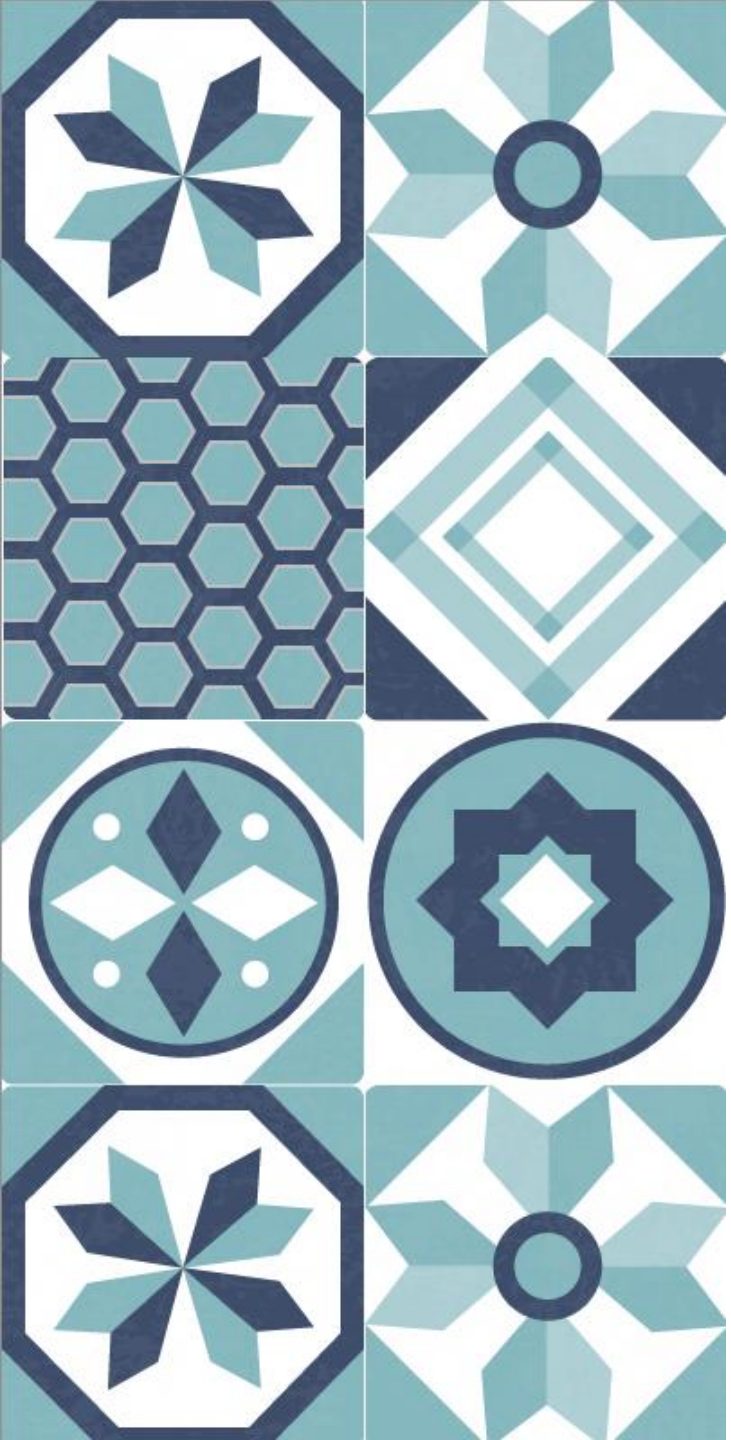


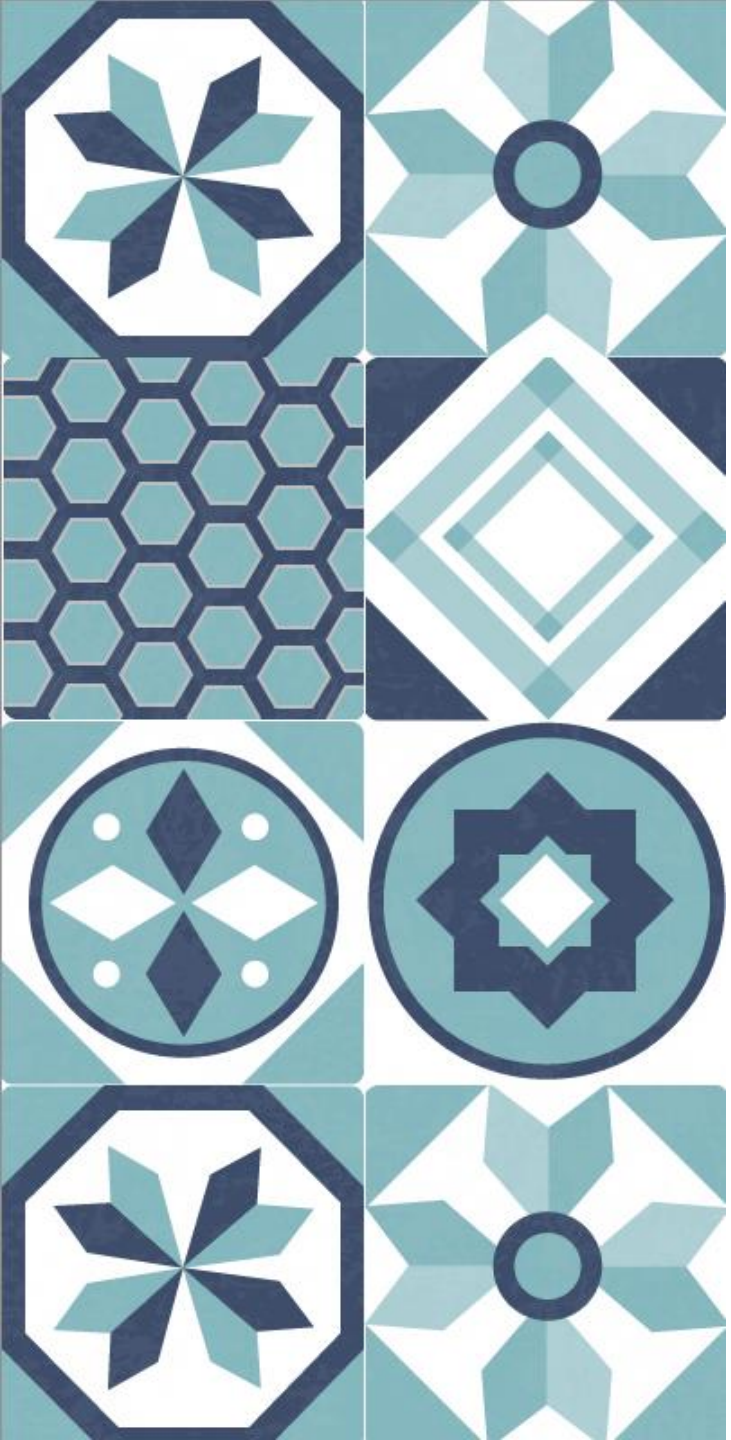
FONDAMENTI DI INFORMATICA

Alma Artis
Francesca Pratesi (ISTI, CNR)

Array - Integrazione

ESERCIZI

- Si scriva una funzione *leggiArray* che legga dall'input *n* numeri (*n* è un parametro) e li salvi in un array, che verrà restituito.
- Si scriva una seconda funzione *elementiPari* che prenda in input l'array salvato al punto precedente e calcoli *true* se tutti gli elementi pari sono in posizione dispari.
- Si scriva una terza funzione *posizioniPari* che prenda in input l'array salvato al primo punto e calcoli *true* se tutti gli elementi in posizione pari sono numeri dispari.



ARRAY DINAMICI

ARRAY DINAMICI

- In JS gli array sono strutture dinamiche: il numero degli elementi può variare durante l'esecuzione del programma
- Per aggiungere un elemento ad array è possibile assegnare un valore ad un elemento di indice successivo all'ultimo attualmente esistente

ESEMPIO

```
var L = 5, i = 0;
var vet = new Array(L); //vet ha lunghezza 5
//gli elementi del vettore vengono riempiti con le potenze di 2
for (i = 0;i < L; i++) {
    vet[i] = Math.pow(2,i);
}
vet[L] = Math.pow(2,L); //<--- vet ora ha lunghezza 6
vet[L+2] = -1; //aggiunge due elementi
console.log(vet[L+1]) // che valore viene stampato?
```

ESEMPIO

```
var L = 5, i = 0;
var vet = new Array(L);

for (i = 0;i < L; i++) {
    vet[i] = Math.pow(2,i);
}
vet[L] = Math.pow(2,L);
vet[L+2] = -1;
console.log(vet[L+1])
```

0	undefined
1	undefined
2	undefined
3	undefined
4	undefined

ESEMPIO

```
var L = 5, i = 0;
var vet = new Array(L);

for (i = 0;i < L; i++) {
    vet[i] = Math.pow(2,i);
}
vet[L] = Math.pow(2,L);
vet[L+2] = -1;
console.log(vet[L+1])
```

0	1
1	2
2	4
3	8
4	16

ESEMPIO

```
var L = 5, i = 0;
var vet = new Array(L);

for (i = 0;i < L; i++) {
    vet[i] = Math.pow(2,i);
}
vet[L] = Math.pow(2,L);
vet[L+2] = -1;
console.log(vet[L+1])
```

0	1
1	2
2	4
3	8
4	16
5	32

ESEMPIO

```
var L = 5, i = 0;
var vet = new Array(L);

for (i = 0;i < L; i++) {
    vet[i] = Math.pow(2,i);
}
vet[L] = Math.pow(2,L);
vet[L+2] = -1;
console.log(vet[L+1])
```

0	1
1	2
2	4
3	8
4	16
5	32
6	undefined
7	-1

STRINGHE E ARRAY

STRINGHE DI CARATTERI

- In JavaScript, le stringhe possono essere gestite come array di caratteri
- In questo caso la sintassi è la stessa vista per gli array in generale, solo che i valori degli elementi di un array stringa sono i caratteri della stringa stessa

STRINGHE COME ARRAY

- Accesso ad un elemento

```
var s = 'ciao’;
var a = s[0];
write(a); //stampa 'c'
```

- Lunghezza di una stringa

```
var l = s.length;
write(l); //stampa 4
```

MANIPOLAZIONE DI STRINGHE

- A differenza degli altri array, non è possibile modificare le stringhe
- Ogni tentativo di modificare una stringa viene ignorato
  - attenzione: non viene segnalato alcun errore

```
var s = 'ciao’;
s[0] = 'm’;
console.log(s); //stampa ciao (non miao)
```

ESEMPIO: CALCOLO DELLE FREQUENZE

- Leggere dall'input una stringa costituita da cifre decimali e stampare le frequenze delle cifre da '0' a '9'
- Come possiamo fare:
  - Usiamo un vettore *freq* di 10 elementi, per memorizzare le frequenze dei caratteri da '0' a '9'
    - *freq[0]* conta il numero di occorrenze del carattere '0'

      …
    - *freq[9]* conta il numero di occorrenze del carattere '9'
  - Usiamo un ciclo per l'acquisizione dei caratteri, in cui aggiorniamo uno degli elementi dell'array ogni volta che il carattere letto è una cifra

CALCOLO DELLE FREQUENZE

```
var i, s = prompt(‘Inserisci una stringa composta da cifre’); //legge la stringa
var freq = new Array(0,0,0,0,0,0,0,0,0,0);  //freq viene inizializzato con 10 elementi uguali a 0
for (i=0; i<s.length;i++)
     switch (s[i]) { //aggiorna l'elemento giusto in freq
          case '0': freq[0]++; break;
          case '1': freq[1]++; break;
          case '2': freq[2]++; break;
          case '3': freq[3]++; break;
          case '4': freq[4]++; break;
          case '5': freq[5]++; break;
          case '6': freq[6]++; break;
          case '7': freq[7]++; break;
          case '8': freq[8]++; break;
          case '9': freq[9]++; break;
     }
//stampa il risultato
for (i = 0; i < freq.length; i++)
     console.log("Frequenza di "+i+" "+ freq[i]);
```

ARRAY MULTIDIMENSIONALI

ARRAY DI ARRAY

- Finora abbiamo visto che:
  - Gli array sono variabili che possono contenere più valori
  - I valori contenuti in un array possono essere di tutti i tipi visti finora
    - Numeri
    - Stringhe
    - Booleani
- Ma cosa succede se un array contiene array?

ARRAY A PIÙ DIMENSIONI

- Gli array a più dimensioni sono estremamente utili per modellare entità matematiche e per modellare oggetti nello spazio
- Gli array a due dimensioni sono detti matrici
- Alcuni linguaggi di programmazione (e.g., Matlab) supportano direttamente array a due (matrici) o più dimensioni (tensori)
- In JavaScript non esistono costrutti specifici per definire array a più dimensioni, ma…
  - Gli elementi di un array possono essere di qualsiasi tipo, quindi anche strutture
  - Gli array a più dimensioni si possono ottenere come array i cui elementi sono a loro volta array, cioè **array di array**

ESEMPIO DI MATRICE

- Matrice 3x4
- `var A = [[1, 2, 3, 4],[2, 4, 6, 8],[4, 8, 12, 16]];`

		colonne			
		0	1	2	3
righe	0	**1**	**2**	**3**	**4**
	1	**2**	**4**	**6**	**8**
	2	**4**	**8**	**12**	**16**

ESEMPIO DI MATRICE (2)

- E’ possibile definire una matrice vuota
- `var A = [new Array(4),new Array(4),new Array(4)];`

		colonne			
		0	1	2	3
righe	0	1	2	3	4
	1	2	4	6	8
	2	4	8	12	16

OPERAZIONI SU MATRICI

- Si possono usare gli stessi costrutti visti per gli array a una sola dimensione, tenendo presente che ogni elemento dell'array è a sua volta un array
  - Es.: A[1][3] indica l'elemento di indice 3 dell'elemento di indice 1 di A

```
var A = [[1, 2, 3, 4],[2, 4, 6, 8],[4, 8, 12, 16]];
```

OPERAZIONI SU MATRICI

- Si possono usare gli stessi costrutti visti per gli array a una sola dimensione, tenendo presente che ogni elemento dell'array è a sua volta un array
  - Es.: A[1][3] indica l'elemento di indice 3 dell'elemento di indice 1 di A

```
var A = [[1, 2, 3, 4],[2, 4, 6, 8],[4, 8, 12, 16]];
```

OPERAZIONI SU MATRICI

- Si possono usare gli stessi costrutti visti per gli array a una sola dimensione, tenendo presente che ogni elemento dell'array è a sua volta un array
  - Es.: A[1][3] indica l'elemento di indice 3 dell'elemento di indice 1 di A

```
var A = [[1, 2, 3, 4],[2, 4, 6, 8],[4, 8, 12, 16]];
```

OPERAZIONI SU MATRICI

- Si possono usare gli stessi costrutti visti per gli array a una sola dimensione, tenendo presente che ogni elemento dell'array è a sua volta un array
  - Es.: A[1][3] indica l'elemento di indice 3 dell'elemento di indice 1 di A

```
var A = [[1, 2, 3, 4],[2, 4, 6, 8],[4, 8, 12, 16]];
```

colonne

righe	0	1	2	3
0	1	2	3	4
1	2	4	6	8
2	4	8	12	16

LA PROPRIETÀ LENGTH

```
var A = [[1, 2, 3, 4],[2, 4, 6, 8],[4, 8, 12, 16]];
```

- A.length è il numero di elementi di A
  - in questo esempio A.length vale 3
- A[1].length è il numero di elementi di A[1]
  - in questo esempio A[1].length vale 4

MANIPOLAZIONE DI MATRICI

- Per scorrere tutti gli elementi di un array multidimensionale si usano cicli annidati
  - un ciclo for esterno (ad esempio, con indice i) scandisce gli elementi di A, cioè le righe
  - un ciclo for interno (ad esempio, con indice j) scandisce gli elementi di ogni A[i]
- Attenzione ad usare nomi diversi per i contatori dei due cicli!

ESEMPIO: STAMPARE UNA MATRICE

```
var riga, colonna;
for (riga = 0;riga<A.length;riga++){
    for(colonna=0;colonna<A[riga].length;colonna++){
        console.log(A[riga][colonna]+‘\t’);
        // \t è la tabulazione, serve a distanziare gli
elementi
    }
    console.log(‘\n’); // alla fine di ogni riga vado a
capo
}
```

ESEMPIO: POPOLARE UNA MATRICE

```
var B = [new Array(4), new Array(4), new Array(4)];
var i, j;
for (i = 0; i<B.length; i++) // iteriamo sulle righe
    for (j=0; j<B[i].length; j++) // iteriamo sulle
colonne
        B[i][j] = Number(prompt('Inserisci un numero');
```

Nota: ogni riga di B può avere una lunghezza differente! Provate per esercizio.

- Scrivere una funzione con un parametro stringa s. La funzione deve restituire in output la stringa inversa di s, cioè la stringa che si ottiene leggendo s al contrario. Es. dato 'ciao' in input deve essere restituito 'oaic’
- Un palindromo è una sequenza di caratteri che, letta al contrario, rimane identica. Esempi di palindromi: 'ada', 'ahaha', 'radar', 'ossesso', 'oro', 'anna’. Scrivere una funzione che data una stringa, verifica se essa è un palindromo o meno.

ESEMPIO

- Date due matrici A e B, entrambe M x N, è possibile calcolare la loro somma A+B = C. C è una matrice MxN in cui ogni elemento è dato dalla somma tra i corrispondenti elementi nelle matrici A e B.
  - Scrivere una funzione con due parametri A e B, aventi la stessa dimensione. La funzione calcola e restituisce un nuovo array (multidimensionale) che rappresenta la matrice A+B.

ESEMPIO – SOMMA DI MATRICI

- Date due matrici A e B, entrambe M x N, è possibile calcolare la loro somma A+B = C. C è una matrice MxN in cui ogni elemento è dato dalla somma tra i corrispondenti elementi nelle matrici A e B.
  - Scrivere una funzione con due parametri A e B, aventi la stessa dimensione. La funzione calcola e restituisce un nuovo array (multidimensionale) che rappresenta la matrice A+B.

1	2	3	+	2	3	4	=	3	5	7
4	5	6		6	5	4		10	10	10

ESEMPIO – SOMMA DI MATRICI

```
function somma_matrici(A,B){
    var C = new Array(A.length);
        for(var riga=0;riga<A.length;riga++){
            C[riga]=new Array(A[riga].length);
            for(var colonna=0;colonna<A[riga].length;colonna++){
                C[riga][colonna] = A[riga][colonna] + B[riga][colonna];
            }
        }
    return C;
}
var A = [[1,2,3],[4,5,6]];
var B = [[2,3,4],[6,5,4]];
console.log(somma_matrici(A,B));
```