Corso Programmazione 2011-2012

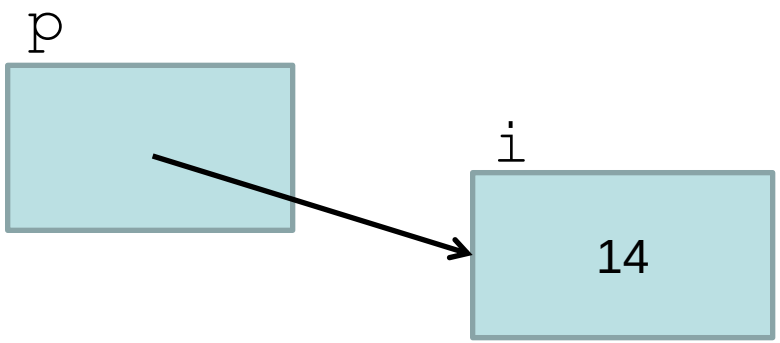
(docente)
Fabio Aiolli
E-mail: aiolli@math.unipd.it
Web: www.math.unipd.it/~aiolli

Dipartimento di Matematica Pura ed Applicata
Torre Archimede, Via Trieste 63

Puntatori, Array, Stringhe

Puntatori

- Un puntatore è una variabile che contiene un indirizzo di memoria
- Dichiarazione di un puntatore
  - `TipoBase *NomePunt;`
  - Esempio: `int *p;`
- Inizializzazione di un puntatore
  - `p=0;`
  - `p=&i;` (se p è un puntatore ad int, i deve essere di tipo int)



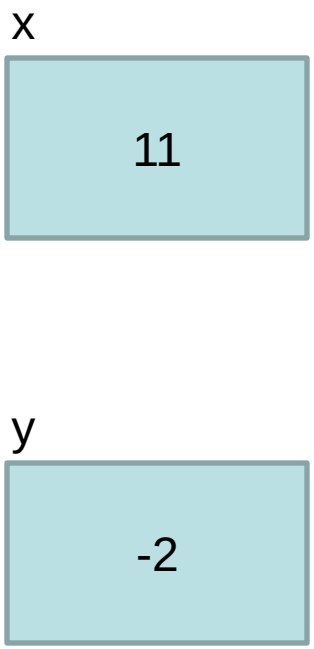

```
i = 14;
p = &i;
```

Per accedere al valore della variabile i posso usare:

```
i
*p
```

Chiamata x Indirizzo

```
Swap(int *p, int *q) {
  …
}
main() {
  int x=11;
  int y=-2;
  Swap(&x,&y);
}
```



Array

- Un Array è un puntatore costante che punta all'inizio di una serie di locazioni di memoria contigue che contengono tutti valori dello stesso tipo
- Puo' essere pensato come un contenitore composto da una serie di scomparti tanti quanti sono I dati che vi si vogliono immagazzinare
- Dichiarazione di un Array
  - `TipoBase NomeArray[DimArray];` (n.b. +lunghezza)
  - Esempio: `int A[10];`
- Inizializzazione di un Array (solo insieme alla dichiarazione)
  - `int a[] = {2,3,5,-7};`

- Indici per accedere agli elementi
  - `a[2]=15`
  - `int b=a[0]+a[1]`
- Tipico uso all'interno di cicli
  - (ricerca del massimo)
  - (lettura n valori)

Aritmetica dei Puntatori

- Operatori ammessi per i puntatori
  - `+,-` (anche per gli array)
  - `++,--` (non per gli array, perché?)
- Quando un operatore e' applicato ad un puntatore `p` di tipo b, `p+5` significa il quinto elemento successivo a `p`, `p+1` significa l'elemento successivo, ecc..
- Cosa fa: `*s++=3` ?

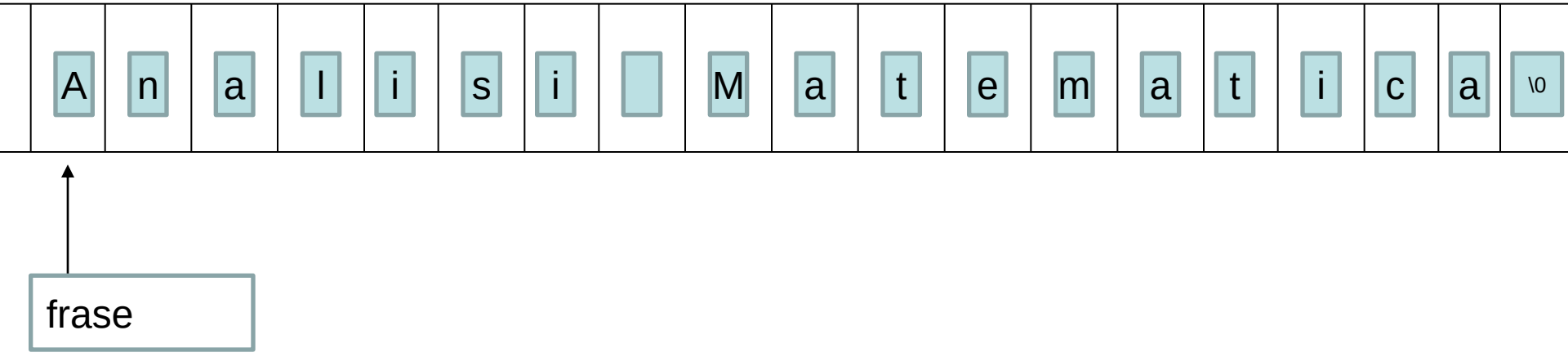
Array e Puntatori

- `char buf[100];`
- `char *s;`
- `s=&buf[0]; oppure s=buf;`

- `Per accedere ad un elemento:`
- `buf[7] oppure *(s+7)`

Stringhe

- Array di caratteri terminato dal carattere null `(\0)`
- `char frase[]="Analisi Matematica"`
- `(rappresentazione in mem)`



Stringhe: Funzioni Libreria

- `strcpy, strcat, strcmp, strlen`
- `atoi, atof`
- Molto importante che le stringhe manipolate da queste funzioni siano correttamente allocate e contengano il carattere di terminazione!

Matrici e Array Multidimensionali

- Dichiarazione delle matrici
  - int M[3][5] alloca spazio per contenere una matrice di 3x5 oggetti di tipo int.
  - Si può generalizzare al caso di più dimensioni
    - (p.e. int H[4][6][8] o int H[9][1][3][6],..)

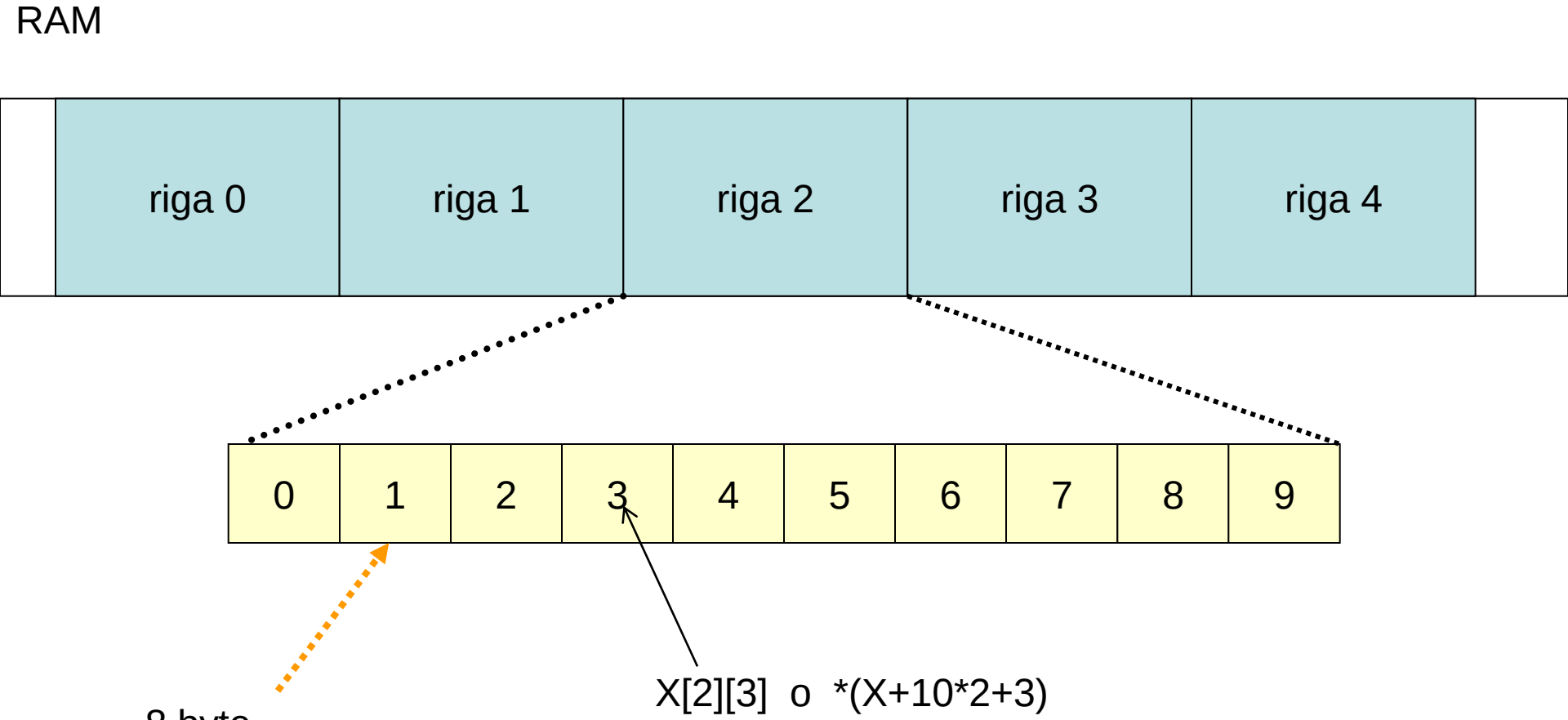
- Come viene effettivamente rappresentato un array (multidimensionale) in memoria?
- Sempre come array + meccanismo del Mapping della memoria

- A[11] equivale ad *(a+11) nel senso che il compilatore traduce le parentesi quadre a tempo di compilazione
- Nota che questa traduzione non necessita della dimensione dell'array!

`int M[2][3][5]`

`M[i][j][k]` e' uguale a `*(M+3*5*i+5*j+k)`

Rappresentazione di matrici come array : double X[5][10]



Matrici

- Gli array (anche quelli multidimensionali) possono essere passati alle funzioni. Nella definizione di funzione la prima dimensione e' opzionale! Le altre sono obbligatorie.
  - Es. void Funz_Mat(int mat[][3][5])
  - Posso passare una qualsiasi matrice tridimensionale di tipo m[x][3][5] con x qualsiasi,
  - p.e. int m[10][3][5]; Funz_Mat(m).
- Gli array (e tanto meno le matrici) non si possono utilizzare come valori di ritorno!