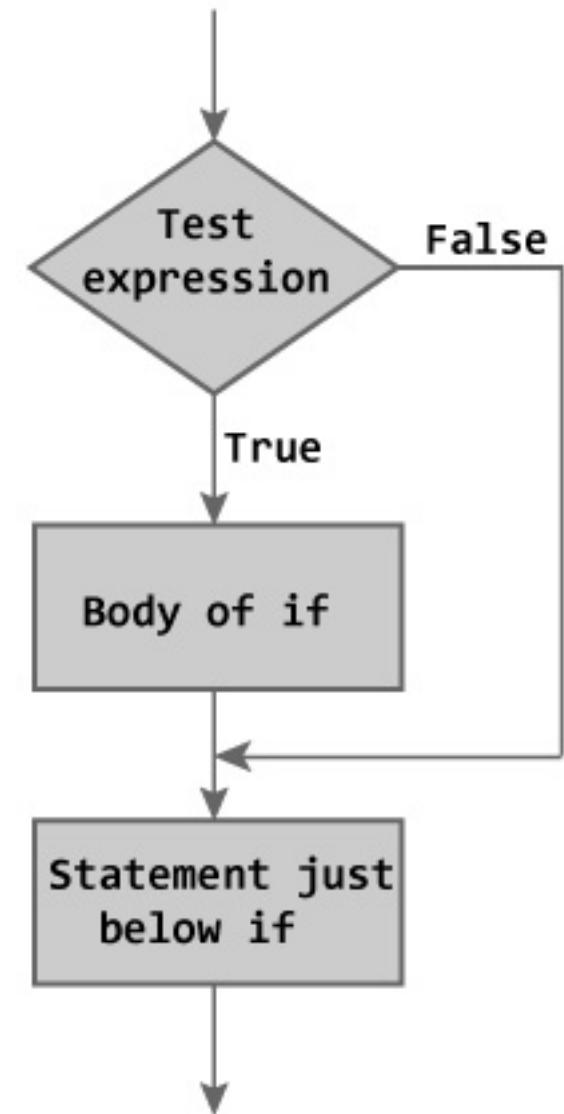


Figure: Flowchart of if Statement

KEYWORD

CONDITION

OPENING  
CURLY BRACE

```
if (score >= 50) {  
    congratulate();  
}
```

CODE TO EXECUTE IF VALUE IS TRUE

CLOSING  
CURLY BRACE

# if-satsen

```
let score = 75;  
  
if (score >= 50) {  
    congratulate();  
}  
  
function congratulate(){  
    let msg ='Congratulations! '  
    msg += 'Proceed to the next round.';  
    console.log(msg);  
}
```

## Jämförelseoperatorer (Comparison Operators)

---

<b>Operator</b>	<b>Description</b>
<code>==</code>	equal to
<code>===</code>	equal value and equal type
<code>!=</code>	not equal
<code>!==</code>	not equal value or not equal type
<code>&gt;</code>	greater than
<code>&lt;</code>	less than
<code>&gt;=</code>	greater than or equal to
<code>&lt;=</code>	less than or equal to
<code>?</code>	ternary operator

== eller ==

```
let x = '5';
```

```
let y = 5;
```

```
if (x === 5) // True! Farligt  
  console.log(x + 2); // 52 !!!
```

```
if (y === 5) // True! Säkrare  
  console.log(y + 2); // 7
```

```
if (score >= 50) {  
    congratulate();  
}  
else {  
    encourage();  
}
```

CODE TO EXECUTE IF VALUE IS TRUE

CODE TO EXECUTE IF VALUE IS FALSE

```
if...else
```

```
let score = 40;
```

```
if (score >= 50) {  
    congratulate();  
}
```

```
else{  
    encourage();  
}
```

```
function encourage(){  
    let msg ='Have another go!';  
    console.log(msg);  
}
```

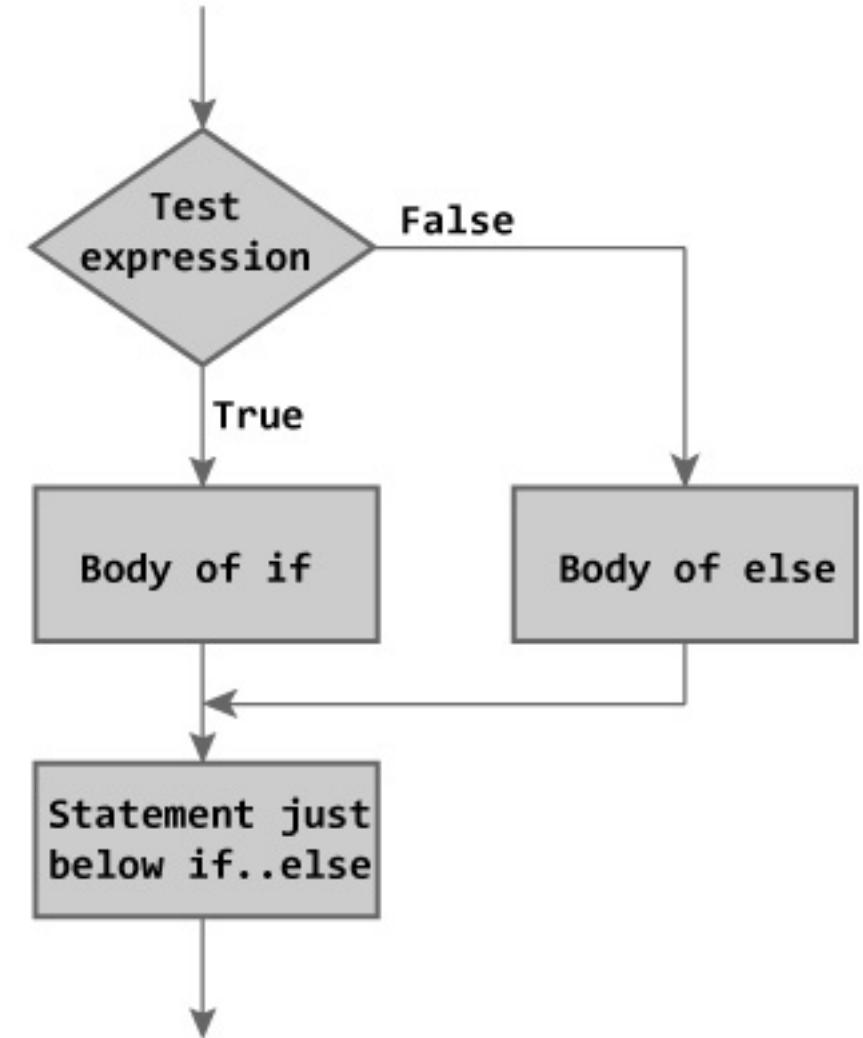


Figure: Flowchart of if...else Statement

if...else if

```
let hour = new Date().getHours();

if (hour < 12) {
    greeting = "God morgen";
}
else if (hour >= 12) {
    greeting = "God dag";
}

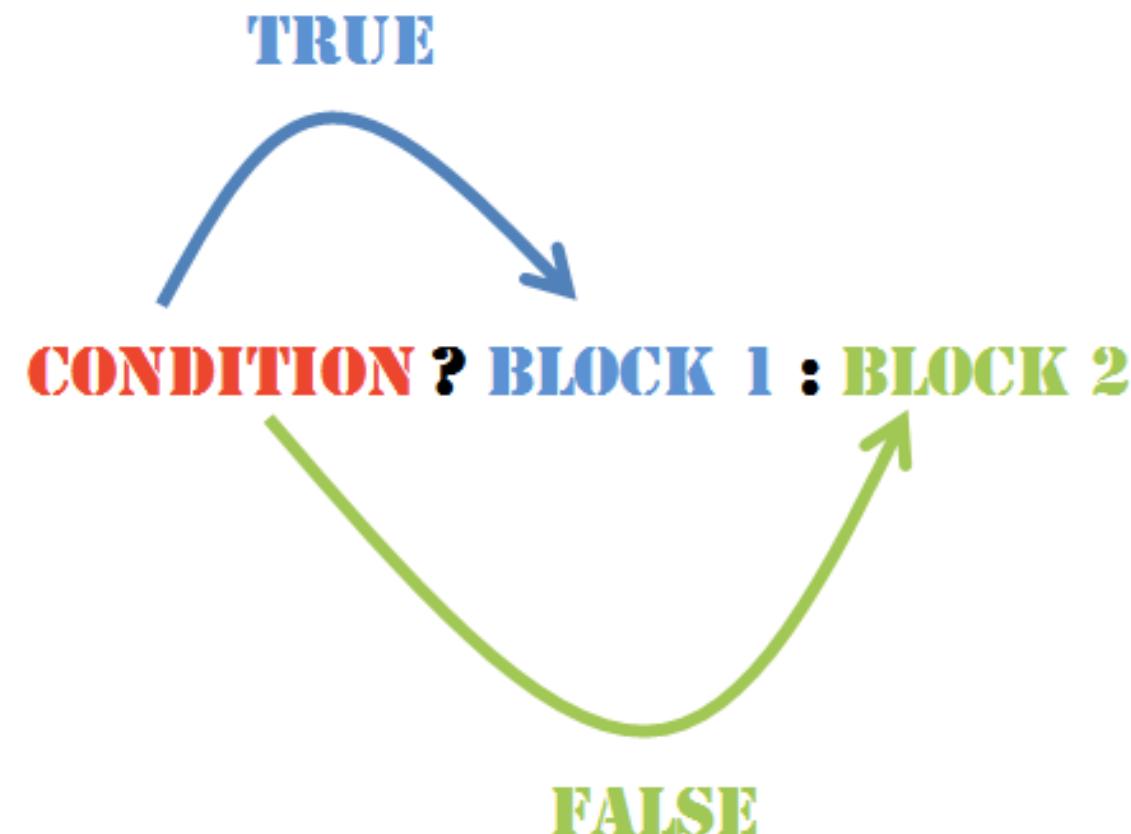
console.log(greeting);
```

# if...else if...else

```
let hour = new Date().getHours();

if (hour < 11) {
    greeting = "God morgon";
}
else if (hour < 20) {
    greeting = "God dag";
}
else{
    greeting = "God kväll";
}
console.log(greeting);
```

# Ternary-operatorn (Conditional operator)



# Ternary-operatorn – Exempel 1

```
function getFee(isMember) {  
    return (isMember ? "20kr" : "100kr");  
}  
  
console.log(getFee(true)); // 20kr  
console.log(getFee(false)); // 100kr
```

# Ternary-operatorn – Exempel 2

# Logiska operatorer (Logical Operators)

---

<b>Operator</b>	<b>Description</b>
-----------------	--------------------

&&	and
----	-----

	or
--	----

!	not
---	-----

# Sanningstabell (The truth table)

A	B	A AND B	A OR B	NOT A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

# Logiska operatorer – Exempel

```
function promptAge() {  
  
    let age = prompt("Hur gammal är du?");  
    if (age == null || age == "")  
        return;  
  
    let voteable = (age < 18) ? "Du får inte rösta!"  
                            : "Ok. Du får rösta";  
    alert(voteable);  
}  
  
promptAge();
```



Truthy  
Falsy

# Truthy & Falsy

- Följande värden är alltid falska (falsy values)
  - 1) false
  - 2) 0 (noll)
  - 3) " eller "" (tom string)
  - 4) null
  - 5) undefined
  - 6) NaN (t.ex. vid division med noll)
- Allt annat är sanna värden (truthy)  
t.ex. strängen "0" och en tom array [ ]
- Tips

<https://www.sitepoint.com/javascript-truthy-falsy/>

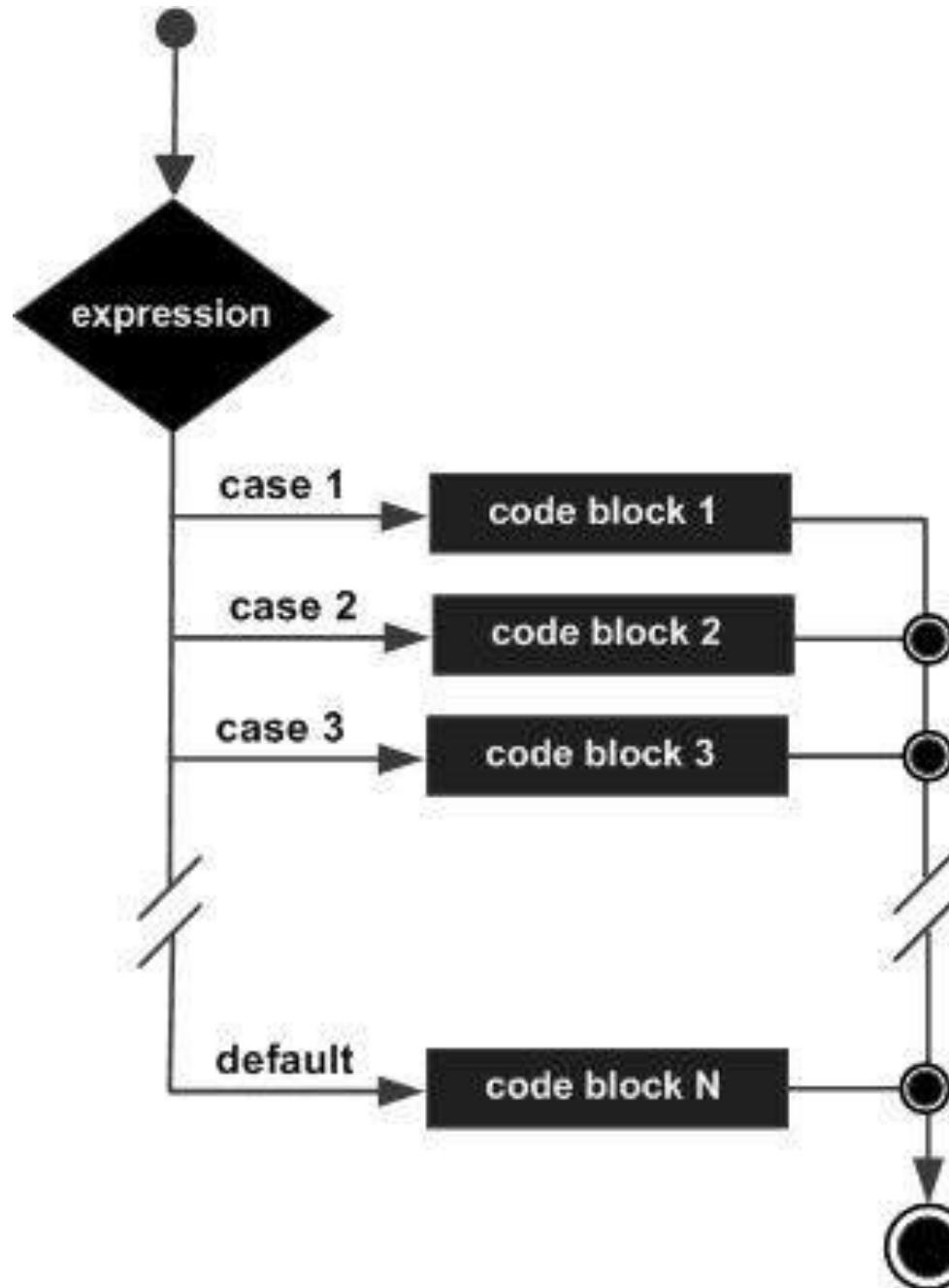
# Truthy & Falsy

```
if (false || 0 || null || undefined || '' || "" || NaN)  
  console.log("Vi kommer aldrig hit!");
```

```
if ([] || {})  
  console.log('En tom array eller ett tomt objekt!');
```

```
if ('0' || 'false')  
  console.log("Strängen '0' eller strängen 'false'");
```

# Switch-satsen



# Switch-satsen – Exempel 1

```
function getDay(){
    let day;
    switch (new Date().getDay()) {
        case 0: day = "söndag"; break;
        case 1: day = "måndag"; break;
        case 2: day = "tisdag"; break;
        case 3: day = "onsdag"; break;
        case 4: day = "torsdag"; break;
        case 5: day = "fredag"; break;
        case 6: day = "lördag"; break;
    }
    return day;
}
console.log('Det är ' + getDay() + ' idag!');
```

# Switch-satsen – Exempel 2

```
let personnummer = prompt("Personnummer (ååmmdd-xxxx)?");

switch (personnummer.charAt(9)) {
    case '0': case '2': case '4': case '6': case '8':
        alert("Kvinna");
        break;
    case '1': case '3': case '5': case '7': case '9':
        alert("Man");
        break;
    default:
        alert("Felaktigt personnummer");
}
```

# Övning 1

Skapa ett program som läser in (via två prompt) antalet minuter man i genomsnitt ringer per månad samt kostnaden per minut.

Om man ringer för minst 1000kr per månad så har man rätt till 10% rabatt.

Som resultat skall programmet visa en dialogruta (alert) där det anges hur stor den beräknade kostnaden per månad blir samt hur mycket rabatt man får.

## Övning 2

På ett gym kan man antingen köpa ett årskort eller köpa en biljett vid varje besök.

Skapa ett program som läser in (via en prompt) priset för ett årskort, priset för en biljett samt antalet gånger man planerar att besöka gymmet under ett år.

Därefter skall programmet visa (via en alert) om det lönar sig att köpa årskort eller inte.

## Övning 3

På ett matteprov kunde man få högst 50 poäng.

Gränsen för betyget E var 25 poäng och för betygen D, C, B och A 30, 35, 40 respektive 45 poäng.

Skriv ett program som läser in (via en prompt) poängen för en elev och som visar vilket betyg hon eller han fick på provet.

# Övningar

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_comparisons1](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_comparisons1)

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_comparisons2](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_comparisons2)

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_comparisons3](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_comparisons3)

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_comparisons4](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_comparisons4)

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_conditions1](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_conditions1)

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_conditions2](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_conditions2)

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_switch1](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_switch1)

[https://www.w3schools.com/js/exercise\\_js.asp?filename=exercise\\_js\\_switch2](https://www.w3schools.com/js/exercise_js.asp?filename=exercise_js_switch2)

# Summering av dagens lektion

- Vi har idag jobbat med selektioner i JavaScript.
- Reflektioner kring dagens lektion?
  - Vad tar du med dig från dagens lektion?
  - Finns det något som var extra svårt att förstå?
  - Finns det något som vi behöver repetera?
  - Hur upplevde du dagens arbetsmetoder?

# Framåtblick inför nästa lektion

- Läs: Bok 2, sid. 145-169
- Under nästa lektion kommer vi att jobba med **iterationer**.