Programmazione

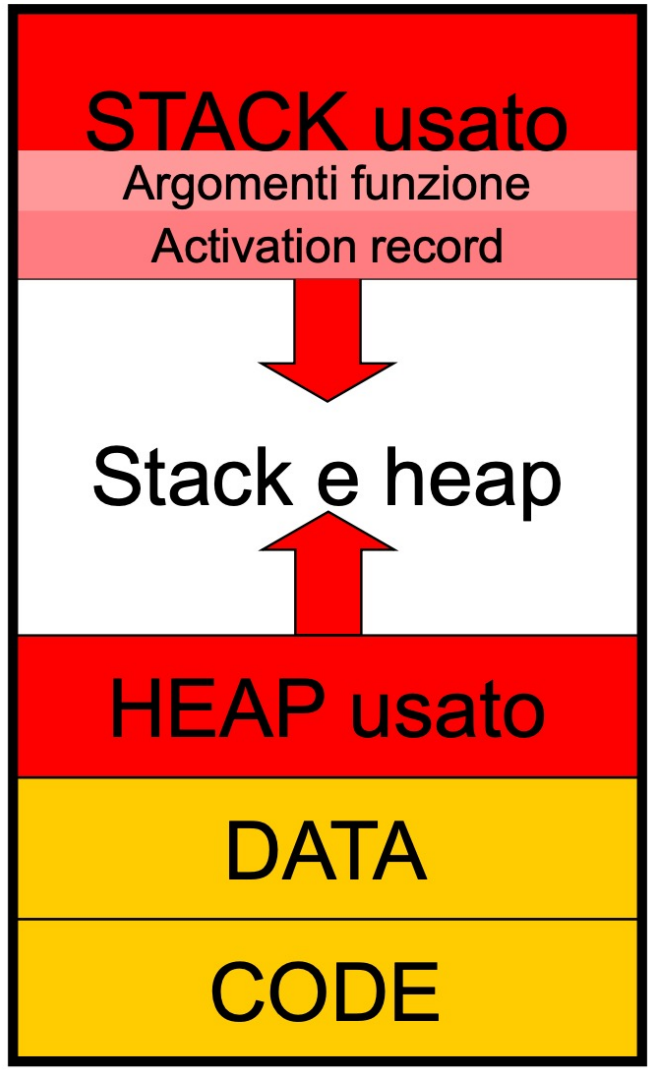
Giovanni Da San Martino

Dipartimento of Matematica, Università degli Studi di Padova
giovanni.dasanmartino@unipd.it
A.A. 2021-2022

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Chiamata di Funzione

- Il codice di una funzione è in code
- i parametri e le variabili locali di una funzione vengono allocati nello stack (pila)
- vengono prima copiati i valori dei suoi argomenti e poi vi viene allocato un Activation Record (anche detto stack frame) in cui sono allocate le variabili locali della funzione e altro
- Quando la funzione termina, l'AR e gli argomenti vengono rimossi dallo stack che quindi ritorna nello stato

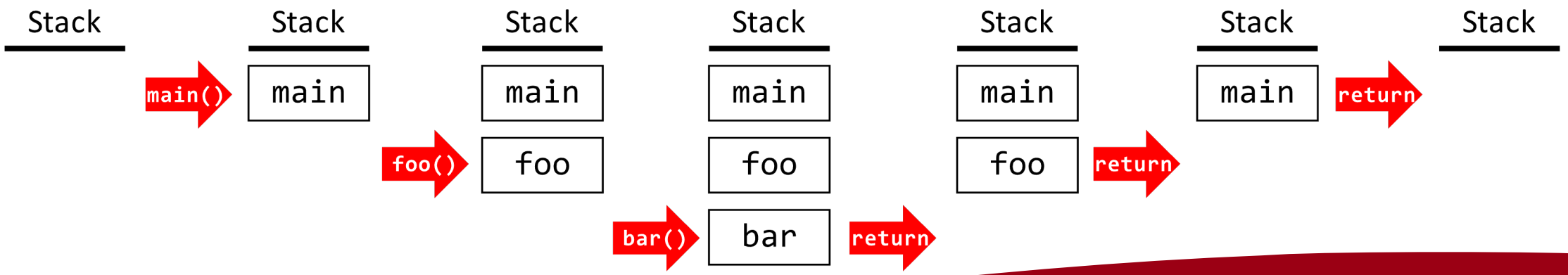


Previuosly on Programmazione


```
void bar() {}

void foo() { bar(); }

int main() {
  foo();
}
```



Tipi

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Tipi di Dato

- Il calcolatore è in grado di elaborare differenti tipi di informazione:
  - numeri, caratteri, immagini, suoni, video

Informazione = Dati + Interpretazione

- Le procedure di codifica/decodifica vengono eseguite dal calcolatore, quindi devono essere pensate in modo che i dati siano facilmente manipolabili dall'elaboratore (più che facilmente comprensibili dall'uomo)

Rappresentazione di Interi Positivi

- Una sequenza di cifre forma un numero secondo la seguente convenzione: $374 = 3 \cdot 10^2 + 7 \cdot 10^1 + 4 \cdot 10^0$
- Per determinare il valore di un numero binario positivo, si utilizza lo stesso algoritmo dove però la base è 2:

$$(1101)_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 8 + 4 + 1 = 13$$

- Il numero di configurazioni diverse di n bit è 2^n, per cui si riescono a rappresentare 2^n numeri diversi.
- Il numero più grande rappresentabile con n bit è $2^n - 1$ (perché si inizia a contare da 0). Il calcolatore non può rappresentare infiniti numeri!

Rappresentazione di Interi Positivi

- Trasformazione da base 10 a base k

1. Dividere il numero per k
2. tenete traccia del resto
3. se il quoziente è maggiore di 0 ripetere il passo 1 con il quoziente
4. scrivere i resti nell'ordine inverso rispetto al quale sono stati ottenuti

numero	quoziente	resto
43/2	21	1
21/2	10	1
10/2	5	0
5/2	2	1
2/2	1	0
1/2	0	1

$$(43)_{10} = (101011)_2$$

Rappresentazione di Interi con Segno

- Il bit più a sinistra rappresenta il segno: 0 = positivo, 1 = negativo
- se si utilizzano n bit, si riescono a rappresentare tutti i numeri

$$x. -(2^{n-1}-1) \leq x \leq 2^{n-1}-1 \Leftrightarrow -2^{n-1}+1 \leq x \leq 2^{n-1}-1$$

Es. con 4 bit si rappresenta [-7,7]

Binario	Decimale		Binario	Decimale
0000	0		1000	0
0001	1		1001	-1
0010	2		1010	-2
0011	3		1011	-3
0100	4		1100	-4
0101	5		1101	-5
0110	6		1110	-6
0111	7		1111	-7

Rappresentazione in complemento a 2

- I numeri positivi sono rappresentati in modo “standard” (come nella notazione modulo e segno), utilizzando n bit
- I numeri negativi sono rappresentati “in complemento a 2”, ovvero si somma 2^n al numero e poi rappresenta in modo “standard”. Es. n = 4

$$(-3)_{10} \rightarrow 2^4 - 3 = 16 - 3 = (13)_{10} = (1101)$$

Rappresentazione in complemento a 2

Rappresentazione in complemento a 2 con 4 bit

Binario	Decimale		Binario	Decimale
0000	0		1000	-8
0001	1		1001	-7
0010	2		1010	-6
0011	3		1011	-5
0100	4		1100	-4
0101	5		1101	-3
0110	6		1110	-2
0111	7		1111	-1

Una sola rappresentazione per lo zero

Rappresentazione di Interi con Segno

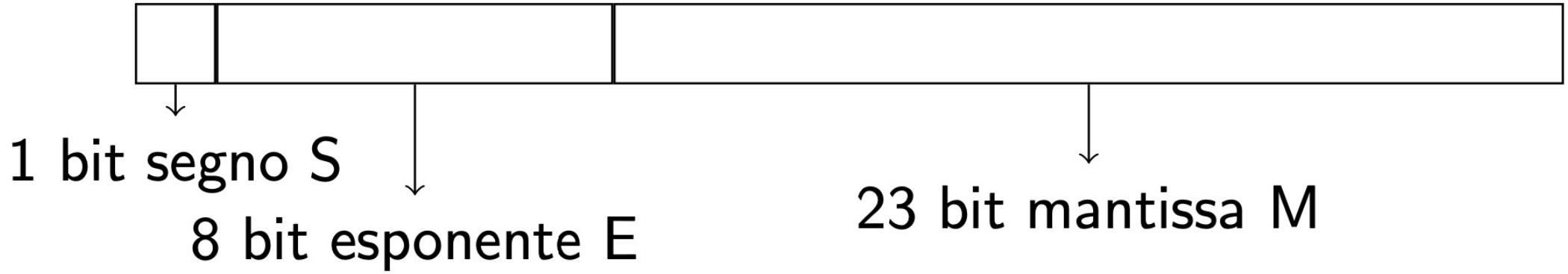
- I numeri interi positivi sono rappresentati all'interno dell'elaboratore utilizzando un multiplo del byte (generalmente 4 o 8 byte)
- se volete verificare la dimensione di un int, il comando sizeof(int) restituisce il numero di byte (celle di memoria) occupati da un int
- Il file limits.h (#include <limits.h>) riporta una serie di costanti numeriche tra cui il massimo numero rappresentabile sul computer che si sta usando: INT_MAX

Rappresentazione di Interi con Segno

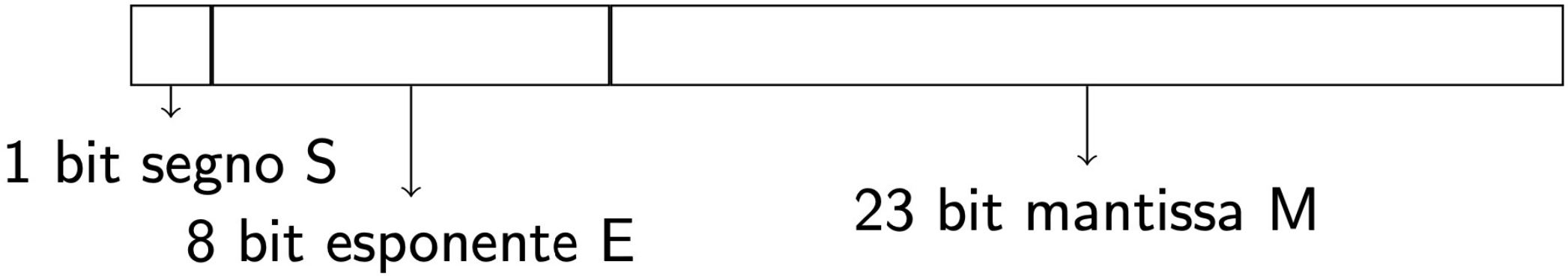
- INT_MAX = 2147483647
- INT_MAX + 1 = -2147483648 //dovuto alla rappresentazione in complemento a 2
- Non esiste un errore di overflow
- Varianti del tipo int
  - long x: un intero che usa il doppio dei bit di un int [printf("%ld",x)]
  - short x: un intero che usa la metà dei bit di un int [printf("%hd",x)]
  - unsigned int x: un intero positivo [printf("%u",x)]
  - unsigned long x: un long positivo [printf("%lu",x)]

Numeri Reali

- I numeri reali utilizzano la rappresentazione in virgola mobile
- Si basa sulla notazione scientifica $1.40 \cdot 102 = 140$ (notate che c'è solo una cifra intera, ovvero la notazione è normalizzata)
- Lo standard IEEE 754 prevede vari tipi di numeri in virgola mobile, tra cui:
  - singola precisione (32 bit) e doppia precisione (64 bit)
- i numeri a singola precisione hanno il seguente formato:



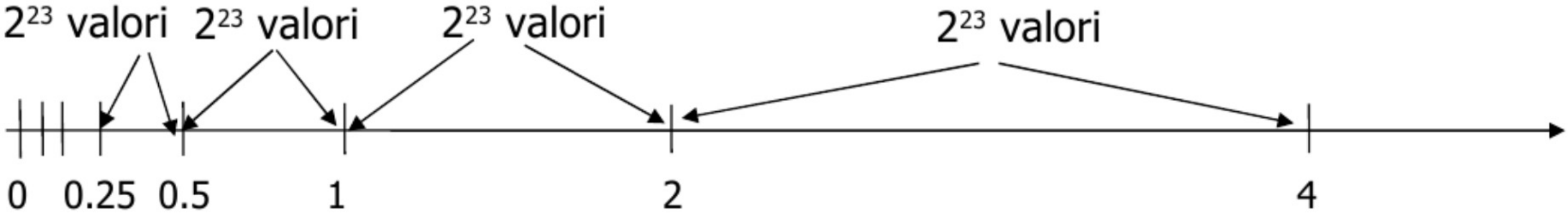
Numeri Reali



- Il formato IEEE 754 è $(-1)^S \cdot 1.M \cdot 2^{E-127}$
- 1.M, la mantissa indica il numero vero e proprio in forma normalizzata
- 2^{E-127} indica dove mettere la virgola (moltiplicare/dividere per 2 un numero binario significare spostare a destra/sinistra la virgola di una posizione)

Numeri Reali

- Intervallo rappresentabile in singola precisione: circa da $1.4 \cdot 10^{-45}$ a $6.81 \cdot 10^{38}$
- in totale si riescono a rappresentare 2^{32} numeri distinti (metà positivi, metà negativi)



- I numeri rappresentabili non sono distribuiti uniformemente,ma sono più densi vicino allo zero.

Numeri Reali in C

- Tipo reale in singola precisione: float x [printf("%f",x)]
- Tipo reale in doppia precisione: double x [printf("%f",x)]
- Es. double x = 3.2;
- Esiste anche il tipo long double x [printf("%Lf",x)]
- Attenzione: poiché i reali non hanno precisione infinita, può darsi che, confrontando due espressioni (complesse) che sappiamo restituire lo stesso valore, l'operatore di uguaglianza restituisca falso per colpa delle approssimazioni durante calcoli intermedi.

Caratteri

- Lo standard di codifica più diffuso è il codice ASCII, per American Standard Code for Information Interchange
- Definisce una tabella di corrispondenza fra ciascun simbolo (carattere minuscolo, maiuscolo, cifre) e un codice a 7 bit (128 caratteri)
- UNICODE (UTF-8 e UTF-16): standard proposto a 8 e 16 bit (65.536 caratteri)
- dichiarazione di una variabile carattere in C: char x [printf(“%c”,x)]
- char x = ‘c’; //dichiarazione ed inizializzazione

Caratteri

Tabella ASCII

	0	1	2	3	4	5	6	7
0	NUL	DLE	space	0	@	P	`	p
1	SOH	DC1 XON	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3 XOFF	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z

Conversioni tra tipi

- Gli operatori aritmetici ed i comandi sono definiti tra termini dello stesso tipo
- Ma il C effettua automaticamente alcune conversioni tra tipi, le promozioni:
  - il tipo con meno capacità espressiva viene promosso al tipo con maggiore capacità espressiva (ovvero presi due elementi nella tabella a fianco, la conversione avviene a quello più in alto)
  - quando si cambia il tipo di un'espressione (non di una variabile), viene fatta una copia del valore temporanea per effettuare il calcolo

long double
double
float
long
int , unsigned int
short, char

Conversioni tra tipi

- Attenzione: alcune conversioni, per esempio tra int e unsigned int, possono produrre effetti inaspettati.

```
int g = -6
unsigned int gg = 3;
printf("%u\n", gg+g);
printf("%d\n", gg+g);
```

- Stampa:

```
4294967293
-3
```

Conversioni tra tipi

- Attenzione: alcune conversioni, per esempio tra int e unsigned int, possono produrre effetti inaspettati.

```
int g = -6
unsigned int gg = 3;
printf("%u\n", gg+g);
printf("%d\n", gg+g);
```

- Stampa:

```
4294967293
-3
```

- E' possibile forzare la conversione ad un tipo. Es.

```
int x = (int) 3.2 + 3;
printf("%d", x); // stampa 6!
```

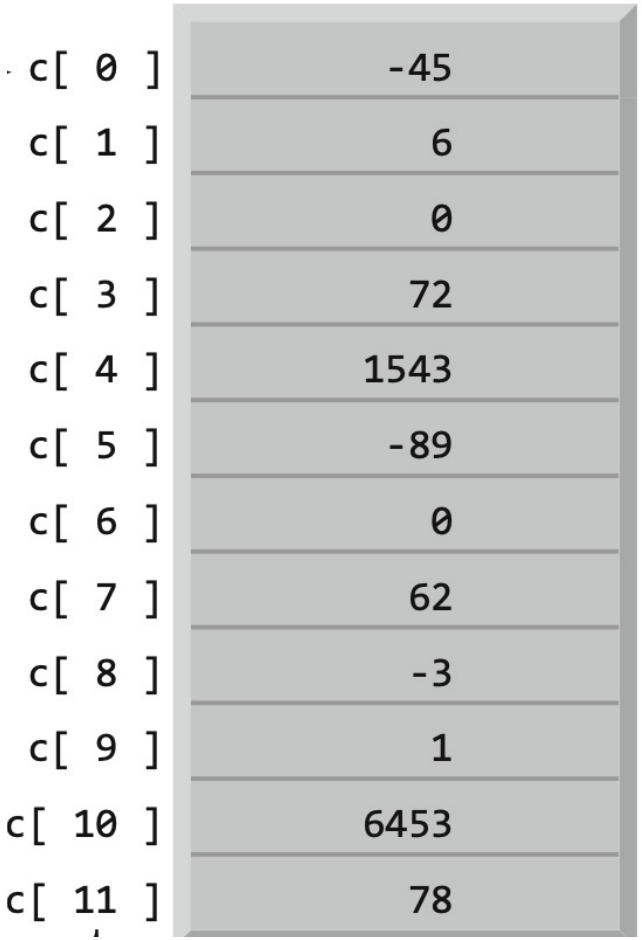
- Attenzione che forzando la conversione, per esempio da float a int, si può perde informazione!

Array

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

Array

- Un array è un gruppo di locazioni di memoria contigue che hanno tutte lo stesso tipo.
- Dichiarazione: tipo nome[dimensione];
- Es. int c[12]; // dichiara un array (una sequenza) di 12 interi
- c[i] si comporta come una variabile
- c[i] accede all'i-esimo elemento dell'array (si usa la parola indice per riferirsi al numero tra []):
  - Il primo elemento ha indice 0
  - l'ultimo ha indice dimensione-1
- Es. printf("secondo elemento di c=%d", c[1])
- c[2] = 1; // il valore del terzo elemento dell'array passa da 0 a 1



Array: Esempi

- int n[5];
  for (int i = 0; i < 5; i=i+1) { // inizializza a zero gli elementi dell'array
  n[i] = 0;
  }
  for (int i = 0; i < 5; i=i+1) { // stampa gli elementi dell'array
  printf("%d = %d\n", i, n[i]);
  }
- int n[5] = {32, 27, 64, 18, 95}; // dichiara ed inizializza l'array
  int n[] = {32, 27, 64, 18, 95}; // dichiarazione equivalente

Confini di un Array

- Il C non ha meccanismi di controllo dei confini di un array
- Un programma può fare riferimento a un elemento che non esiste e non ricevere un errore!

- int n[5] = {32, 27, 64, 18, 95};

```
for (int i = 0; i < 6; i=i+1) { // stampa gli elementi dell'array
        printf("%d = %d\n", i, n[i]);
    }
```

- Nell'ultima iterazione si stampa il contenuto della cella di memoria seguente all'array; se siamo fortunati questo genera un errore, altrimenti viene stampato un valore impredicibile.

Stringhe (Sequence di Caratteri)

- Una sequenza di caratteri viene anche chiamata stringa. Es.

- char string1[] = ”ciao mondo";

printf(“la stringa inizia per %c\n”, string1[0]); // stampa c

printf("%s\n", string1); // stampa ciao mondo, %s sta per stringa

Nota per il prossimo laboratorio

- Il C fornisce il comando scanf per ricevere un input da tastiera durante l'esecuzione. Introduciamo il comando in modo da poter effettuare, per alcuni esercizi, un maggior numero di test automatici.
- La sintassi ricorda quella di printf:

int x; scanf("%d", &x);

- nell'esempio legge un intero (%d) da tastiera (durante l'esecuzione in locale si dovrà digitare un intero e premere invio, quando si sottopone la soluzione su Moodle il sistema farà tutto automaticamente)
- Vi forniremo noi il codice per scanf, per adesso basta che abbiate un'idea di cosa succede. Nelle prossime lezioni capiremo cosa significa &.
- Solo per il prossimo laboratorio non usate array come parametri di funzione