Corso Sistemi Operativi
AA 2018-2019
Data structures

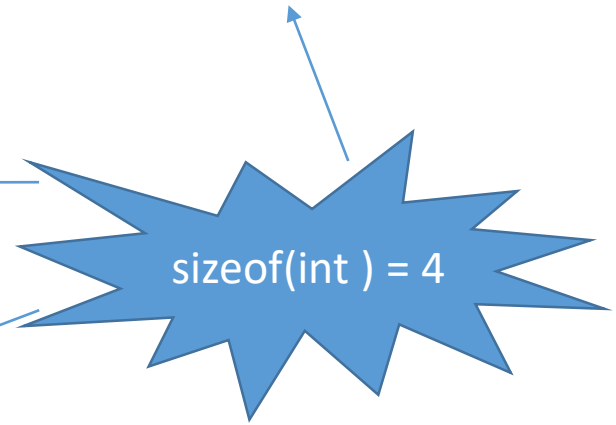
Marco Tessarotto

Array

- Un array ha una dimensione costante, stabilita al momento della creazione; tutte le celle dell'array sono adiacenti in memoria

int array[1024]; // array statico; dimensione totale in bytes = 1024***sizeof(int)**

- Si accede tramite indici: array[n], 0 <= n < 1024
- Ogni cella ha dimensione **sizeof(int)**
- Posso cambiare dimensione dell'array statico? No!

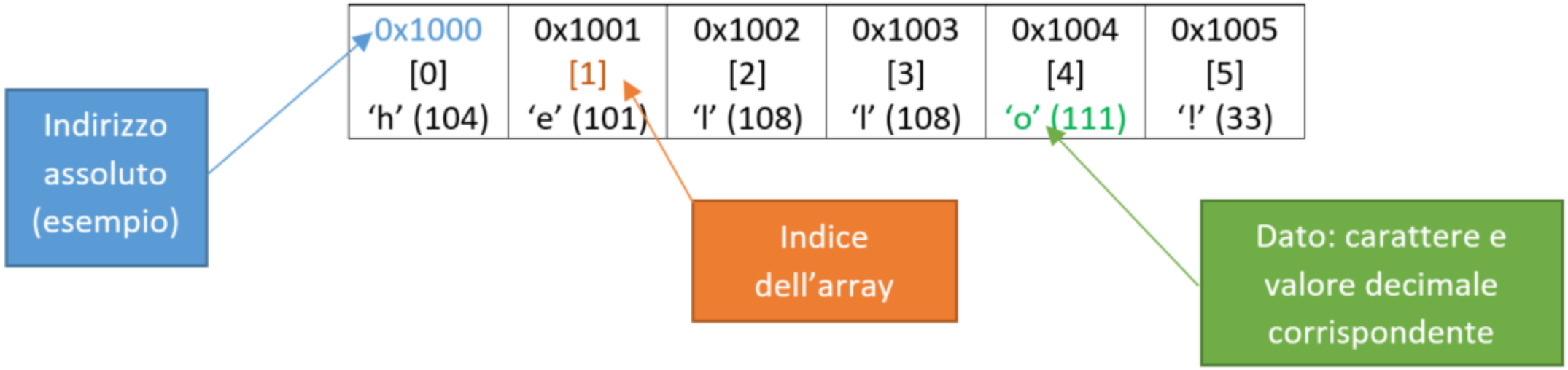


int arrayDinamico [] = calloc(1024, **sizeof(int));**

free(arrayDinamico);

Stringhe di caratteri – array statico di char

- char messaggio[] = «hello!»; // 0 terminated!
- sizeof(char) = 1
- **dimensione in bytes totale**: sizeof(messaggio) = 7 bytes
- **numero di elementi dell'array**: numero di elementi dell'array: 7
- **lunghezza stringa**: strlen(messaggio) = 6



Stringhe di caratteri

- char messaggio_con_spazio_aggiuntivo[100] = «hello!»;

- sizeof(char) = 1
- sizeof(messaggio_con_spazio_aggiuntivo) = 100 bytes
- numero di elementi dell'array: 100
- strlen(messaggio_con_spazio_aggiuntivo) = 6

- messaggio_con_spazio_aggiuntivo può contenere stringhe di lunghezza da 0 a 99 caratteri.

Array statico di int

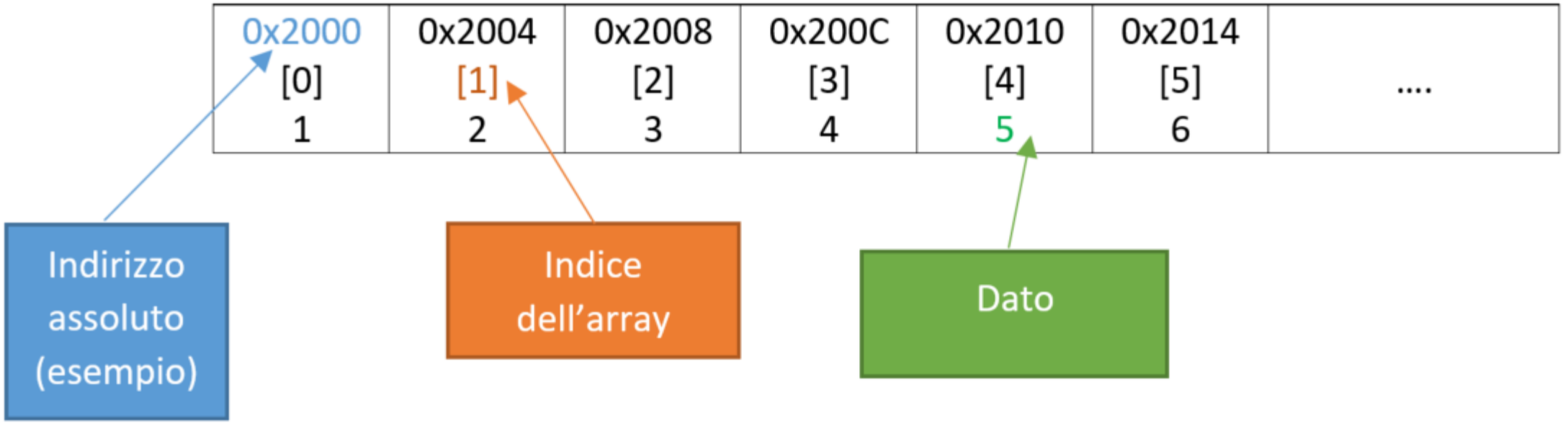
// inizializza i primi elementi ai valori specificati, gli altri a zero

int my_array[100] = { 1,2,3,4 };

sizeof(int) = 4 ← ogni cella ha dimensione pari a 4 bytes

dimensione in bytes totale: sizeof(my_array) = 400 bytes

numero di elementi dell'array: sizeof(my_array) / sizeof(int) = 100



Array statico di int

```
void funzione(int size) { // variable length array, C99
        int array[size];  // ok
        …
}
// https://en.m.wikipedia.org/wiki/Variable-length_array
```

array viene allocato a runtime sulla stack, durante la chiamata di funzione.

La dimensione dell'array dipende dal parametro "size"

Array dinamico di int

// allochiamo dinamicamente un array di int di dimensione 1024

int arr [] = calloc(1024, **sizeof(int)**); // memory set to zero by calloc

arr[0] = 0xDEADBEEF;

…

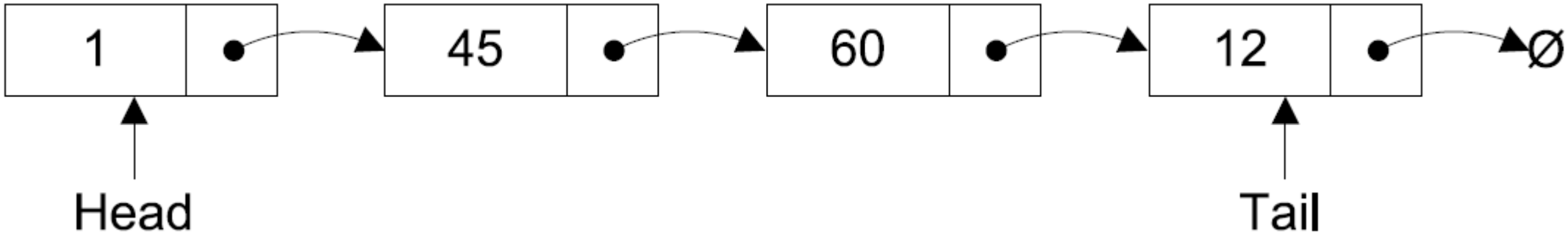
arr[2048] = 0; // nessun controllo automatico su indice!!!
Sovrascriviamo qualche altro dato!!! BUG!!!

arr = realloc(arr, 2048); // ridimensionamento, può comportare la copiatura di tutto l'array originale nel nuovo array

…

free(arr); // importantissimo!!! Altrimenti «memory leak»!!!

Linked List



- Linked list è una serie di nodi; ogni nodo ha al più un puntatore al nodo successivo; è «dinamica»
- L'ultimo nodo punta a «null» (cioè non ci sono ulteriori nodi nella linked list)
- La linked list ha una «testa» (head) ed una coda (tail), per permettere l'inserimento di un nodo aggiuntivo in testa od in coda in modo efficiente
- Head == null → Linked List vuota
- Testo: «Data Structures and Algorithms», Barnett, Del Tongo

Inserimento elemento in Linked List

- Aggiungere il nuovo nodo «in fondo» alla Linked List:
- Se head == null, → head = tail = nuovo_nodo; end
- else: accodiamo il nuovo nodo alla fine della tail
- (in alternativa: lo inserisco in testa)

```
1) algorithm Add(value)
2)     Pre: value is the value to add to the list
3)     Post: value has been placed at the tail of the list
4)     n ← node(value)
5)     if head = ∅
6)         head ← n
7)         tail ← n
8)     else
9)         tail.Next ← n
10)        tail ← n
11)    end if
12) end Add
```

Ricerca nella Linked List

- Ricerca: attraversiamo la lista alla ricerca di un certo valore, partendo da head

```
1) algorithm Contains(head, value)
2)     Pre:  head is the head node in the list
3)           value is the value to search for
4)     Post: the item is either in the linked list, true; otherwise false
5)     n ← head
6)     while n ≠ ∅ and n.Value ≠ value
7)         n ← n.Next
8)     end while
9)     if n = ∅
10)        return false
11)    end if
12)    return true
13) end Contains
```

Rimozione nodo da Linked List

- Vogliamo rimuovere un certo valore dalla lista (non sappiamo dove si trova), allora i casi da gestire sono:
- 1. la lista è vuota; oppure
- 2. il nodo da rimuovere è l'unico della lista; oppure
- 3. il nodo da rimuovere è head; oppure
- 4. il nodo da rimuovere è tail; oppure
- 5. il nodo da rimuovere si trova da qualche parte tra head e tail; oppure
- 6. il valore da rimuovere non è presente nella lista!