ITIS-LS "Francesco Giordani" Caserta

prof. Ennio Ranucci

a.s. 2020-2021

Astrazione dal linguaggio macchina

Strutture astratte e strutture interne

Memoria RAM 40 indirizzi

Ciascuna cella può contenere 8 bit

1000101			
	00000000	0000101	
01100011	01101001	01100001	01101111
00001010			

Tavola dei simboli

identidficatore	indirizzo base	tipo	
		chiave di interpretazione	dimensione in byte
carVar	A10	char	1
intVar	B12	int16	2
stringVar	A15	string	5

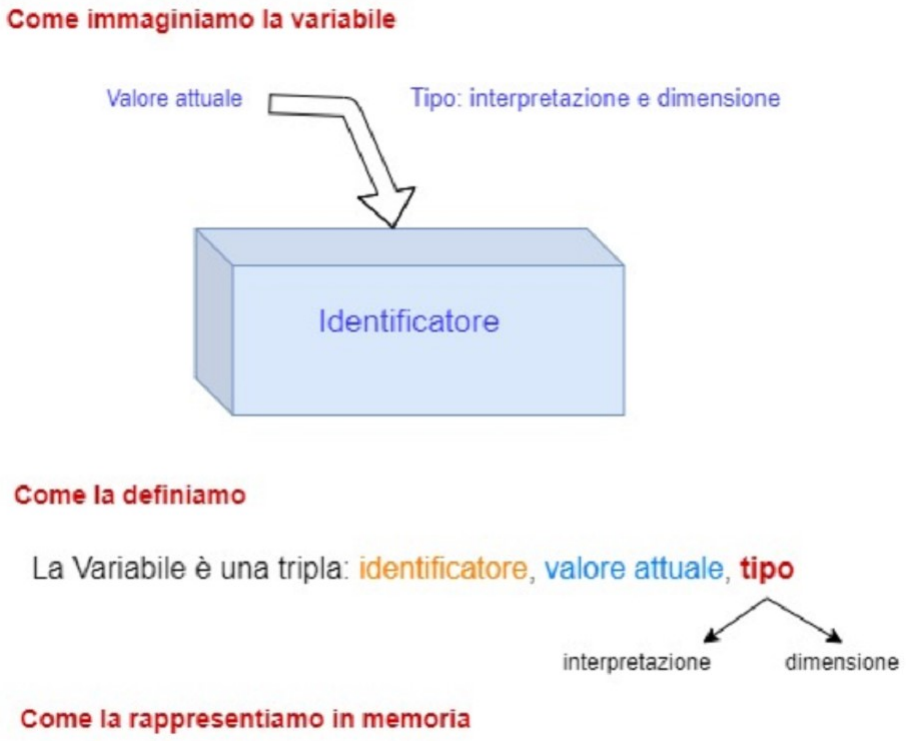
Codice C++

```
char carVar='E';
int intVar=5;
string stringVar="ciao";
```

Codice assembler

```
.model small
.data
intVar  db 5
carVar  db 'E'
stringVar db "ciao"
.code
inizio:
;un indirizzo è segmento:offset
;per usare nel codice solo l'offset senza il segmento
;si scrivono le due prossime istruzioni
  mov ax,@data
  mov ds,ax
  mov ah,4ch          ;uscita dal programma
  int 21h             ;ritorno al sistema operativo
end inizio
```

VARIABILE



Istruzione linguaggio ad alto livello: **int num=3**

1) L'identificatore della variabile è : **num**
2) Il valore attuale assegnato è : **3** (in binario 0000000000000011)
3) Il tipo è : **intero a 16 bit**

RAM organizzata a 8 bit (un indirizzo ogni 8 bit)

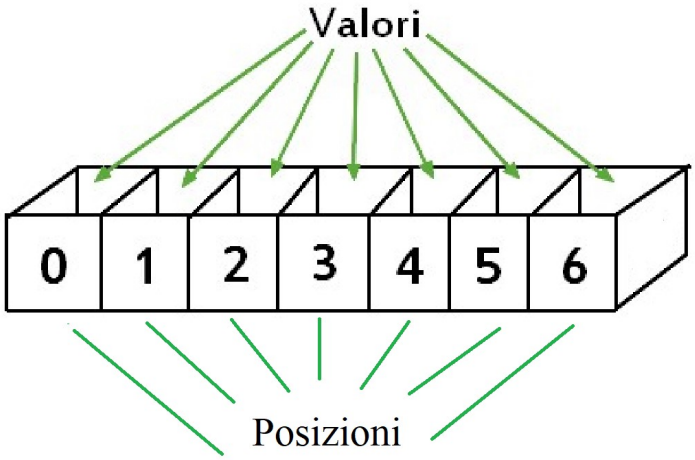
indirizzi	Valore attuale
1AX00	
1AX01	00000000
1AX02	00000011
1AX03	
1AX04	

Il Compilatore costruisce la Tavola dei simboli :

identificatore	Indirizzo base di memoria	Dimensione fisica	tipo
num	1AX01	2 byte	int

VETTORE

Come immaginiamo il vettore



Come definiamo il vettore

Il vettore è una quadrupla: **identificatore**, **insieme di valori attuali**, **tipo**, **dim**

Come rappresentiamo il vettore in memoria (implementazione)

Istruzione linguaggio ad alto livello: **int vet[3] = {1,2,3}**

1) L'identificatore della variabile è : **vet**
2) L'insieme dei valori attuali è : **1,2,3**
3) Il tipo è : **intero a 16 bit**
4) La dimensione è : **3**

RAM

indirizzi	Valore attuale
1AX00	
1AX01	00000000
1AX02	00000011
1AX03	00000000
1AX04	00000001
1AX05	00000000
1AX06	00000010
1AX07	00000000
1AX08	00000011
1AX09	
1AX10	

Il Compilatore costruisce la Tavola dei simboli :

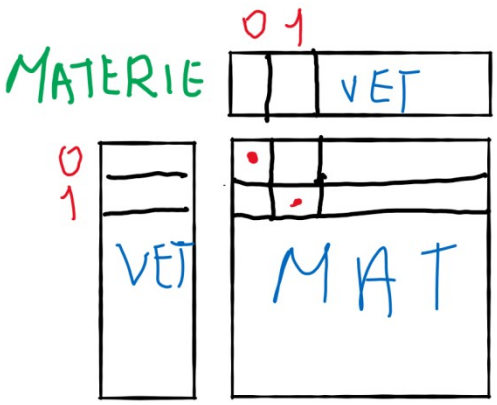
identificatore	Indirizzo base di memoria	Dimensione fisica	tipo
num	1AX01	2 byte	int
vet	1AX03	6 byte	int

Dato l'indice del vettore calcolare la posizione:

posizione= indirizzo base + d*(i-1) // i è l'indice del vettore, d la dimensione di ciascun elemento

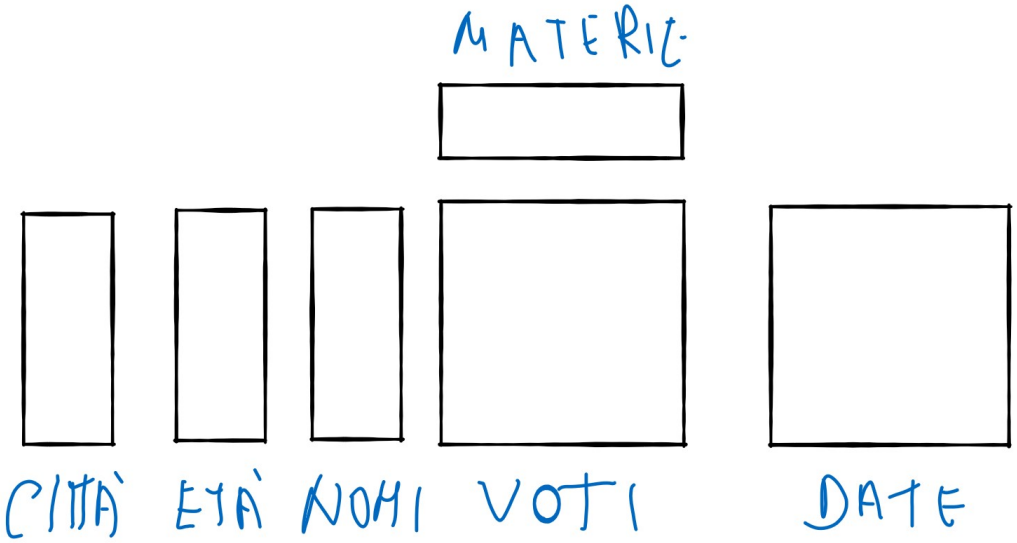
int Vet[3] // posizione=1AX03+2*(3-1) -> 1AX03+4 -> 1AX07

MODELLO DEI DATI



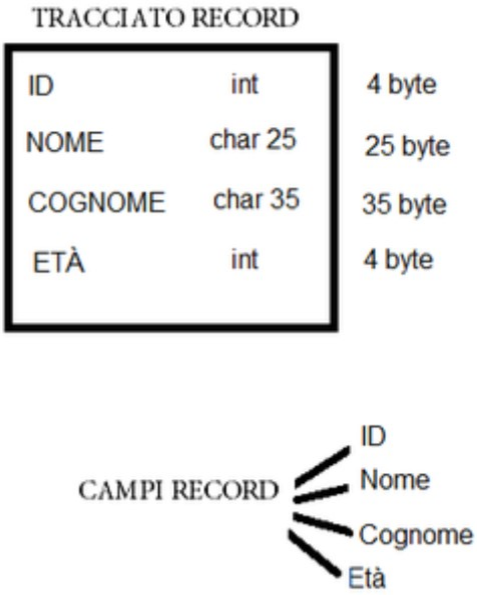
ALUNNI → VETTORE
MATERIE → VETTORE
VOTI → MATRICE

STRUTTURE DATI



RECORD

Come immaginiamo il record



Come definiamo il record

Il vettore è una quadrupla:

Identificatore del record, insieme di identificatori di campo, insieme di tipi, insieme di valori attuali

Come rappresentiamo il record in memoria (implementazione)

In memoria per leggere o scrivere un valore dobbiamo sommare all'indirizzo base (corrispondente all'identificatore del record) l'indirizzo del campo (corrispondente all'identificatore del campo).

Istruzione linguaggio ad alto livello:

```
struct persona
  {
   string nome;
  string cognome;
   int eta;
}
```

identificatore	Indirizzo base di memoria	Dimensione fisica	tipo
num	1AX01	2 byte	int
vet	1AX03	6 byte	int
personaRec	1AX09	514 byte	struct

Dato il nome del campo e il nome del record calcolare la posizione:

posizione= indirizzo base del record + dimensione dei campi precedenti

personaRec.cognome -> posizione=1AX09+ 256

personaRec.eta -> posizione=1AX09+256+256