

 UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

Corso di Laurea in Informatica (Track B) - A.A. 2018/2019

Laboratorio di Informatica

Linguaggio C

(Parte 2 – Array, Matrici, Struct)

docente: Veronica Rossano

veronica.rossano@uniba.it

Recap: Programmazione Strutturata

- Teorema di Bohm e Jacopini:** tutti i programmi possono essere scritti usando tre strutture di controllo fondamentali
 - Sequenza:**
 - Nativa nel C. I programmi vengono eseguiti sequenzialmente per default
 - Selezione:**
 - Il C ne ha tre tipi: if, if...else, e switch
 - Iterazione:**
 - Il C ne ha tre tipi: while, do...while e for



Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

Recap: Problem Solving

- Prima** di scrivere un programma
 - Comprendere a **fondo il problema (analisi)**
 - Pianificare con cura un **approccio per risolverlo** (approccio **top-down, bottom-up**)
 - Produrre una soluzione in pseudo-codice o con i flow-chart
- Mentre** scrivete un programma
 - Individuate quali “building blocks” sono disponibili (**riuso** del codice)
 - La maggior parte dei programmi segue una struttura “standard”
 - Definizione e inizializzazione** delle variabili
 - Elaborazione** dei dati
 - Visualizzazione** in output dei risultati

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19

2

Struttura di Selezione

Usata per scegliere tra diverse alternative

Istruzione	Pseudocodice	Traduzione in C
if	Se il voto dello studente è maggiore di 18 Stampa "Promosso"	if (voto >= 18) puts("Promosso\n");
if...else	Se il voto dello studente è maggiore di 18 Stampa "Promosso" Altrimenti Stampa "Bocciato"	if (voto >= 18) { puts("Promosso\n"); } else puts("Bocciato\n"); Oppure voto >= 60 ? puts("Promosso\n"): puts("Bocciato\n");

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19

4

Struttura di Selezione

Usata per scegliere tra diverse alternative

Istruzione	Pseudocodice	Traduzione in C
switch	Se il voto dello studente è maggiore di 18 Stampa "Promosso" Altrimenti Stampa "Bocciato"	<pre>switch(voto { case 0: case 1: case 2: case 3: case 4: case 5: case 6: case 7: case 8: case 9: case 10: case 11: case 12: case 13: case 14: case 15: case 16: case 17: puts("Bocciato"); break; default: puts("Promosso"); }</pre>



Lo pseudocodice è analogo a quella dell'istruzione if...else
ma si esprime in modo diverso.

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

5

Struttura di Iterazione

Utilizzata per esprimere operazioni che si ripetono finché una determinata condizione resta vera.

Istruzione	Pseudocodice	Traduzione in C
while	finché(numero_prodotto<=5) leggi costo aggiungi il costo al totale	<pre>while(products<=5){ scanf("%d",&costo); totale = totale + costo; }</pre>
do...while	ripeti leggi costo aggiungi il costo al totale finché(numero_prodotto<=5)	<pre>do { scanf("%d",&costo); totale = totale + costo; } while(products<=5);</pre>
for	finché(numero_prodotto<=5) leggi costo aggiungi il costo al totale	<pre>for(products=0; products<=5; products++) { scanf("%d",&costo); totale = totale + costo; }</pre>

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

6

Struttura di Iterazione (non controllata)

Utilizzata per esprimere operazioni che si ripetono finché una determinata condizione resta vera.

Istruzione	Pseudocodice	Traduzione in C
while	finché(sentinella=true) leggi costo aggiungi il costo al totale	<pre>while(costo != -1){ scanf("%d",&costo); totale = totale + costo; }</pre>
Do..while	ripeti leggi costo aggiungi il costo al totale finché(sentinella=true)	<pre>do { scanf("%d",&costo); totale = totale + costo; } while(costo != -1);</pre>
for	finché(sentinella=true) leggi costo aggiungi il costo al totale	<pre>for(costo = 0; costo != -1;) { scanf("%d",&costo); totale = totale + costo; }</pre>

Suggerimento: **while** e **do...while** si tendono a preferire per le iterazioni non controllate, mentre il **for** è la struttura più semplice da adottare quando sappiamo a priori il numero di iterazioni da eseguire è noto.

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

7

Recap: operatori di assegnamento

Siano: int c = 3, d = 5, e = 4, f = 6, g = 12;			
Operatore di assegnamento	Esempio di espressione	Spiegazione	Assegna
+=	c += 7	c = c + 7	10 a c
-=	d -= 4	d = d - 4	1 a d
*=	e *= 5	e = e * 5	20 a e
/=	f /= 3	f = f / 3	2 a f
%=	g %= 9	g = g % 9	3 a g

Sono più efficienti perché riducono gli accessi in memoria non creando oggetti temporanei

Veronica Rossano - Linguaggio C (parte 1)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

Recap: precedenza degli operatori

Operatori						Associatività	Tipo
++	--	+	-	!	(type)	right to left	unary
*	/	%				left to right	multiplicative
+	-					left to right	additive
<	<=	>	>=			left to right	relational
==	!=					left to right	equality
&&						left to right	logical AND
						left to right	logical OR
? :						right to left	conditional
=	+=	-=	*=	/=	%=	right to left	assignment
,						left to right	comma

Veronica Rossano - Linguaggio C (parte 1)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

Array

- Cosa è un array?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

11

Fine Recap.

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

10

Array

- Cosa è un array?

- Gruppo di locazioni di memoria consecutive
- Stesso nome e tipo
- Modella **elementi di tipo omogeneo**

14
31
55
64
78
32
146
6
35
9
0
234

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

12

Array

- Cosa è un array?**
 - Gruppo di locazioni di memoria consecutive
 - Stesso nome e tipo
 - Modella elementi di tipo omogeneo
- Per riferirsi a un elemento, specificare**
 - Nome dell'array + Posizione**
 - Normalmente, per le variabili usiamo solo il nome

Nome dell'array
(Tutti gli elementi di questo array hanno lo stesso nome, c)

c[0]	14
c[1]	31
c[2]	55
c[3]	64
c[4]	78
c[5]	32
c[6]	146
c[7]	6
c[8]	35
c[9]	9
c[10]	0
c[11]	234

Numero di posizione dell'elemento nell'array c

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 13

Array

- Cosa è un array?**
 - Gruppo di locazioni di memoria consecutive
 - Stesso nome e tipo
 - Modella elementi di tipo omogeneo
- Per riferirsi a un elemento, specificare**
 - Nome dell'array + Posizione**
 - Normalmente, per le variabili usiamo solo il nome

Nome dell'array
(Tutti gli elementi di questo array hanno lo stesso nome, c)

c[0]	14
c[1]	31
c[2]	55
c[3]	64
c[4]	78
c[5]	32
c[6]	146
c[7]	6
c[8]	35
c[9]	9
c[10]	0
c[11]	234

Numero di posizione dell'elemento nell'array c

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 14

Array

- Sono delle normali variabili**
 - Possono essere utilizzate nelle operazioni di lettura/scrittura
 - `printf("%d", c[0]) - scanf("%d", c[0])`
 - Gli indici possono anche essere sostituiti da espressioni
 - `c[5+2] == c[3] == c[x]`
- Sintassi C**
 - Tipo dell'array + Nome + Numero di elementi


```
arrayType arrayName[ numberOfElements ];
```
 - Esempi:**

```
int c[10];
float myArray[3284];
```

Nome dell'array
(Tutti gli elementi di questo array hanno lo stesso nome, c)

c[0]	14
c[1]	31
c[2]	55
c[3]	64
c[4]	78
c[5]	32
c[6]	146
c[7]	6
c[8]	35
c[9]	9
c[10]	0
c[11]	234

Numero di posizione dell'elemento nell'array c

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 15

Array

- Sono delle normali variabili**
 - Devono essere inizializzate**
 - `int n[5] = { 1, 2, 3, 4, 5 };`
 - `int n[5] = {0} (tutti i valori vengono impostati pari a zero)`
 - `int n[5] = {1, 2, 3} (i valori mancanti vengono impostati pari a zero)`
 - Se inizializzati, la dimensione può anche essere omessa**
 - `int n[] = { 1, 2, 3, 4, 5 };`
 - Se un array non è inizializzato in fase di definizione, occorre procedere a un ciclo di inizializzazione**
 - Importante:** il linguaggio C non effettua alcun controllo sugli indici degli array! In fase di codifica del programma bisognerà controllare che il ciclo non vada oltre l'array.

Nome dell'array
(Tutti gli elementi di questo array hanno lo stesso nome, c)

c[0]	14
c[1]	31
c[2]	55
c[3]	64
c[4]	78
c[5]	32
c[6]	146
c[7]	6
c[8]	35
c[9]	9
c[10]	0
c[11]	234

Numero di posizione dell'elemento nell'array c

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 16

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19     // ciclo di assegnazione
20     for (i=0; i<SIZE; i++) {
21         n[i] = rand() % MAX_VALUE + MIN_VALUE;
22     }
23
24     // ciclo di stampa
25     for (i=0; i<SIZE; i++) {
26         printf("%d \t %d\n", i, n[i]);
27     }
28 }
```

Il concetto di array è strettamente correlato al **concetto di iterazione**

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

17

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19     // ciclo di assegnazione
20     for (i=0; i<SIZE; i++) {
21         n[i] = rand() % MAX_VALUE + MIN_VALUE;
22     }
23
24     // ciclo di stampa
25     for (i=0; i<SIZE; i++) {
26         printf("%d \t %d\n", i, n[i]);
27     }
28 }
```

Ogni qual volta utilizziamo un array **abbiamo bisogno di uno (o più) cicli** per effettuare le operazioni previste dall'algoritmo

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

18

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19     // ciclo di assegnazione
20     for (i=0; i<SIZE; i++) {
21         n[i] = rand() % MAX_VALUE + MIN_VALUE;
22     }
23
24     // ciclo di stampa
25     for (i=0; i<SIZE; i++) {
26         printf("%d \t %d\n", i, n[i]);
27     }
28 }
```

Ciascun ciclo «copre» i classici passaggi previsti dagli algoritmi (**inizializzazione delle variabili, elaborazione, stampa dell'output**)

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

19

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19     // ciclo di assegnazione
20     for (i=0; i<SIZE; i++) {
21         n[i] = rand() % MAX_VALUE + MIN_VALUE;
22     }
23
24     // ciclo di stampa
25     for (i=0; i<SIZE; i++) {
26         printf("%d \t %d\n", i, n[i]);
27     }
28 }
```

Riga 1-3
Includo le librerie necessarie

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

20

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19 [...]

```

```

19    // ciclo di assegnazione
20    for (i=0; i<SIZE; i++) {
21        n[i] = rand() % MAX_VALUE + MIN_VALUE;
22    }
23
24    // ciclo di stampa
25    for (i=0; i<SIZE; i++) {
26        printf("%d \t %d\n", i, n[i]);
27    }
28

```

Riga 4-6

Dichiaro le costanti simboliche
Importante farlo sempre!

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

21

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19 [...]

```

```

19    // ciclo di assegnazione
20    for (i=0; i<SIZE; i++) {
21        n[i] = rand() % MAX_VALUE + MIN_VALUE;
22    }
23
24    // ciclo di stampa
25    for (i=0; i<SIZE; i++) {
26        printf("%d \t %d\n", i, n[i]);
27    }
28

```

Riga 9-12

Dichiaro le variabili e preparo la
randomizzazione

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

22

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19 [...]

```

```

19    // ciclo di assegnazione
20    for (i=0; i<SIZE; i++) {
21        n[i] = rand() % MAX_VALUE + MIN_VALUE;
22    }
23
24    // ciclo di stampa
25    for (i=0; i<SIZE; i++) {
26        printf("%d \t %d\n", i, n[i]);
27    }
28

```

Riga 15-17

Primo ciclo: inizializzo gli elementi
dell'array (sono delle variabili quindi è
importante inizializzarli!)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

23

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19 [...]

```

```

19    // ciclo di assegnazione
20    for (i=0; i<SIZE; i++) {
21        n[i] = rand() % MAX_VALUE + MIN_VALUE;
22    }
23
24    // ciclo di stampa
25    for (i=0; i<SIZE; i++) {
26        printf("%d \t %d\n", i, n[i]);
27    }
28

```

Riga 20-22

Secondo ciclo: assegno un valore random
a ciascun elemento dell'array

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

24

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19     // ciclo di assegnazione
20     for (i=0; i<SIZE; i++) {
21         n[i] = rand() % MAX_VALUE + MIN_VALUE;
22     }
23
24     // ciclo di stampa
25     for (i=0; i<SIZE; i++) {
26         printf("%d \t %d\n", i, n[i]);
27     }
28 }
```

Riga 25-27

Terzo ciclo: stampo i valori in output

Array



Output del programma

```

1  gcc version 4.6.3
2
3
4
5  0    94
6  1    49
7  2    11
8  3    96
9  4    76
10 5    33
11 6    57
12 7    41
13 8    17
14 9    68
15
16
17
18
19
20
21
22
23
24
25
26
27
28
```

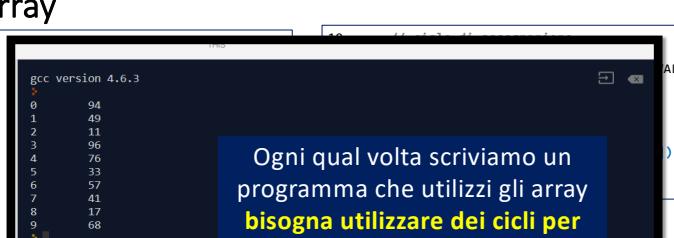
Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 25

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 26

Array



**Ogni qual volta scriviamo un programma che utilizza gli array
bisogna utilizzare dei cicli per inizializzare i dati, elaborarli e mostrare i risultati!**

Array

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <time.h>
4  #define SIZE 10
5  #define MIN_VALUE 1;
6  #define MAX_VALUE 100;
7
8  int main() {
9      int n[SIZE];
10     int i = 0;
11     int seed = time(NULL);
12     srand(seed);
13
14     // ciclo di inizializzazione
15     for (i=0; i<SIZE; i++) {
16         n[i] = 0;
17     }
18
19     // ciclo di assegnazione
20     for (i=0; i<SIZE; i++) {
21         n[i] = rand() % MAX_VALUE + MIN_VALUE;
22     }
23
24     // ciclo di stampa
25     for (i=0; i<SIZE; i++) {
26         printf("%d \t %d\n", i, n[i]);
27     }
28 }
```

Supponiamo di estendere il programma, calcolando la somma degli elementi dell'array

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 27

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 28

Array

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 #define SIZE 10
5 #define MIN_VALUE 1
6 #define MAX_VALUE 100
7
8 int main() {
9     int n[SIZE];
10    int i = 0;
11    int seed = time(NULL);
12    srand(seed);
13
14    // ciclo di inizializzazione
15    for (i=0; i<SIZE; i++) {
16        n[i] = 0;
17    }
18
19    // ciclo di assegnazione
20    for (i=0; i<SIZE; i++) {
21        n[i] = rand() % MAX_VALUE + MIN_VALUE;
22    }
23
24    // ciclo di stampa
25    for (i=0; i<SIZE; i++) {
26        printf("%d \t %d\n", i, n[i]);
27    }
28 }
```

Come modifichiamo il codice sorgente?

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 29

Array

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4 #define SIZE 10
5 #define MIN_VALUE 1
6 #define MAX_VALUE 100
7
8 int main() {
9     int n[SIZE];
10    int i = 0;
11    int seed = time(NULL);
12    srand(seed);
13    int somma = 0; ←
14
15    // ciclo di inizializzazione
16    for (i=0; i<SIZE; i++) {
17        n[i] = 0;
18    }
19
20    // ciclo di assegnazione
21    for (i=0; i<SIZE; i++) {
22        n[i] = rand() % MAX_VALUE + MIN_VALUE;
23    } ←
24
25    // ciclo di stampa
26    for (i=0; i<SIZE; i++) {
27        printf("%d \t %d\n", i, n[i]);
28    }
29    // stampa dell'output finale
30    printf("%d", somma); ←
31 }
```

E' sufficiente modificare solo tre istruzioni. La struttura del programma resta identica!

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 30

Problema 2.1

Scrivere un programma che calcoli la media delle calorie assunte in una settimana. Requisiti: utilizzare un array

Input?
Vettore di valori. *Quanti?*

Output?
Media dei valori

Quale tipologia di istruzioni ci serve?
Ciclo di inizializzazione dei valori, Ciclo per il calcolo del valore medio, Stampa del valore.

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 31

Problema 2.1

Scrivere un programma che calcoli la media delle calorie assunte in una settimana. Requisiti: utilizzare un array

Input?
Vettore di valori. *Quanti?*

Output?
Media dei valori

Quale tipologia di istruzioni ci serve?
Ciclo di inizializzazione dei valori, Ciclo per il calcolo del valore medio, Stampa del valore.

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 32

Problema 2.1

Scrivere un programma che calcoli la media delle calorie assunte in una settimana. Requisiti: utilizzare un array

Input?

Vettore di valori. Quanti ?

Output?

Media dei valori

Quale tipologia di istruzioni ci serve?

Ciclo di inizializzazione dei valori, Ciclo per il calcolo del valore medio, Stampa del valore.

Codificare la soluzione con una sessione su Repl.it

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

33

Soluzione

```
gcc version 4.6.3
>
Inserire calorie assunte il giorno 1: 1700
Inserire calorie assunte il giorno 2: 2100
Inserire calorie assunte il giorno 3: 1500
Inserire calorie assunte il giorno 4: -100
Inserire un valore valido
Inserire calorie assunte il giorno 4: 1300
Inserire calorie assunte il giorno 5: 2300
Inserire calorie assunte il giorno 6: 2700
Inserire calorie assunte il giorno 7: 2000
Inserire calorie assunte il giorno 8: 1942.86>
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

34

Soluzione

```
gcc version 4.6.3
>
Inserire calorie assunte il giorno 1: 1700
Inserire calorie assunte il giorno 2: 2100
Inserire calorie assunte il giorno 3: 1500
Inserire calorie assunte il giorno 4: -100
Inserire un valore valido
Inserire calorie assunte il giorno 4: 1300
Inserire calorie assunte il giorno 5: 2300
Inserire calorie assunte il giorno 6: 2700
Inserire calorie assunte il giorno 7: 2000
Inserire calorie assunte il giorno 8: 1942.86>
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Importante: Implementare i controlli sulla correttezza dell'input (**programmazione difensiva**)

Soluzione

```
1 #include <stdio.h>
2 #define GIORNI_SETTIMANA 7 // costanti simboliche
3
4 int main(void) {
5     int calorie[GIORNI_SETTIMANA]; // vettore delle calorie
6     int somma_calorie = 0; // variabili locali
7     float media_calorie = 0.0;
8
9     // ciclo di inizializzazione
10    for(int i=0; i<GIORNI_SETTIMANA; i++) {
11        calorie[i] = 0;
12    }
13}
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

36

Innanzitutto dichiariamo le variabili necessarie e risolvere il problema e inizializziamo l'array

Soluzione (cont.)

```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF-Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

37

Soluzione (cont.)

```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF-Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

38

Soluzione (cont.)

Riga 15
Chiede all'utente di inserire un valore
di input finché non è corretto



```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF-Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

39

Soluzione (cont.)

Riga 19
Se il valore non è corretto, stampa un
messaggio e resta nel ciclo



```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF-Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

40

Soluzione (cont.)

Riga 22
Se è corretto registra il valore ed esce dal ciclo

```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19
Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

41

Soluzione (cont.)

Riga 26
Calcolo della media (attenti al cast!)

```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19
Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

42

Soluzione (cont.)

Extra

Modificare il programma per fare in modo che invece di «... il giorno X», appaia il vero nome del giorno (Lunedì, Martedì, Mercoledì, etc.).

```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19
Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

43

Soluzione (cont.)

L'utilizzo dei cicli rende il programma più **leggibile e modulare**. La parte di **inizializzazione** viene divisa rispetto a quella di **elaborazione e visualizzazione**.

```

11 // ciclo di acquisizione ed elaborazione
12 for(int i=0; i<GIORNI_SETTIMANA; i++) {
13
14     // programmazione difensiva - controlla valori di input non corretti
15     while(calorie[i] <= 0) {
16         printf("\nInserire calorie assunte il giorno %d: ", i+1);
17         scanf("%d",&calorie[i]);
18
19         if(calorie[i] <= 0) // mostra un messaggio d'errore
20             printf("\nInserire un valore valido\n");
21         else // altrimenti registra il valore
22             somma_calorie = somma_calorie + calorie[i];
23     }
24 }
25
26 media_calorie = (float) somma_calorie / GIORNI_SETTIMANA;
27 printf("\n Media calorie assunte: %.2f", media_calorie);
28 }
```

11/03/19
Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

44

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

45

Utilizzo Avanzato degli Array



```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

11/03/19

Riga 2-3
Costanti Simboliche: SEMPRE!

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

46

Utilizzo Avanzato degli Array

Riga 8-9

Dichiaro due array: uno contiene le valutazioni
degli studenti, uno la frequenza dei voti



```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

47

Utilizzo Avanzato degli Array

Riga 8-9

Dichiaro due array: uno contiene le valutazioni
degli studenti, uno la frequenza dei voti



```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

48

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10        4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 8-9

Dichiaro due array: uno contiene le valutazioni
degli studenti, uno la frequenza dei voti

Inizializza tutti gli
elementi a zero

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

49

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10        4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 8-9

Dichiaro due array: uno contiene le valutazioni
degli studenti, uno la frequenza dei voti

Inizializziamo con
valori espliciti

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

50

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10        4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 12-14

Cosa succede?

Ciclo di Elaborazione

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

51

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10        4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 12

Scorriamo l'array delle valutazioni

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

52

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 13
Incrementiamo l'array delle frequenze

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

53

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 13
Incrementiamo l'array delle frequenze

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

54

IMPORTANTE: Usiamo il valore di
un array come indice di un altro
array!

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 13
Incrementiamo l'array delle frequenze

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

55

IMPORTANTE: Usiamo il valore di
un array come indice di un altro
array!

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10    4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 13
Incrementiamo l'array delle frequenze

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

56

(Simulazione, al ciclo 1)
Answer = 0
Responses[answer]-1 = 2
Frequency[2]++ = 1;

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10        4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

Riga 16-18
Stampiamo i valori

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

57

Utilizzo Avanzato degli Array

```

1 #include "stdio.h"
2
3 gcc version 4.6.3
4
5 Rating: 1      Number: 5
6 Rating: 2      Number: 7
7 Rating: 3      Number: 2
8 Rating: 4      Number: 5
9 Rating: 5      Number: 1
10
11
12
13
14
15
16
17
18
19
20 }
```

Riga 16-18
Stampiamo i valori

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

58

Utilizzo Avanzato degli Array

Anche in questo esempio si conferma la classica struttura degli algoritmi: **fase di inizializzazione (righe 8-10)** **fase di elaborazione dei dati (righe 12-14)** **fase di visualizzazione dell'output (righe 16-18)**

```

1 #include "stdio.h"
2 #define RESPONSE_SIZE 20
3 #define FREQUENCY_SIZE 5
4
5 int main(void) {
6     int rating = 0; // variabili contatore
7     int answer = 0;
8     int frequency[FREQUENCY_SIZE] = {0}; // inizializziamo il vettore
9     int responses[RESPONSE_SIZE] = {3, 1, 2, 4, 2, 3, 1, 1, 5, 2, 2, 2, 1, 4,
10        4, 2, 2, 4, 1, 4};
11
12     for(answer = 0; answer < RESPONSE_SIZE; answer++) {
13         frequency[ responses[answer]-1 ]++;
14     }
15
16     for(rating = 0; rating < FREQUENCY_SIZE; rating++) {
17         printf("Rating: %d \t Number: %d \n", rating+1, frequency[rating]);
18     }
19     return 0;
20 }
```

11/03/19

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

59

Array Multidimensionali (Matrici)

Veronica Rossano – Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

60

Array Multidimensionali (Matrici)

- Cosa è un array multidimensionale?
 - Tabelle con righe e colonne ($m \times n$ array)
 - Come le matrici: specificare le righe, poi le colonne
- Per riferirsi a un elemento, specificare
 - Nome dell'array + Riga + Colonna
- Formato
 - nome[riga][colonna]
 - array di $M \times N$ elementi di nome c:
 - $c[0][0], c[0][1] \dots c[M-1][0], c[M-1][1] \dots c[M-1][N-1]$

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

61

Array Multidimensionali (Matrici)

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

62

Array Multidimensionali (Matrici)

- Inizializzazione
 - `int b[2][2] = {{1, 2}, {3, 4}};`
 - Inizializzatori raggruppati per righe tra parentesi graffe
 - Se non sufficienti, gli elementi non specificati sono settati a zero
 - `int b[2][2] = {{1}, {3, 4}}; → {{1, 0}, {3, 4}}`
- Referenziazione degli elementi
 - Specificare la riga, poi la colonna
 - `printf("%d", b[0][1]);`

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

63

Utilizzo delle Matrici

```

1 #include "stdio.h"
2 [...]
3 #define RIGHE 10 // righe della matrice
4 #define COLONNE 10 // colonne della matrice
5 #define MAX 100
6 #define MIN 1
7
8 int main(void) {
9     int seed = time(NULL); // generazione random dei valori
10    srand(seed);
11    int matrix[RIGHE][COLONNE] = {{0},{0}}; // dichiaro la matrice
12
13    for(int i=0; i<RIGHE; i++) {
14        for(int j=0; j<COLONNE; j++) {
15            matrix[i][j] = rand() % MAX + MIN;
16        }
17    }
  
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

64

Utilizzo delle Matrici

```

1 #include "stdio.h"
2 [...]
3 #define RIGHE 10 // righe della matrice
4 #define COLONNE 10 // colonne della matrice
5 #define MAX 100
6 #define MIN 1
7
8 int main(void) {
9     int seed = time(NULL); // generazione random dei valori
10    srand(seed);
11    int matrix[RIGHE][COLONNE] = {{0},{0}}; // dichiaro la matrice
12
13    for(int i=0; i<RIGHE; i++) {
14        for(int j=0; j<COLONNE; j++) {
15            matrix[i][j] = rand() % MAX + MIN;
16        }
17    }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

65

Riga 4-5
Dichiaro le costanti. Tutti i numeri nel codice devono diventare delle costanti!

Utilizzo delle Matrici

```

1 #include "stdio.h"
2 [...]
3 #define RIGHE 10 // righe della matrice
4 #define COLONNE 10 // colonne della matrice
5 #define MAX 100
6 #define MIN 1
7
8 int main(void) {
9     int seed = time(NULL); // generazione random dei valori
10    srand(seed);
11    int matrix[RIGHE][COLONNE] = {{0},{0}}; // dichiaro la matrice
12
13    for(int i=0; i<RIGHE; i++) {
14        for(int j=0; j<COLONNE; j++) {
15            matrix[i][j] = rand() % MAX + MIN;
16        }
17    }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

66

Riga 11
Dichiaro la matrice e la inizializzo

Utilizzo delle Matrici

```

1 #include "stdio.h"
2 [...]
3 #define RIGHE 10 // righe della matrice
4 #define COLONNE 10 // colonne della matrice
5 #define MAX 100
6 #define MIN 1
7
8 int main(void) {
9     int seed = time(NULL); // generazione random dei valori
10    srand(seed);
11    int matrix[RIGHE][COLONNE] = {{0},{0}}; // dichiaro la matrice
12
13    for(int i=0; i<RIGHE; i++) {
14        for(int j=0; j<COLONNE; j++) {
15            matrix[i][j] = rand() % MAX + MIN;
16        }
17    }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

67

Riga 13-17
Generazione random di valori nella matrice

Utilizzo delle Matrici

```

1 #include "stdio.h"
2 [...]
3 #define RIGHE 10 // righe della matrice
4 #define COLONNE 10 // colonne della matrice
5 #define MAX 100
6 #define MIN 1
7
8 int main(void) {
9     int seed = time(NULL); // generazione random dei valori
10    srand(seed);
11    int matrix[RIGHE][COLONNE] = {{0},{0}}; // dichiaro la matrice
12
13    for(int i=0; i<RIGHE; i++) {
14        for(int j=0; j<COLONNE; j++) {
15            matrix[i][j] = rand() % MAX + MIN;
16        }
17    }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

68

L'uso delle matrici prevede dei DOPPI cicli. Il primo scandisce le righe, il secondo scandisce le colonne

Utilizzo delle Matrici (cont.)

```

18     for(int i=0; i<RIGHE; i++) {
19         for(int j=0; j<COLONNE; j++) {
20             printf("%d\t", matrix[i][j]);
21         }
22     }
23 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

69

Utilizzo delle Matrici (cont.)

Riga 18-22
 Stampa dell'output

```

18     for(int i=0; i<RIGHE; i++) {
19         for(int j=0; j<COLONNE; j++) {
20             printf("%d\t", matrix[i][j]);
21         }
22     }
23 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

70

Utilizzo delle Matrici (cont.)

Riga 18-22
 Stampa dell'output

```

18     for(int i=0; i<RIGHE; i++) {
19         for(int j=0; j<COLONNE; j++) {
20             printf("%d\t", matrix[i][j]);
21         }
22     }
23 }
```



11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

71

Utilizzo delle Matrici (cont.)

Riga 18-22
 Stampa dell'output

```

18     for(int i=0; i<RIGHE; i++) {
19         for(int j=0; j<COLONNE; j++) {
20             printf("%d\t", matrix[i][j]);
21         }
22     }
23 }
```



Serve ad andare a capo quando abbiamo stampato esattamente «COLONNE» valori (in questo caso 10)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

72

Utilizzo delle Matrici (cont.)

Riga 18-22
Stampa dell'output

```
18     for(int i=0; i<RIGHE; i++) {
19         for(j=0; j<COLONNE; j++) {
20             printf("%d\t", matrix[i][j]);
21         }
22     }
23 }
```

```
gcc version 4.6.3
5   47  31  98  3   8   31  93  52  89
7   61  39  93  68  5   76  8   45  83
17  68  29  11  2   38  2   94  83  72
13  63  33  13  26  72  69  55  56  11
86  23  23  24  15  83  88  98  90  25
34  59  36  61  21  17  50  93  83  84
28  53  21  63  5   56  35  17  28  58
27  39  72  2   15  86  36  46  28  25
70  11  75  57  23  56  98  24  78  28
7   23  100  28  49  36  83  71  52  88
```

Serve ad andare a capo quando abbiamo stampato esattamente «COLONNE» valori (in questo caso 10)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

73

Problema 2.2

Scrivere un programma che acquisisca le calorie assunte in una settimana da un insieme di 5 individui e ne calcoli la media (per ogni individuo e complessiva).
Requisiti: utilizzare una matrice

Input?

Output?

Quale tipologia di istruzioni ci serve?

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

74

Domande?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

81

Strutture (**struct**)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

82

(un passo indietro)

- A cosa servono i tipi di dato?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

81

(un passo indietro)

- A cosa servono i tipi di dato?

- A modellare **le variabili** che caratterizzano **il problema** che vogliamo risolvere
- Alcune variabili sono descrivibili attraverso numeri “interi” (es. **età**)
- Alcune variabili sono descrivibili attraverso numeri con la virgola (es. **peso**, **stipendio**, **BMI**, etc.)
- Alcune variabili sono descrivibili in forma “testuale” (es. **nome**, **cognome**)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

84

(un passo indietro)

- A cosa servono i tipi di dato?

- A modellare **le variabili** che caratterizzano **il problema** che vogliamo risolvere
- Alcune variabili sono descrivibili attraverso numeri “interi” (es. **età**)
- Alcune variabili sono descrivibili attraverso numeri con la virgola (es. **peso**, **stipendio**, **BMI**, etc.)
- Alcune variabili sono descrivibili in forma “testuale” (es. **nome**, **cognome**)

- Come facciamo a rappresentare delle variabili **di tipo più complesso**?
 - **Esempi:** una **data** ha due elementi numerici (**giorno** e **anno**) e un elemento testuale (**il mese**)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

85

(un passo indietro)

- A cosa servono i tipi di dato?

- A modellare **le variabili** che caratterizzano **il problema** che vogliamo risolvere
- Alcune variabili sono descrivibili attraverso numeri “interi” (es. **età**)
- Alcune variabili sono descrivibili attraverso numeri con la virgola (es. **peso**, **stipendio**, **BMI**, etc.)
- Alcune variabili sono descrivibili in forma “testuale” (es. **nome**, **cognome**)

- Come facciamo a rappresentare delle variabili **di tipo più complesso**?
 - **Esempi:** una **data** ha due elementi numerici (**giorno** e **anno**) e un elemento testuale (**il mese**)
 - **Esempi:** un **libro** ha svariati elementi che lo descrivono, alcuni testuali (autore, titolo, editore, genere, etc.) altri di tipo numerico (costo, numero di pagine, etc.)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

86

Struct

- Cosa è una struct?

- Aggregazione di dati che può contenere elementi di tipo **eterogeneo**
- **Conoscete altre aggregazioni di dati?**

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

87

Struct

- Cosa è una struct?

- Aggregazione di dati che può contenere elementi di tipo **eterogeneo**
 - Eterogeneo = diversi tra loro
 - Una struct può gestire **elementi diversi tra loro**, quindi è utile a rappresentare tipologie di entità (oggetti, persone, etc.) più complesse
 - Si parla di "aggregati" di dati

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

89

Struct

- Cosa è una struct?

- Aggregazione di dati che può contenere elementi di tipo **eterogeneo**
- **Conoscete altre aggregazioni di dati?**
- Gli array! Qual è la differenza?

Recap: Array

- **Cosa è un array?**
 - Gruppo di locazioni di memoria consecutive
 - Stesso nome e tipo
 - Modello **elementi di tipo omogeneo**
 - Per riferirsi a un elemento, specificare
 - Nome dell'array + Posizione

di questo array	hanno lo stesso
c[0]	14
c[1]	31
c[2]	3
c[3]	64
c[4]	78
c[5]	32
c[6]	146
c[7]	6

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

88

Struct

- Cosa è una struct?

- Aggregazione di dati che può contenere elementi di tipo **eterogeneo**
 - Eterogeneo = diversi tra loro
 - Una struct può gestire **elementi diversi tra loro**, quindi è utile a rappresentare tipologie di entità (oggetti, persone, etc.) più complesse
 - Si parla di "aggregati" di dati
- **Esempi**
 - **Persona** (nome, cognome, sesso, data di nascita, codice fiscale, indirizzo, gruppo sanguigno)
 - **Esame** (nome, votazione, data)
 - **Carta da gioco** (seme, numero)
 - **Libro** (titolo, autore, genere, numero di pagine, anno di pubblicazione, editore)
 - ... etc.

Ogni elemento di una **struct** si dice **membro**. Ogni membro ha un **tipo**.

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

90

Struct

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9 // Dichiaro una struct di tipo persona
10 * struct persona {
11     char *nome;
12     char *cognome;
13     int eta;
14     char sesso;
15 } p1; // dichiaro una variabile p1 di quel tipo
16
17 }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

11/03/19 91

Struct

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9 // Dichiaro una struct di tipo persona
10 * struct persona {
11     char *nome;
12     char *cognome;
13     int eta;
14     char sesso;
15 } p1; // dichiaro una variabile p1 di quel tipo
16
17 }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

11/03/19 92

Struct

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9 // Dichiaro una struct di tipo persona
10 * struct persona {
11     char *nome;
12     char *cognome;
13     int eta;
14     char sesso;
15 } p1; // dichiaro una variabile p1 di quel tipo
16
17 }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

11/03/19 93

Struct

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9 // Dichiaro una struct di tipo persona
10 * struct persona {
11     char *nome;
12     char *cognome;
13     int eta;
14     char sesso;
15 } p1; // dichiaro una variabile p1 di quel tipo
16
17 }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

11/03/19 94

Struct

Dichiaro una
variabile di
quel tipo
(non dimenticate il
punto e virgola)

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9     // Dichiaro una struct di tipo persona
10    struct persona {
11        char *nome;
12        char *cognome;
13        int eta;
14        char sesso;
15    } p1; // dichiaro una variabile p1 di quel tipo
16
17 }

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

95

Struct

Dichiaro una
variabile di
quel tipo
(non dimenticate il
punto e virgola)

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9     // Dichiaro una struct di tipo persona
10    struct persona {
11        char *nome;
12        char *cognome;
13        int eta;
14        char sesso;
15    } p1; // dichiaro una variabile p1 di quel tipo
16
17 }

```

Ho dichiarato
una variabile **p1** di tipo
«persona»

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

96

Attraverso le **struct**
possiamo «creare» nuovi di tipi di dato

Più complessi dei tipi di dato primitivi del C
(come int, float, char, etc.)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

97

typedef

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9     // Dichiaro una struct di tipo "data"
10    typedef struct {
11        int giorno;
12        char mese[15];
13        int anno;
14    } data ;
15
16    data d1;
17    data d2;
18

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

98

typedef

L'istruzione **typedef** avvisa il compilatore che stiamo definendo un nuovo «tipo» di dato

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9 // Dichiavo una struct di tipo "data"
10 typedef struct {
11     int giorno;
12     char mese[15];
13     int anno;
14 } data ;
15
16 data d1;
17 data d2;
18

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

99

typedef

L'istruzione **typedef** avvisa il compilatore che stiamo definendo un nuovo «tipo» di dato

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9 // Dichiavo una struct di tipo "data"
10 typedef struct {
11     int giorno;
12     char mese[15];
13     int anno;
14 } data ;
15
16 data d1;
17 data d2;
18

```

100

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

Parola chiave **struct**

Nome
della struct

typedef

Una volta definito il nuovo tipo di dato posso creare nuovi elementi di quel tipo, come se fosse una qualsiasi variabile.

```

1 // Laboratorio di Informatica
2 // Informatica - TPS - 2017/2018
3 // docente: Cataldo Musto
4
5 #include <stdio.h>
6
7 int main() {
8
9 // Dichiavo una struct di tipo "data"
10 typedef struct {
11     int giorno;
12     char mese[15];
13     int anno;
14 } data ;
15
16 data d1;
17 data d2;
18

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

101

Come facciamo ad accedere ai singoli elementi di una **struct** (es. il giorno del mese o il nome della persona)?

Accedere = leggere / scrivere

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

102

Per accedere agli elementi di una **struct** (stampare o acquisire il valore) si utilizza il cosiddetto «operatore punto»

Sintassi: **nomeVariabile.nomeCampo**

esempio: **d1.giorno**

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

103

Una struct si inizializza come i vettori, mettendo **tra parentesi graffe** i valori dei **singoli membri**

```

16 // Inizializzo e stampo una nuova variabile
17 data d1 = {22, "Marzo", 2017};
18 printf("Oggi: %d %s %d", d1.giorno, d1.mese, d1.anno);
19
20 // Dichiaro una nuova variabile e acquisisco i valori
21 data d2;
22
23 // Memorizza i valori della seconda variabile
24 printf("\nInserisci nuova data: ");
25 scanf("%d %s %d", &d2.giorno, &d2.mese, &d2.anno);
26
27 // Stampa i valori: IMPORTANTE, non c'è nessun controllo
28 printf("Oggi: %d %s %d", d2.giorno, d2.mese, d2.anno);
29 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

104

Accesso a una struct

Utilizzo l'operatore «punto» per stampare i valori dei singoli campi

```

16 // Inizializzo e stampo una nuova variabile
17 data d1 = {22, "Marzo", 2017};
18 printf("Oggi: %d %s %d", d1.giorno, d1.mese, d1.anno);
19
20 // Dichiaro una nuova variabile e acquisisco i valori
21 data d2;
22
23 // Memorizza i valori della seconda variabile
24 printf("\nInserisci nuova data: ");
25 scanf("%d %s %d", &d2.giorno, &d2.mese, &d2.anno);
26
27 // Stampa i valori: IMPORTANTE, non c'è nessun controllo
28 printf("Oggi: %d %s %d", d2.giorno, d2.mese, d2.anno);
29 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

105

Accesso a una struct

Utilizzo l'operatore «punto» per acquisire i valori dei singoli campi

```

16 // Inizializzo e stampo una nuova variabile
17 data d1 = {22, "Marzo", 2017};
18 printf("Oggi: %d %s %d", d1.giorno, d1.mese, d1.anno);
19
20 // Dichiaro una nuova variabile e acquisisco i valori
21 data d2;
22
23 // Memorizza i valori della seconda variabile
24 printf("\nInserisci nuova data: ");
25 scanf("%d %s %d", &d2.giorno, &d2.mese, &d2.anno);
26
27 // Stampa i valori: IMPORTANTE, non c'è nessun controllo
28 printf("Oggi: %d %s %d", d2.giorno, d2.mese, d2.anno);
29 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

106

Una volta definito un nuovo tipo di dato attraverso le **struct**, posso usarlo anche per creare degli **array** di quel tipo di dato

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

107

Una volta definito un nuovo tipo di dato attraverso le **struct**, posso usarlo anche per creare degli **array** di quel tipo di dato

esempio: supponiamo di voler creare una variabile di tipo «calendario».

Come lo descrivo?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

108

Una volta definito un nuovo tipo di dato attraverso le **struct**, posso usarlo anche per creare degli **array** di quel tipo di dato

**esempio: definisco struct data
data d1 → variabile di tipo data
data cal [365] → array di date**

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

109

Una volta definito un nuovo tipo di dato attraverso le **struct**, posso usarlo anche per creare degli **array** di quel tipo di dato

**esempio: definisco struct data
data d1 → variabile di tipo data
data cal [365] → array di date**

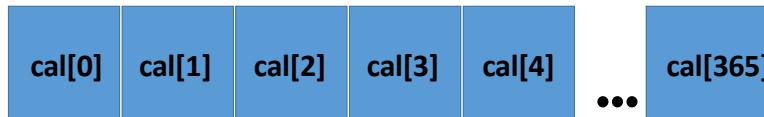


11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

110

esempio: definisco **struct data**
data d1 → variabile di tipo data
data cal[365] → array di date



Ciascun **cal[i]** è una **struct** di tipo «**data**» quindi posso usare l'operatore punto per accedere ai suoi elementi.

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

111

esempio: definisco **struct data**
data d1 → variabile di tipo data
data cal[365] → array di date



Ciascun **cal[i]** è una **struct** di tipo «**data**» quindi posso usare l'operatore punto per accedere ai suoi elementi.

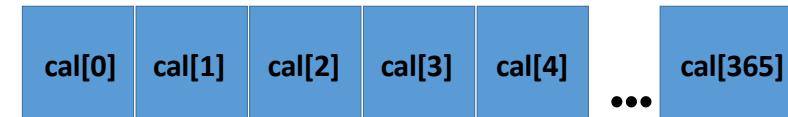
1 Gennaio

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

113

esempio: definisco **struct data**
data d1 → variabile di tipo data
data cal[365] → array di date



Ciascun **cal[i]** è una **struct** di tipo «**data**» quindi posso usare l'operatore punto per accedere ai suoi elementi.

Cosa rappresenta **cal[0]** ?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

112

esempio: definisco **struct data**
data d1 → variabile di tipo data
data cal[365] → array di date



Esempio: **cal[0].giorno = ? , cal[0].mese = ?**
cal[0].anno = ?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
 Laboratorio di Informatica (INF, Track B) – Università degli Studi
 di Bari – A.A. 2018/2019

114

esempio: definisco **struct data**
data d1 → variabile di tipo data
data cal[365] → array di date



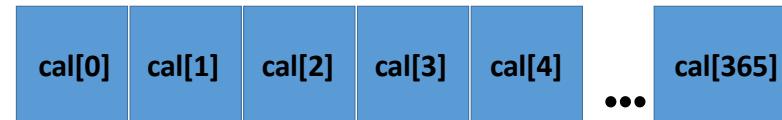
Esempio: `cal[0].giorno = 1 , cal[0].mese = «Gennaio» ,
cal[0].anno = 2018`

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

115

esempio: definisco **struct data**
data d1 → variabile di tipo data
data cal[365] → array di date



Esempio: `cal[0].giorno = 1 , cal[0].mese = «Gennaio» ,
cal[0].anno = 2018
cal[365].giorno = 31 , cal[365].mese = «Dicembre» ,
cal[365].anno = 2018`

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

116

Utilizzo delle **struct**

- **IMPORTANTE**

- I membri delle struct possono essere di due tipi
 - Tipi di dato primitivi (int, float, char, etc.)
 - Tipi di dato complessi (array e altre struct!)
- Partendo da questo principio possiamo usare le **struct** per creare aggregati di dati molto più complessi, ad esempio.

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Utilizzo delle **struct**

- **IMPORTANTE**

- I membri delle struct possono essere di due tipi
 - Tipi di dato primitivi (int, float, char, etc.)
 - Tipi di dato complessi (array e altre struct!)
- Partendo da questo principio possiamo usare le **struct** per creare aggregati di dati molto più complessi, ad esempio.
 - Posso definire un tipo di dato **STUDENTE** come aggregato di:
 - ?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Utilizzo delle struct

• IMPORTANTE

- I membri delle struct possono essere di due tipi
 - Tipi di dato primitivi** (`int`, `float`, `char`, etc.)
 - Tipi di dato complessi** (`array` e altre `struct`!)
- Partendo da questo principio possiamo usare le **struct** per creare aggregati di dati molto più complessi, ad esempio.
 - Posso definire un tipo di dato **STUDENTE** come aggregato di:
 - Nome
 - Cognome
 - Data di Nascita
 - Esami
 - Di che tipo sono i membri?**

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Utilizzo delle struct

• IMPORTANTE

- I membri delle struct possono essere di due tipi
 - Tipi di dato primitivi** (`int`, `float`, `char`, etc.)
 - Tipi di dato complessi** (`array` e altre `struct`!)
- Partendo da questo principio possiamo usare le **struct** per creare aggregati di dati molto più complessi, ad esempio.
 - Posso definire un tipo di dato **STUDENTE** come aggregato di:
 - Nome → `char[10]`
 - Cognome → `char[10]`
 - Data di Nascita
 - Esami
 - Di che tipo sono i membri?**

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Utilizzo delle struct

• IMPORTANTE

- I membri delle struct possono essere di due tipi
 - Tipi di dato primitivi** (`int`, `float`, `char`, etc.)
 - Tipi di dato complessi** (`array` e altre `struct`!)
- Partendo da questo principio possiamo usare le **struct** per creare aggregati di dati molto più complessi, ad esempio.
 - Posso definire un tipo di dato **STUDENTE** come aggregato di:
 - Nome → `char[10]`
 - Cognome → `char[10]`
 - Data di Nascita → `data` (definite in precedenza!)
 - Esami
 - Di che tipo sono i membri?**

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Una struct può
essere membro di
un'altra struct

Utilizzo delle struct

• IMPORTANTE

- I membri delle struct possono essere di due tipi
 - Tipi di dato primitivi** (`int`, `float`, `char`, etc.)
 - Tipi di dato complessi** (`array` e altre `struct`!)
- Partendo da questo principio possiamo usare le **struct** per creare aggregati di dati molto più complessi, ad esempio.
 - Posso definire un tipo di dato **STUDENTE** come aggregato di:
 - Nome → `char[10]`
 - Cognome → `char[10]`
 - Data di Nascita → `data` (definite in precedenza!)
 - Esami → `array di interi` oppure `array di struct`
 - Una struct può essere membro di un'altra struct!**
 - Anche un array può essere membro di una struct**

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Una struct può
essere membro di
un'altra struct

Anche un array può
essere membro di
una struct

Recap - **struct**

- Le **struct** servono a “creare” **nuovi tipi di dato**, **aggregando** tipi di dati più semplici
- Dentro una **struct** possono essere uniti tipologie di dato diverse (es. **char**, **int**, **float**, ma anche **array** e persino altre **struct**)
- Una **struct** diventa una variabile a tutti gli effetti, quindi si può istanziare come **variabile singola**, ma è possibile anche creare degli **array di struct**
- L'accesso** ai membri di una struct avviene con l'operatore “.”, indicando il nome della variabile e poi il nome del campo (es. **persona.nome**)

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

35

Problema 2.2

- Definire una variabile “**archivio**” come un **array di cinque film**.
- Ciascun film deve essere rappresentato come una **struct composta dai seguenti campi: nome del film, genere, data di uscita, durata**
 - Nome e genere sono array di caratteri, la durata è un valore intero. La data deve essere a sua volta una struct composta da mese e anno
- Scrivere un programma che acquisisca i dati in input per ciascuno dei cinque film e li mostri sullo schermo.
- I valori devono essere inseriti dall'utente
 - Aggiungere dei controlli sulla correttezza dei valori inseriti.

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

124

Soluzione 2.2

```

1 #include "stdio.h"
2 #define NUM_FILMS 5 // numero dei films
3 #define MAX_LUNGHEZZA_TITOLO 10 // lunghezza titolo massima
4 #define MAX_LUNGHEZZA_GENERE 10 // lunghezza genere massima
5
6 typedef struct { // struct DATA
7     unsigned char mese;
8     unsigned int anno;
9 } data;
10
11 typedef struct { // struct FILM
12     char titolo[MAX_LUNGHEZZA_TITOLO];
13     char genere[MAX_LUNGHEZZA_GENERE];
14     data uscita; // NOTA: struct come membro di una struct
15     unsigned int durata;
16 } film;

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

126

Soluzione 2.2

Riga 2-4
Definizione delle costanti simboliche
Riga 6-9
Dichiarazione della struct di tipo «data»

```

1 #include "stdio.h"
2 #define NUM_FILMS 5 // numero dei films
3 #define MAX_LUNGHEZZA_TITOLO 10 // lunghezza titolo massima
4 #define MAX_LUNGHEZZA_GENERE 10 // lunghezza genere massima
5
6 typedef struct { // struct DATA
7     unsigned char mese;
8     unsigned int anno;
9 } data;
10
11 typedef struct { // struct PERSONA
12     char titolo[MAX_LUNGHEZZA_TITOLO];
13     char genere[MAX_LUNGHEZZA_GENERE];
14     data uscita; // NOTA: struct come membro di una struct
15     unsigned int durata;
16 } film;

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

127

Soluzione 2.2

```

1 #include "stdio.h"
2 #define NUM_FILMS 5 // numero dei films
3 #define MAX_LUNGHEZZA_TITOLO 10 // lunghezza titolo massima
4 #define MAX_LUNGHEZZA_GENERE 10 // lunghezza genere massima
5
6 typedef struct { // struct DATA
7     unsigned char mese;
8     unsigned int anno;
9 } data;
10
11 typedef struct { // struct PERSONA
12     char titolo[MAX_LUNGHEZZA_TITOLO];
13     char genere[MAX_LUNGHEZZA_GENERE];
14     data uscita; // NOTA: struct come membro di una struct
15     unsigned int durata;
16 } film;

```

Riga 2-4
Definizione delle costanti simboliche
Riga 6-9
Dichiarazione della struct di tipo «data»

unsigned perché i valori non possono mai essere negativi. **char** perché i valori del mese possono variare tra 1 e 12

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

128

Soluzione 2.2

```

1 #include "stdio.h"
2 #define NUM_FILMS 5 // numero dei films
3 #define MAX_LUNGHEZZA_TITOLO 10 // lunghezza titolo massima
4 #define MAX_LUNGHEZZA_GENERE 10 // lunghezza genere massima
5
6 typedef struct { // struct DATA
7     unsigned char mese;
8     unsigned int anno;
9 } data;
10
11 typedef struct { // struct PERSONA
12     char titolo[MAX_LUNGHEZZA_TITOLO];
13     char genere[MAX_LUNGHEZZA_GENERE];
14     data uscita; // NOTA: struct come membro di una struct
15     unsigned int durata;
16 } film;

```

Riga 2-4
Definizione delle costanti simboliche
Riga 6-9
Dichiarazione della struct di tipo «data»

unsigned perché i valori non possono mai essere negativi. **char** perché i valori del mese possono variare tra 1 e 12

IMPORTANTE: L'uso di un tipo di dato appropriato NON RENDE AUTOMATICO IL CONTROLLO DEI VALORI.

«mese» può anche accettare 255 come input, sebbene non sia un valore corretto. Bisogna comunque scrivere del codice per controllare l'input!

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

129

Soluzione 2.2

```

1 #include "stdio.h"
2 #define NUM_FILMS 5 // numero dei films
3 #define MAX_LUNGHEZZA_TITOLO 10 // lunghezza titolo massima
4 #define MAX_LUNGHEZZA_GENERE 10 // lunghezza genere massima
5
6 typedef struct { // struct DATA
7     unsigned char mese;
8     unsigned int anno;
9 } data;
10
11 typedef struct { // struct PERSONA
12     char titolo[MAX_LUNGHEZZA_TITOLO];
13     char genere[MAX_LUNGHEZZA_GENERE];
14     data uscita; // NOTA: struct come membro di una struct
15     unsigned int durata;
16 } film;

```

Riga 11-16
Dichiarazione della struct di tipo «film»

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

130

Soluzione 2.2

```

1 #include "stdio.h"
2 #define NUM_FILMS 5 // numero dei films
3 #define MAX_LUNGHEZZA_TITOLO 10 // lunghezza titolo massima
4 #define MAX_LUNGHEZZA_GENERE 10 // lunghezza genere massima
5
6 typedef struct { // struct DATA
7     unsigned char mese;
8     unsigned int anno;
9 } data;
10
11 typedef struct { // struct PERSONA
12     char titolo[MAX_LUNGHEZZA_TITOLO];
13     char genere[MAX_LUNGHEZZA_GENERE];
14     data uscita; // NOTA: struct come membro di una struct
15     unsigned int durata;
16 } film;

```

Riga 11-16
Dichiarazione della struct di tipo «film»

Una struct può essere membro di un'altra struct!

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

131

Soluzione 2.2 (cont.)

```

17 int main() {
18     film archivio[NUM_FILMS] = {{«», «», {'«', 0}, 0}}; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         printf("film n.%d Inserisci titolo:", i+1);
22         scanf("%15s", archivio[i].titolo); //NOTA: non accetta spazi
23         printf("film n.%d Inserisci genere:", i+1);
24         scanf("%10s", archivio[i].genere); //NOTA: non accetta spazi
25         printf("film n.%d Inserisci mese di pubblicazione:", i+1);
26         scanf("%u", &archivio[i].uscita.mese);
27         printf("film n.%d Inserisci anno di pubblicazione:", i+1);
28         scanf("%u", &archivio[i].uscita.anno);
29         printf("film n.%d Inserisci durata:", i+1);
30         scanf("%u", &archivio[i].durata);
31     }
32     // Ciclo di Stampa
33     for(unsigned int j=0; j<NUM_FILMS; j++) {
34         printf("\n%«\t%«\t%u\n", archivio[j].titolo, archivio[j].genere,
35             archivio[j].uscita.mese, archivio[j].uscita.anno, archivio[j].durata);
36     }
}

```

11/03/19

Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

132

Riga 18
Dichiarazione dell'array di struct di tipo «film»

Soluzione 2.2 (cont.)

```

17 int main() {
18     film archivio[NUM_FILMS]; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         printf("film n.%d Inserisci titolo:", i+1);
22         scanf("%15s", archivio[i].titolo); //NOTA: non accetta spazi
23         printf("film n.%d Inserisci genere:", i+1);
24         scanf("%10s", archivio[i].genere); //NOTA: non accetta spazi
25         printf("film n.%d Inserisci mese di pubblicazione:", i+1);
26         scanf("%u", &archivio[i].uscita.mese);
27         printf("film n.%d Inserisci anno di pubblicazione:", i+1);
28         scanf("%u", &archivio[i].uscita.anno);
29         printf("film n.%d Inserisci durata:", i+1);
30         scanf("%u", &archivio[i].durata);
31     }
32     // Ciclo di Stampa
33     for(unsigned int j=0; j<NUM_FILMS; j++) {
34         printf("\n%«\t%«\t%u\n", archivio[j].titolo, archivio[j].genere,
35             archivio[j].uscita.mese, archivio[j].uscita.anno, archivio[j].durata);
36     }
}

```

11/03/19

Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

133

Soluzione 2.2 (cont.)

Riga 20-30
Acquisizione dell'input

```

17 int main() {
18     film archivio[NUM_FILMS]; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         printf("film n.%d Inserisci titolo:", i+1);
22         scanf("%15s", archivio[i].titolo); //NOTA: non accetta spazi
23         printf("film n.%d Inserisci genere:", i+1);
24         scanf("%10s", archivio[i].genere); //NOTA: non accetta spazi
25         printf("film n.%d Inserisci mese di pubblicazione:", i+1);
26         scanf("%u", &archivio[i].uscita.mese);
27         printf("film n.%d Inserisci anno di pubblicazione:", i+1);
28         scanf("%u", &archivio[i].uscita.anno);
29         printf("film n.%d Inserisci durata:", i+1);
30         scanf("%u", &archivio[i].durata);
31     }
32     // Ciclo di Stampa
33     for(unsigned int j=0; j<NUM_FILMS; j++) {
34         printf("\n%«\t%«\t%u\n", archivio[j].titolo, archivio[j].genere,
35             archivio[j].uscita.mese, archivio[j].uscita.anno, archivio[j].durata);
36     }
}

```

11/03/19

Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

134

Riga 22

Bisogna sempre dichiarare il numero massimo di caratteri che la stringa può leggere ("%15s")

Soluzione 2.2 (cont.)

```

17 int main() {
18     film archivio[NUM_FILMS]; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         printf("film n.%d Inserisci titolo:", i+1);
22         scanf("%15s", archivio[i].titolo); //NOTA: non accetta spazi
23         printf("film n.%d Inserisci genere:", i+1);
24         scanf("%10s", archivio[i].genere); //NOTA: non accetta spazi
25         printf("film n.%d Inserisci mese di pubblicazione:", i+1);
26         scanf("%u", &archivio[i].uscita.mese);
27         printf("film n.%d Inserisci anno di pubblicazione:", i+1);
28         scanf("%u", &archivio[i].uscita.anno);
29         printf("film n.%d Inserisci durata:", i+1);
30         scanf("%u", &archivio[i].durata);
31     }
32     // Ciclo di Stampa
33     for(unsigned int j=0; j<NUM_FILMS; j++) {
34         printf("\n%«\t%«\t%u\n", archivio[j].titolo, archivio[j].genere,
35             archivio[j].uscita.mese, archivio[j].uscita.anno, archivio[j].durata);
36     }
}

```

11/03/19

Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

135

Soluzione 2.2 (cont.)

Riga 26
Quando una **struct** è membro di un'altra **struct**
bisogna utilizzare l'operatore 'punto' due volte

```

17 int main() {
18     film archivio[NUM_FILMS]; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         printf("Film n.%d Inserisci titolo:", i+1);
22         scanf("%15s", &archivio[i].titolo); //NOTA: non accetta spazi
23         printf("Film n.%d Inserisci genere:", i+1);
24         scanf("%10s", &archivio[i].genere); //NOTA: non accetta spazi
25         printf("Film n.%d Inserisci mese di pubblicazione:", i+1);
26         scanf("%c", &archivio[i].uscita.mese);
27         printf("Film n.%d Inserisci anno di pubblicazione:", i+1);
28         scanf("%u", &archivio[i].uscita.anno);
29         printf("Film n.%d Inserisci durata:", i+1);
30         scanf("%u", &archivio[i].durata);
31     }
32     // Ciclo di Stampa
33     for(unsigned int j=0; j<NUM_FILMS; j++) {
34         printf("\n%8s%15s%10s%u%u\n", archivio[j].titolo, archivio[j].genere,
35               archivio[j].uscita.mese, archivio[j].uscita.anno, archivio[j].durata);
36     }
}

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

136

(Esempio) →

Simulazione



```

17 int main() {
18     film archivio[NUM_FILMS]; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         [...]
22         printf("Film n.%d Inserisci mese di pubblicazione:", i+1);
23         scanf("%u", &archivio[i].uscita.mese);
24         printf("Film n.%d Inserisci anno di pubblicazione:", i+1);
25         scanf("%u", &archivio[i].uscita.anno);
26         [...]
27     [...]
28 }
29
30
31
32 [...]

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

137

Simulazione

Se **i==0**, mi riferisco a **archivio[0]**,
che è una variabile di tipo **film**



```

17 int main() {
18     film archivio[NUM_FILMS]; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         [...]
22         printf("Film n.%d Inserisci mese di pubblicazione:", i+1);
23         scanf("%u", &archivio[i].uscita.mese);
24         printf("Film n.%d Inserisci anno di pubblicazione:", i+1);
25         scanf("%u", &archivio[i].uscita.anno);
26         [...]
27     [...]
28 }
29
30
31
32 [...]

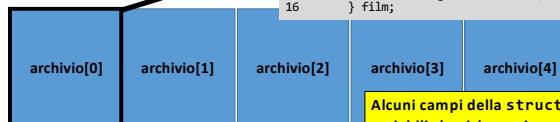
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

138

Simulazione



```

11 typedef struct { // struct PERSONA
12     char titolo[MAX_LUNGHEZZA_TITOLO];
13     char genere[MAX_LUNGHEZZA_GENERE];
14     data uscita;
15 } film;
16

```

Alcuni campi della **struct** sono delle
variabili classiche, e ci accedo
attraverso l'operatore punto
(es. **archivio[0].titolo**)

```

17 int main() {
18     film archivio[NUM_FILMS]; // variabile archivio
19     // Acquisizione input
20     for(unsigned int i=0; i<NUM_FILMS; i++) {
21         [...]
22         printf("Film n.%d Inserisci mese di pubblicazione:", i+1);
23         scanf("%u", &archivio[i].uscita.mese);
24         printf("Film n.%d Inserisci anno di pubblicazione:", i+1);
25         scanf("%u", &archivio[i].uscita.anno);
26         [...]
27     [...]
28 }
29
30
31
32 [...]

```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

139

Simulazione

```

11     typedef struct { // struct PERSONA
12         char titolo[MAX_LUNGHEZZA_TITOLO];
13         char genere[MAX_LUNGHEZZA_GENERE];
14         data uscita;
15         unsigned int durata;
16     } film;
17
18     int main() {
19         film archivio[NUM_FILMS]; // variabile archivio
20         // Acquisizione input
21         for(unsigned int i=0; i<NUM_FILMS; i++) {
22             [...]
23             printf("Film n.%d Inserisci mese di pubblicazione:", i+1);
24             scanf("%u", &archivio[i].uscita.mese);
25             printf("Film n.%d Inserisci anno di pubblicazione:", i+1);
26             scanf("%u", &archivio[i].uscita.anno);
27             [...]
28             [...]
29             [...]
30             [...]
31         }
32         [...]
33     }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 140

**Riga 32-35
Stampa dell'output**

Soluzione 3.1 (cont.)

```

17     int main() {
18         film archivio[NUM_FILMS]; // variabile archivio
19         // Acquisizione input
20         for(unsigned int i=0; i<NUM_FILMS; i++) {
21             printf("Film n.%d Inserisci titolo:", i+1);
22             scanf("%15s", archivio[i].titolo); //NOTA: non accetta spazi
23             printf("Film n.%d Inserisci genere:", i+1);
24             scanf("%10s", archivio[i].genere); //NOTA: non accetta spazi
25             printf("Film n.%d Inserisci mese di pubblicazione:", i+1);
26             scanf("%u", &archivio[i].uscita.mese);
27             printf("Film n.%d Inserisci anno di pubblicazione:", i+1);
28             scanf("%u", &archivio[i].uscita.anno);
29             printf("Film n.%d Inserisci durata:", i+1);
30             scanf("%u", &archivio[i].durata);
31         }
32         // Ciclo di Stampa
33         for(unsigned int j=0; j<NUM_FILMS; j++) {
34             printf("\n%u\t%u\t%u\t%u\n", archivio[j].titolo, archivio[j].genere,
35             archivio[j].uscita.mese, archivio[j].uscita.anno, archivio[j].durata);
36         }

```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 141

struct - memorizzazione

```

struct contact
{
    int nl;
    float n2;
    char st[10];
} d1;

```

I campi di una struttura vengono memorizzati in locazioni di memoria consecutive

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 142

struct - memorizzazione

```

struct contact
{
    int nl;
    float n2;
    char st[10];
} d1;

```

Qual è la dimensione di questa struttura?

I campi di una struttura vengono memorizzati in locazioni di memoria consecutive

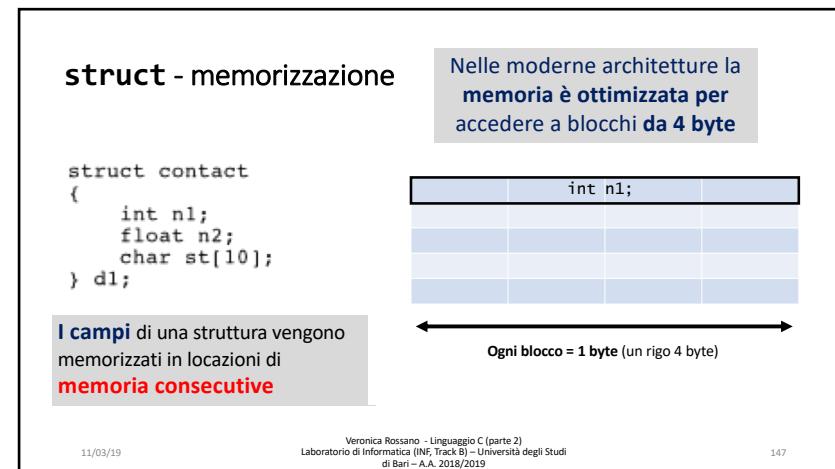
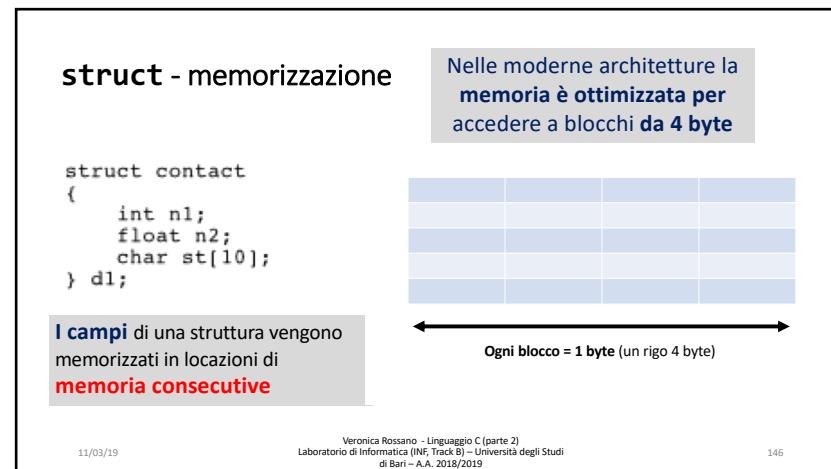
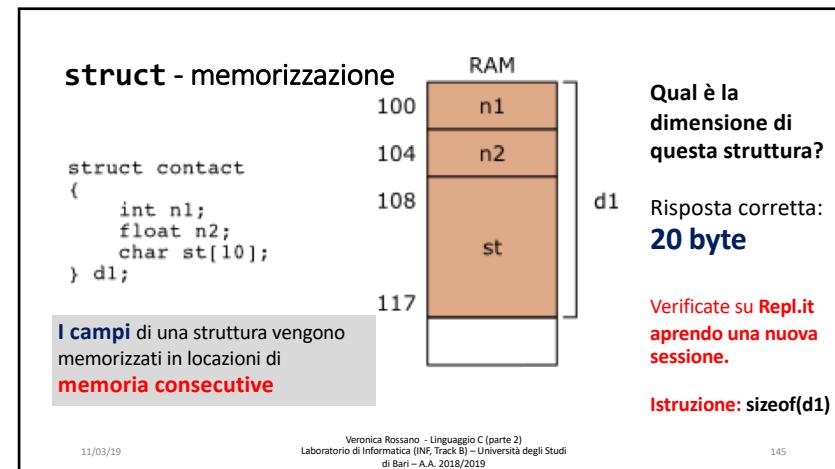
Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 143

(RECAP) Linguaggio C: tipi di dato				
Tipo di dato	Dimensione in Byte (Macchina a 32 bit)	Dimensione in Byte (Macchina a 64 bit)	Range (Macchina a 64 bit)	
char	1	1	-128	127
unsigned char	1	1	0	255
short	2	2	-32768	32767
unsigned short	2	2	0	65535
int	4	4	-2147483648	2147483647
unsigned int	4	4	0	4294967295
long	4	8	-9223372036854775808	9223372036854775808
unsigned long	4	8	0	18446744073709551615
float	4	4
double	8	8
Long double	12	16

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

11/03/19 144



struct - memorizzazione

Nelle moderne architetture la memoria è ottimizzata per accedere a blocchi da 4 byte

```
struct contact
{
    int n1;
    float n2;
    char st[10];
} d1;
```

I campi di una struttura vengono memorizzati in locazioni di memoria consecutive

Ogni blocco = 1 byte (un rigo 4 byte)

struct - memorizzazione

Nelle moderne architetture la memoria è ottimizzata per accedere a blocchi da 4 byte

```
struct contact
{
    int n1;
    float n2;
    char st[10];
} d1;
```

I campi di una struttura vengono memorizzati in locazioni di memoria consecutive

Ogni blocco = 1 byte (un rigo 4 byte)

11/03/19 Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019 148

11/03/19 Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019 149

struct - memorizzazione

Nelle moderne architetture la memoria è ottimizzata per accedere a blocchi da 4 byte

```
struct contact
{
    int n1;
    float n2;
    char st[10];
} d1;
```

I campi di una struttura vengono memorizzati in locazioni di memoria consecutive

Ogni blocco = 1 byte (un rigo 4 byte)

struct - memorizzazione

Per ottimizzare l'accesso alla memoria si aggiungono 2-byte extra (il totale deve essere sempre un multiplo di 4)

Questo fenomeno si chiama padding.

I campi di una struttura vengono memorizzati in locazioni di memoria consecutive

Ogni blocco = 1 byte (un rigo 4 byte)

11/03/19 Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019 150

11/03/19 Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019 151

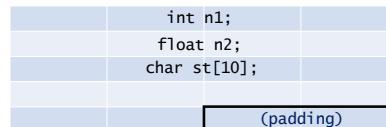
struct - memorizzazione

```
struct contact
{
    int n1;
    float n2;
    char st[10];
} dl;
```

I campi di una struttura vengono memorizzati in locazioni di memoria consecutive

Per ottimizzare l'accesso alla memoria si aggiungono 2-byte extra (il totale deve essere sempre un multiplo di 4)

Questo fenomeno si chiama padding.



Ogni blocco = 1 byte (un rigo 4 byte)

Verifica: dichiarate char st[9] o char st[11]
La struttura occuperà sempre 20 byte.

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

152

Padding

- È una tecnica utilizzata per allineare i dati in memoria

Come?

- Uno o più byte (indirizzi) vuoti vengono inseriti (o lasciati vuoti) tra gli indirizzi di memoria assegnati per gli altri membri della struttura durante l'allocazione della memoria. L'architettura di un processore del computer è tale da poter leggere 1 word (4 byte nel processore a 32 bit) dalla memoria alla volta.
- Per sfruttare questo vantaggio del processore, i dati sono sempre allineati come un pacchetto da 4 byte che porta a inserire indirizzi vuoti tra l'indirizzo di un altro membro.
- Questo porta ad occupare un numero di byte diversi a seconda dell'ordine di dichiarazione.

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

153

Padding

```
#include <stdio.h>
#include <string.h>

/* Below structure1 and structure2 are same.
   They differ only in member's alignment */

struct structure1
{
    int id1;
    int id2;
    char name;
    char c;
    float percentage;
};

struct structure2
{
    int id1;
    char name;
    int id2;
    char c;
    float percentage;
};
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

154

Padding

```
#include <stdio.h>
#include <string.h>

/* Below structure1 and structure2 are same.
   They differ only in member's alignment */

struct structure1
{
    int id1;
    int id2;
    char name;
    char c;
    float percentage;
};

struct structure2
{
    int id1;
    char name;
    int id2;
    char c;
    float percentage;
};
```

Stessa struttura diversa occupazione di memoria!!!

Esecuzione

```
size of structure1 in bytes : 16
size of structure2 in bytes : 20
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi di Bari – A.A. 2018/2019

155

Nota finale

Le variabili di tipo
struct
possono essere
assegnate una
all'altra

```

5  #include <stdio.h>
6
7  int main() {
8  typedef struct {
9    char nome[10];
10   char cognome[10];
11 } persona;
12
13 // Inizializzazione
14 persona p1 = {"Mister", "X"};
15 persona p2 = p1; // istruzione consentita
16
17 printf("\nP1: %s %s", p1.nome, p1.cognome);
18 printf("\nP2: %s %s", p2.nome, p2.cognome);
19 }
```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

156

11/03/19

Nota finale

Le variabili di tipo
struct
possono essere
assegnate una
all'altra

```

5  #include <stdio.h>
6
7  int main() {
8  typedef struct {
9    char nome[10];
10   char cognome[10];
11 } persona;
12
13 // Inizializzazione
14 persona p1 = {"Mister", "X"};
15 persona p2 = p1; // istruzione consentita
16
17 printf("\nP1: %s %s", p1.nome, p1.cognome);
18 printf("\nP2: %s %s", p2.nome, p2.cognome);
19 }
```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

157

11/03/19

Nota finale

Le variabili di tipo
struct
Non possono
essere confrontate

```

1  #include <stdio.h>
2
3  int main() {
4
5  typedef struct{
6    int x;
7    int y;
8  } point;
9
10 point a = {10, 20};
11 point b = {10, 20};
12
13 if(a==b)
14   puts("Equals");
15 else
16   puts("Not equals");
17
18 }
```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

158

11/03/19

Nota finale

Le variabili di tipo
struct
Non possono
essere confrontate

```

1  #include <stdio.h>
2
3  int main() {
4
5  typedef struct{
6    int x;
7    int y;
8  } point;
9
10 point a = {10, 20};
11 point b = {10, 20};
12
13 if(a==b)
14   puts("Equals");
15 else
16   puts("Not equals");
17
18 }
```

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

159

11/03/19

Nota finale

Versione corretta:
bisogna confrontare i
singoli campi della
struct.

```
1 #include <stdio.h>
2
3 int main() {
4
5     typedef struct{
6         int x;
7         int y;
8     } point;
9
10    point a = {10, 20};
11    point b = {10, 20};
12
13    if((a.x == b.x) && (a.y == b.y))
14        puts("Equals");
15    else
16        puts("Not equals");
17
18 }
```

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

160

Domande?

11/03/19

Veronica Rossano - Linguaggio C (parte 2)
Laboratorio di Informatica (INF, Track B) – Università degli Studi
di Bari – A.A. 2018/2019

161