

# ESERCIZI

- **Operatore XOR**
  - L'operatore booleano binario  $\text{XOR}(a,b)$  calcola:
    - true, quando solamente uno tra a e b è true,
    - false, in tutti gli altri casi.
  - Si scriva un programma che definisca una funzione XOR che, avendo come parametri due valori booleani a e b, restituisca il risultato di  $\text{XOR}(a,b)$ . Si completi il programma invocando la funzione precedentemente definita per tutte le possibili combinazioni di valori booleani per i suoi parametri, e si stampino i risultati ottenuti.
- Si definisca una funzione **XOR\_senza\_if**, con funzionamento simile alla funzione XOR (stesso input e output) ma che non faccia uso dell'istruzione if.



# OPERATORE XOR – VERSIONE SEMPLIFICATA

// iniziamo da una funzione più semplice: restituiamo true solo se a è vero  
e b è falso

```
function XOR_semplice(a,b){  
  if(a==true)  
    if(b==false)  
      return true;  
    else return false;  
  else return false;  
}
```

```
function XOR_semplice2(a,b){  
  if (a==true && b==false)  
    return true;  
  else return false;  
}
```



# OPERATORE XOR

```
function XOR(a,b){  
  if ((a==true && b==false) || (a==false && b==true))  
    return true;  
  else return false;  
}
```

```
console.log(XOR(true,true)); // false  
console.log(XOR(true,false)); // true  
console.log(XOR(false,true)); // true  
console.log(XOR(false,false)); // false
```

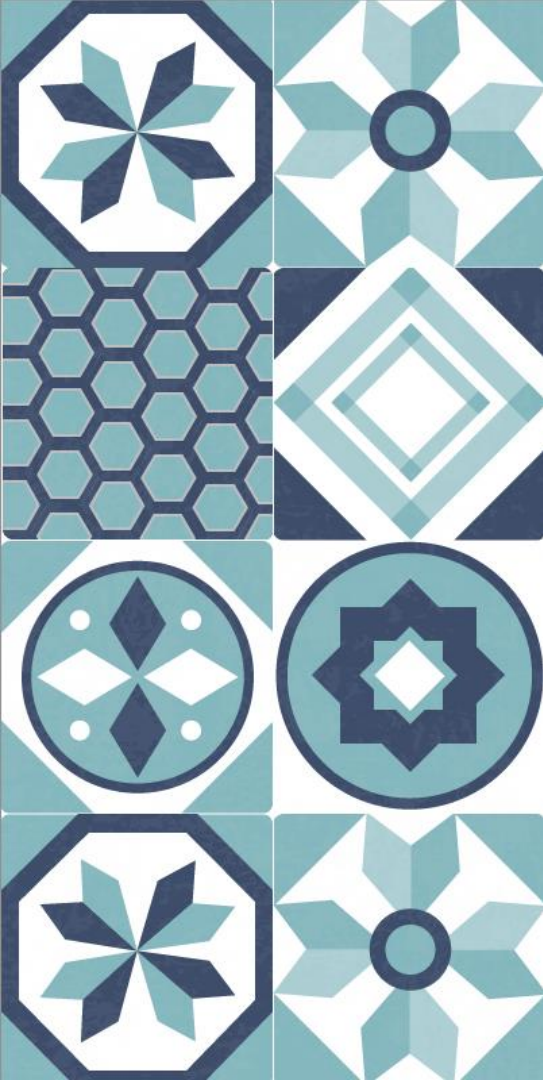


# OPERATORE XOR (2)

```
function XOR2(a,b){  
  if ((a && !b) || (!a && b))  
    return true;  
  else return false;  
}
```

```
function XOR3(a,b){  
  if (a != b)  
    return true;  
  else return false;  
}
```





# FONDAMENTI DI INFORMATICA

Alma Artis  
Francesca Pratesi (ISTI, CNR)

Visibilità delle variabili



# **VISIBILITÀ DELLE VARIABILI**

# AMBIENTE E VISIBILITÀ DELLE VARIABILI

- Un ambiente è costituito da un insieme di variabili e di funzioni
  - in un ambiente non possono esserci due variabili o due funzioni con lo stesso nome (identificatore)
- Un ambiente è modificato:
  - dalle dichiarazioni: aggiungono una variabile o una funzione all'ambiente
  - dai comandi che modificano i valori delle variabili
- Ogni variabile è visibile nei punti del programma in cui fa parte dell'ambiente
  - Anche i blocchi racchiusi tra { } definiscono ambienti diversi



# VARIABILI GLOBALI E LOCALI

- Una variabile globale è visibile in tutto il programma
  - A meno che non vengano mascherate
- Una variabile locale è visibile solo in alcuni punti del programma
- Per i nostri scopi un parametro formale si comporta come una variabile locale
  - NB: il parametro formale non deve essere dichiarato dalla keyword var





# VARIABILI GLOBALI E LOCALI - ESEMPIO


```
function funzione_prova(numero){  
    var nome = 'Francesca';  
    var testo = '';  
    var b = 5;  
    if (numero>0) testo = nome;  
    else testo = ''+b;  
    return testo;  
}
```

```
var a, b, x;  
a = 12;  
b = 45;  
x = 2**3;  
console.log(funzione_prova(a));
```



# VARIABILI GLOBALI E LOCALI - ESEMPIO

```
function funzione_prova(numero){  
    var nome = 'Francesca';  
    var testo = '';  
    var b = 5;  
    if (numero>0) testo = nome;  
    else testo = ''+b;  
    return testo;  
}
```

 variabili globali

```
var a, b, x;  
a = 12;  
b = 45;  
x = 2**3;  
console.log(funzione_prova(a));
```

# VARIABILI GLOBALI E LOCALI - ESEMPIO

```
function funzione_prova(numero){  
  var nome ← 'Francesca';  
  var testo ← '';  
  var b = 5;  
  if (numero>0) testo = nome;  
  else testo = ''+b;  
  return testo;  
}  
  
var a, b, x;  
a = 12;  
b = 45;  
x = 2**3;  
console.log(funzione_prova(a));
```

variabili locali

variabili globali

# SHADOWING O NAME MASKING

- Una variabile globale può essere nascosta da una variabile locale che ha lo stesso identificatore
  - una variabile nascosta non è accessibile



# NAME MASKING - ESEMPIO

```
function funzione1(){  
  var x, y;  
  x = 12;  
  y = 25;  
}  
function funzione2(){  
  var x, a;  
  x = 32;  
  a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;
```



# NAME MASKING - ESEMPIO

```
function funzione1(){  
  var x, y;  
  x = 12;  
  y = 25;  
}  
function funzione2(){  
  var x, a;  
  x = 32;  
  a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;
```

In funzione1 sono accessibili le variabili:

- locali: x e y
- globali: a e b

Le variabili globali x e y non sono visibili!



# NAME MASKING - ESEMPIO

```
function funzione1(){  
  var x, y;  
  x = 12;  
  y = 25;  
}  
function funzione2(){  
  var x, a;  
  x = 32;  
  a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;
```

In funzione2 sono accessibili le variabili:

- locali: x e a
- globali: b e y

Le variabili globali a e x non sono visibili!



# NAME MASKING - ESEMPIO

```
function funzione1(){  
  var x, y;  
  x = 12;  
  y = 25;  
}  
function funzione2(){  
  var x, a;  
  x = 32;  
  a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;
```

Nel corpo del programma sono accessibili le variabili:

- globali: a, b, x e y

Le variabili locali a funzione1 e funzione2 non sono visibili!





# LO STATO E LE VARIABILI DI AMBIENTE

- Come già detto, lo stato viene generato a partire dalle funzioni e dalle variabili accessibili in un particolare punto del programma



# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){  
    var x, y;  
    x = 12;  
    y = 25;  
}  
function funzione2(){  
    var x, a;  
    x = 32;  
    a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;  
funzione1();
```



# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
    var x, y;
    x = 12;
    y = 25;
}
function funzione2(x){
    var a;
    x = 32;
    a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
funzione1();
```

← `{{funzione1, function(){...}}}`

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){  
    var x, y;  
    x = 12;  
    y = 25;  
}  
function funzione2(x){  
    var a;  
    x = 32;  
    a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;  
funzione1();
```

← {(funzione1, function(){...}), (funzione2, function(x){...})}

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
    var x, y;
    x = 12;
    y = 25;
}
function funzione2(x){
    var a;
    x = 32;
    a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
funzione1();
```


{(a,undefined), (b,undefined), (x,undefined), (y,undefined), (funzione1, function(){...}), (funzione2, function(x){...})}

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){  
    var x, y;  
    x = 12;  
    y = 25;  
}  
function funzione2(x){  
    var a;  
    x = 32;  
    a = 42;  
}
```

// corpo del programma

```
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;  
funzione1();
```

 `{(a,12), (b,undefined), (x,undefined), (y,undefined), (funzione1, function(){...}), (funzione2, function(x){...})}`

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){  
    var x, y;  
    x = 12;  
    y = 25;  
}  
function funzione2(x){  
    var a;  
    x = 32;  
    a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;  
funzione1();
```


 `{{(a,12), (b,45), (x,undefined), (y,undefined), (funzione1, function(){...}), (funzione2, function(x){...})}}`

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
    var x, y;
    x = 12;
    y = 25;
}
function funzione2(x){
    var a;
    x = 32;
    a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
function1();
```

Environment state after execution of the first function call:

{(a,12), (b,45), (x,42), (y,undefined), (funzione1, function(){...}), (funzione2, function(x){...})}





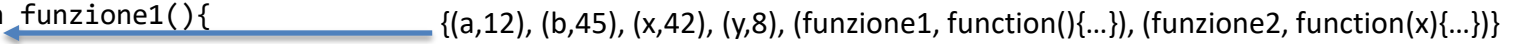
# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
    var x, y;
    x = 12;
    y = 25;
}
function funzione2(x){
    var a;
    x = 32;
    a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
function1();
```

$\{(a,12), (b,45), (x,42), (y,8), (funzione1, function()\{...\}), (funzione2, function(x)\{...\})\}$

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
    var x, y;
    x = 12;
    y = 25;
}
function funzione2(x){
    var a;
    x = 32;
    a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
funzione1();
```



# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){  
  var x, y;                                     {(x,undefined), (y,undefined), (a,12), (b,45), (funzione1, function(x){...}), (funzione2,  
  x = 12;                                       function(){...})}  
  y = 25;  
}  
function funzione2(x){  
  var a;  
  x = 32;  
  a = 42;  
}  
// corpo del programma  
var a, b, x, y;  
a = 12;  
b = 45;  
x = 42;  
y = 2**3;  
funzione1();
```

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
  var x, y;
  x = 12;
  y = 25;
}
function funzione2(x){
  var a;
  x = 32;
  a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
function1();
```

{(x,12), (y,undefined), (a,12), (b,45), (funzione1, function(){...}), (funzione2, function(x){...})}

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
  var x, y;
  x = 12;
  y = 25;
}
function funzione2(x){
  var a;
  x = 32;
  a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
function1();
```

Diagram illustrating the environment state after the execution of the provided JavaScript code. A blue arrow points from the assignment `y = 25;` in the `funzione1` function to the entry `(y,25)` in the environment object.

Environment state (Environment Object):

```
{(x,12), (y,25), (a,12), (b,45), (funzione1, function(){...}), (funzione2, function(x){...})}
```

# LO STATO E LE VARIABILI DI AMBIENTE - ESEMPIO

```
function funzione1(){
    var x, y;
    x = 12;
    y = 25;
}
function funzione2(x){
    var a;
    x = 32;
    a = 42;
}
// corpo del programma
var a, b, x, y;
a = 12;
b = 45;
x = 42;
y = 2**3;
function1();
```

← {(a,12), (b,45), (x,42), (y,8), (funzione1, function(){...}), (funzione2, function(x){...})}