Programmazione — Variabili e tipi elementari

Eugenio Omodeo

UNIVERSITÀ
DEGLI STUDI DI TRIESTE

Trieste, 22.09.2015

Cos'è un problema ? — Esempio Quattro quadrati

- Ci viene dato un ottetto di numeri

$$\mathbf{g} \quad \mathbf{h} \quad \mathbf{i} \quad \mathbf{j} \quad \mathbf{k} \quad \boldsymbol{\ell} \quad \mathbf{m} \quad \mathbf{n}$$

Cos'è un problema ? — Esempio Quattro quadrati

- Ci viene dato un ottetto di numeri

$$\mathbf{g} \quad \mathbf{h} \quad \mathbf{i} \quad \mathbf{j} \quad \mathbf{k} \quad \boldsymbol{\ell} \quad \mathbf{m} \quad \mathbf{n}$$

- e ci vengono chiesti dei numeri

$$\mathbf{a} \quad \mathbf{b} \quad \mathbf{c} \quad \mathbf{d}$$

Cos'è un problema ? — Esempio Quattro quadrati

- Ci viene dato un ottetto di numeri

$$\mathbf{g} \quad \mathbf{h} \quad \mathbf{i} \quad \mathbf{j} \quad \mathbf{k} \quad \boldsymbol{\ell} \quad \mathbf{m} \quad \mathbf{n}$$

- e ci vengono chiesti dei numeri

$$\mathbf{a} \quad \mathbf{b} \quad \mathbf{c} \quad \mathbf{d}$$

- tali che risulti

$$\mathbf{a}^2 + \mathbf{b}^2 + \mathbf{c}^2 + \mathbf{d}^2 =$$
$$(\mathbf{g}^2 + \mathbf{h}^2 + \mathbf{i}^2 + \mathbf{j}^2)\,(\mathbf{k}^2 + \boldsymbol{\ell}^2 + \mathbf{m}^2 + \mathbf{n}^2)$$

Cos'è un problema ? — Esempio Quattro quadrati

Ad es.

g	h	i	j
0	2	0	0
0	1	1	1
k	ℓ	m	n

$\Rightarrow$

?	?	?	?
a	b	c	d

1

Cos'è un problema ? — Esempio Quattro quadrati

Ad es.[1]

g	h	i	j
0	2	0	0
0	1	1	1
k	ℓ	m	n

$\Vert \Rightarrow$

3	1	1	1
a	b	c	d

[1] Trovare altre soluzioni è un esercizio per voi...

Come lo si distingue da un rompicapo ? — Es. 8 regine

C'è una differenza importante fra il **problema dei quattro quadrati** e molti classici della 'matematica ricreativa',

quale ad esempio il

rompicapo delle 8 regine

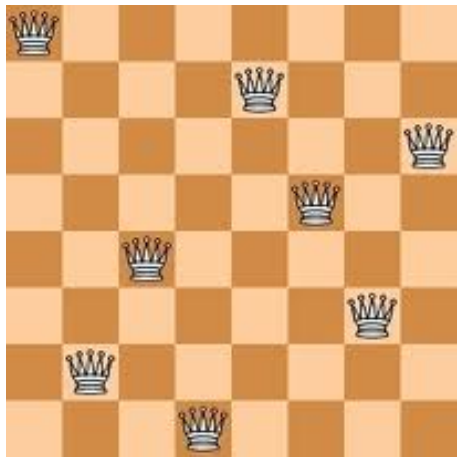
Come lo si distingue da un rompicapo ? — Es. 8 regine

C'è una differenza importante fra il **problema dei quattro quadrati** e molti classici della 'matematica ricreativa',

quale ad esempio il

rompicapo delle 8 regine

(da collocare sulla scacchiera in modo che nessuna regina ne tenga altre sotto scacco)



È un problema con infinite *istanze* !

Ogni ottetto di numeri ci pone un 'esemplare'[2] diverso del problema...

[2] In inglese 'instance'

È un problema con infinite *istanze* !

Ogni ottetto di numeri ci pone un 'esemplare'[2] diverso del problema...

...Vorremmo risolvere le infinite istanze con un metodo solo

[2] In inglese 'instance'

L'algebra viene in aiuto

Identità dei quattro quadrati ($\pm$ Eulero, 1749):

$$\begin{aligned}(g^2 + h^2 + i^2 + j^2)\,(k^2 + \ell^2 + m^2 + n^2) \;=\; & \\ (gk - h\ell - im - jn)^2 \;+\; & \\ (g\ell + hk + in - jm)^2 \;+\; & \\ (gm - hn + ik + j\ell)^2 \;+\; & \\ (gn + hm - i\ell + jk)^2 & \end{aligned}$$

[3] Trovarne altri è un esercizio per voi...

L'algebra viene in aiuto

Identità dei quattro quadrati ($\pm$ Eulero, 1749):

$$\begin{aligned}(g^2 + h^2 + i^2 + j^2)\,(k^2 + \ell^2 + m^2 + n^2) \;=\; & \\ (gk - h\ell - im - jn)^2 \;+\; & \\ (g\ell + hk + in - jm)^2 \;+\; & \\ (gm - hn + ik + j\ell)^2 \;+\; & \\ (gn + hm - i\ell + jk)^2 & \end{aligned}$$

Disponiamo dunque di almeno un *modo* (di quanti?)[3] per calcolare i numeri richiesti:

$$\begin{array}{lcl|lcl} a & = & gk - h\ell - im - jn & b & = & g\ell + hk + in - jm \\ c & = & gm - hn + ik + j\ell & d & = & gn + hm - i\ell + jk \end{array}$$

[3] Trovarne altri è un esercizio per voi...

La programmazione è piú impacciata dell'algebra

Un frammento di Java che implementa l'identità di Eulero:

$$\textbf{int}\quad a, b, c, d;$$

$$a = g*k - h*\ell - i*m - j*n; \qquad b = g*\ell + h*k + i*n - j*m;$$

$$c = g*m - h*n + i*k + j*\ell; \qquad d = g*n + h*m - i*\ell + j*k;$$

La programmazione è piú impacciata dell'algebra

Un frammento di Java che implementa l'identità di Eulero:

```
int a, b, c, d;
a = g*k - h*l - i*m - j*n;   b = g*l + h*k + i*n - j*m;
c = g*m - h*n + i*k + j*l;   d = g*n + h*m - i*l + j*k;
```

Qui Java *ci ha obbligato* a introdurre informazione extra:

lessicale: il **terminatore ";"** dopo ogni istruzione ;

La programmazione è piú impacciata dell'algebra

Un frammento di Java che implementa l'identità di Eulero:

$$\textbf{int}\quad a\,,b\,,c\,,d\,;$$

$$a = g*k-h*\ell-i*m-j*n\,; \qquad b = g*\ell+h*k+i*n-j*m\,;$$

$$c = g*m-h*n+i*k+j*\ell\,; \qquad d = g*n+h*m-i*\ell+j*k\,;$$

Qui Java *ci ha obbligato* a introdurre informazione extra:

- lessicale: il **terminatore ";"** dopo ogni istruzione ;
- sintattica: il segno **"*"** per la moltiplicazione

La programmazione è piú impacciata dell'algebra

Un frammento di Java che implementa l'identità di Eulero:

$$\textbf{int}\quad a\,,b\,,c\,,d\,;$$

$$a = g*k-h*\ell-i*m-j*n\,; \qquad b = g*\ell+h*k+i*n-j*m\,;$$

$$c = g*m-h*n+i*k+j*\ell\,; \qquad d = g*n+h*m-i*\ell+j*k\,;$$

Qui Java *ci ha obbligato* a introdurre informazione extra:

lessicale: il **terminatore ";"** dopo ogni istruzione ;

sintattica: il segno **"*"** per la moltiplicazione

semantica statica: la **dichiarazione di tipo** per le quattro **variabili** del risultato

La programmazione è piú impacciata dell'algebra

Un frammento di Java che implementa l'identità di Eulero:

```
int a, b, c, d;
a = g*k - h*ℓ - i*m - j*n;   b = g*ℓ + h*k + i*n - j*m;
c = g*m - h*n + i*k + j*ℓ;   d = g*n + h*m - i*ℓ + j*k;
```

Qui Java *ci ha obbligato* a introdurre informazione extra:

- lessicale: il **terminatore ";"** dopo ogni istruzione ;
- sintattica: il segno **"*"** per la moltiplicazione
- semantica statica: la **dichiarazione di tipo** per le quattro **variabili** del risultato (similmente, da qualche altra parte, andrà dichiarato il tipo degli 8 dati—anch'esso **int**)

L'istruzione di assegnamento

L'ASSEGNAZIONE ha la forma

$$\langle \text{variabile} \rangle \quad = \quad \langle \text{espressione} \rangle ;$$

L'istruzione di assegnamento

L'ASSEGNAZIONE ha la forma

$$\langle \text{variabile} \rangle \quad = \quad \langle \text{espressione} \rangle ;$$

ne è esempio la

$$a \quad = \quad g * k - h * \ell - i * m - j * n;$$

vista sopra.

Cosa intendiamo per variabile ? (metafora)

Cosa intendiamo per variabile ? (metafora)

Cosa intendiamo per variabile ? (metafora)

Morale:

1. Una variabile contiene un valore '*di un certo* **tipo**'...

Cosa intendiamo per variabile ? (metafora)

Morale:

1. Una variabile contiene un valore '*di un certo* **tipo**'...
2. ...al quale si può accedere.

Cosa intendiamo per variabile ? (metafora)

Morale:

1. Una variabile contiene un valore '*di un certo* **tipo**'...
2. ...al quale si[4] può accedere.

4

'Chi', con precisione?

Cosa intendiamo per variabile ? (metafora)

Cosa intendiamo per variabile ? (metafora)

Importante:

4. Il valore, per lo piú, è sostituibile . . .

Cosa intendiamo per variabile ? (metafora)

Importante:

4. Il valore, per lo piú, è sostituibile . . .
5. . . . anche molte volte . . .

Cosa intendiamo per variabile ? (metafora)

Importante:

4. Il valore, per lo piú, è sostituibile . . .
5. . . . anche molte volte . . .
6. . . . ma è ammesso un solo valore per volta.

Cosa intendiamo per variabile ? (metafora)

Importante:

4. Il valore, per lo piú, è sostituibile . . .
5. . . . anche molte volte . . .
6. . . . ma è ammesso un solo valore per volta.
7. *Battaglieremo per far stare in una variabile oggetti aggregati!*

= significa 'diventa'

Hanno senso sia il frammento di programma Java:

```
int j;
j = 0;
j = j+1;  // a dx ha il vecchio valore, a sn il nuovo
```

(che 'inizializza' j e subito dopo lo incrementa) che il frammento

= significa 'diventa'

Hanno senso sia il frammento di programma Java:

```
int j;
j = 0;
j = j+1; // a dx ha il vecchio valore, a sn il nuovo
```

(che 'inizializza' j e subito dopo lo incrementa) che il frammento

```
int i, j, k, ℓ;
i = 0;  j = 1;  k = 2;
ℓ = i;  i = j;  j = k;  k = ℓ;
```

(che assegna alle variabili i, j, k dei valori che poi fa circolare)

Esercizio: Scrivere assegnamenti che

°F: convertano gradi Celsius in gradi Fahrenheit

Esercizio: Scrivere assegnamenti che

°C: convertano gradi Fahrenheit in gradi Celsius

Allora cos'è una variabile?

Abbiamo notato un'usanza ben piú comune in programmazione che in matematica: la **variabile** è intesa come una 'scatola' contenente un valore del tipo appropriato, valore che può essere cambiato un numero indefinito di volte.

Allora cos'è una variabile?

Abbiamo notato un'usanza ben piú comune in programmazione che in matematica: la **variabile** è intesa come una 'scatola' contenente un valore del tipo appropriato, valore che può essere cambiato un numero indefinito di volte.

Ha un

nome che inizia con una lettera, o con il carattere "_";

Allora cos'è una variabile?

Abbiamo notato un'usanza ben piú comune in programmazione che in matematica: la **variabile** è intesa come una 'scatola' contenente un valore del tipo appropriato, valore che può essere cambiato un numero indefinito di volte.

Ha un

nome che inizia con una lettera, o con il carattere "_";

tipo che stabilisce quali valori le si potranno assegnare;

Allora cos'è una variabile?

Abbiamo notato un'usanza ben piú comune in programmazione che in matematica: la **variabile** è intesa come una 'scatola' contenente un valore del tipo appropriato, valore che può essere cambiato un numero indefinito di volte.

Ha

nome che inizia con una lettera, o con il carattere "_";

tipo che stabilisce quali valori le si potranno assegnare;

taglia che dipende dal suo tipo;

Allora cos'è una variabile?

Abbiamo notato un'usanza ben piú comune in programmazione che in matematica: la **variabile** è intesa come una 'scatola' contenente un valore del tipo appropriato, valore che può essere cambiato un numero indefinito di volte.

Ha

nome	che inizia con una lettera, o con il carattere "_";
tipo	che stabilisce quali valori le si potranno assegnare;
taglia	che dipende dal suo tipo;
valore	che le assegnazioni cambiano, sovrascrivendo il valore preesistente con il valore dell'espressione.

Allora cos'è una variabile?

Abbiamo notato un'usanza ben piú comune in programmazione che in matematica: la **variabile** è intesa come una 'scatola' contenente un valore del tipo appropriato, valore che può essere cambiato un numero indefinito di volte.

Ha

nome	che inizia con una lettera, o con il carattere "_";
tipo	che stabilisce quali valori le si potranno assegnare;
taglia	che dipende dal suo tipo;
valore	che le assegnazioni cambiano, sovrascrivendo il valore preesistente con il valore dell'espressione.

(Vedremo che ha anche un '*ciclo di vita*', legato al contesto dove la si è dichiarata)

Dichiarazione con assegnazione

Talvolta l'istruzione di assegnamento viene fusa con la dichiarazione di tipo come in questo esempio:

$$\textbf{int} \quad a \quad = \quad g * k - h * \ell - i * m - j * n \ ;$$

Dichiarazione con assegnazione

Talvolta l'istruzione di assegnamento viene fusa con la dichiarazione di tipo come in questo esempio:

$$\textbf{int} \quad a \quad = \quad g * k - h * \ell - i * m - j * n \ ;$$

ma *non* è permesso dichiarare *piú volte* il tipo di una stessa variabile: neanche se le dichiarazioni collimano.

Compatibilità fra assegnatario e assegnando

$$\langle \text{variabile} \rangle \quad = \quad \langle \text{espressione} \rangle ;$$

la variabile ha un tipo, cui deve **accordarsi** il valore dell'espressione.

Compatibilità fra assegnatario e assegnando

$$\langle \text{variabile} \rangle \quad = \quad \langle \text{espressione} \rangle ;$$

la variabile ha un tipo, cui deve **accordarsi** il valore dell'espressione.

Può darsi che l'espressione sia, essa pure, una variabile: in tal caso anche il suo tipo è stato dichiarato.

Compatibilità fra assegnatario e assegnando

$$\langle \text{variabile} \rangle \quad = \quad \langle \text{espressione} \rangle ;$$

la variabile ha un tipo, cui deve **accordarsi** il valore dell'espressione.

Può darsi che l'espressione sia, essa pure, una variabile: in tal caso anche il suo tipo è stato dichiarato.
Nel caso di un'espressione composita, il suo tipo è comunque determinabile (*ricorsivamente*), in base al tipo degli operandi.

Compatibilità fra assegnatario e assegnando

$$\langle \text{variabile} \rangle \quad = \quad \langle \text{espressione} \rangle ;$$

la variabile ha un tipo, cui deve **accordarsi** il valore dell'espressione.

Può darsi che l'espressione sia, essa pure, una variabile: in tal caso anche il suo tipo è stato dichiarato.
Nel caso di un'espressione composita, il suo tipo è comunque determinabile (*ricorsivamente*), in base al tipo degli operandi.

Eventuali **discordanze** di tipo si manifestano

prima che il programma vada in esecuzione .

Discordanza di tipo

Ecco un esempio di *in*eseguibilità è dovuta a una dichiarazione erronea:

```
int   piGreca;
piGreca        =   355.0 / 113; // Perché quel ".0" ?
```

Discordanza di tipo

Ecco un esempio di *in*eseguibilità è dovuta a una dichiarazione erronea:

```
int   piGreca;
piGreca        =   355.0 / 113; // Perché quel ".0" ?
```

Analoga incompatibilità fra valore e tipo in quest'esempio in due variabili:

```
int   pi;        double   piGreca;
piGreca    =     355.0 / 113;
pi         =     piGreca;
```

Ma piGreca non dovrebbe essere una costante?

Possiamo assegnare un valore *immodificabile* cosí:

Ma piGreca non dovrebbe essere una costante?

Possiamo assegnare un valore *immodificabile* cosí:

```
final double piGreca;
piGreca = 4.0 * Math.atan(1.0);
```

Alla luce di quanto precede. . .

Come scrivereste in Java le usuali formule di conversione

$$^{o}C = (5/9) \cdot (^{o}F - 32)$$

$$^{o}F = (9/5) \cdot {}^{o}C + 32$$

?

Si può indulgere al 'polimorfismo'? (Una scelta di campo)

Cenno: Sarebbe ammissibile cambiare il tipo di una variabile durante l'esecuzione di un programma ?

Si può indulgere al 'polimorfismo'? (Una scelta di campo)

Cenno: Sarebbe ammissibile cambiare il tipo di una variabile durante l'esecuzione di un programma ?

Qualche linguaggio di programmazione, specie fra quelli orientati al '*quick prototyping*', tratta i tipi con leggera disinvoltura. Però

Java rappresenta, al riguardo, una posizione piuttosto rigida

Si può indulgere al 'polimorfismo'? (Una scelta di campo)

Cenno: Sarebbe ammissibile cambiare il tipo di una variabile durante l'esecuzione di un programma ?

Qualche linguaggio di programmazione, specie fra quelli orientati al '*quick prototyping*', tratta i tipi con leggera disinvoltura. Però

Java rappresenta, al riguardo, una posizione piuttosto rigida

(che gode di maggior credito nell'industria).

Si può indulgere al 'polimorfismo'? (Una scelta di campo)

Cenno: Sarebbe ammissibile cambiare il tipo di una variabile durante l'esecuzione di un programma ?

Qualche linguaggio di programmazione, specie fra quelli orientati al '*quick prototyping*', tratta i tipi con leggera disinvoltura. Però

Java rappresenta, al riguardo, una posizione piuttosto rigida

(che gode di maggior credito nell'industria).

Nondimeno, incontreremo anche in Java qualche utile compromesso fra tipizzazione forte e debole.

Si può indulgere al ‘polimorfismo’?

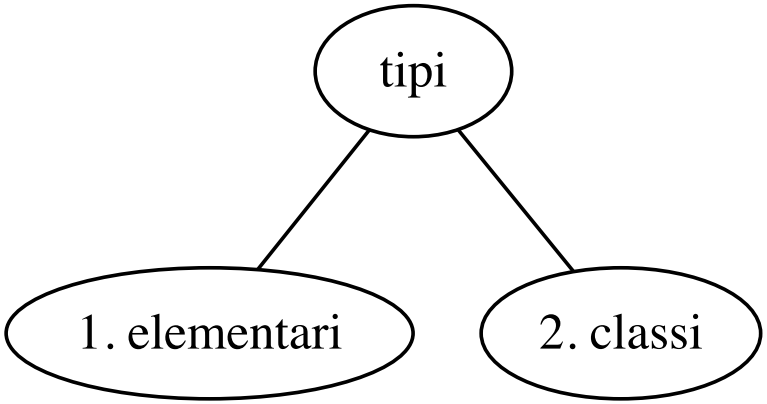
Sulle questioni relative ai tipi, l’atteggiamento matematico è in genere molto piú disinvolto e leggero. Per esempio, in merito all’identità di Eulero leggiamo:

Si può indulgere al 'polimorfismo'?

Sulle questioni relative ai tipi, l'atteggiamento matematico è in genere molto piú disinvolto e leggero. Per esempio, in merito all'identità di Eulero leggiamo:

> *Essa si può dimostrare con semplici passaggi di algebra elementare ed è valida in ogni anello commutativo.*
> *Se le a e le b sono numeri reali, esiste una dimostrazione piú elegante.*

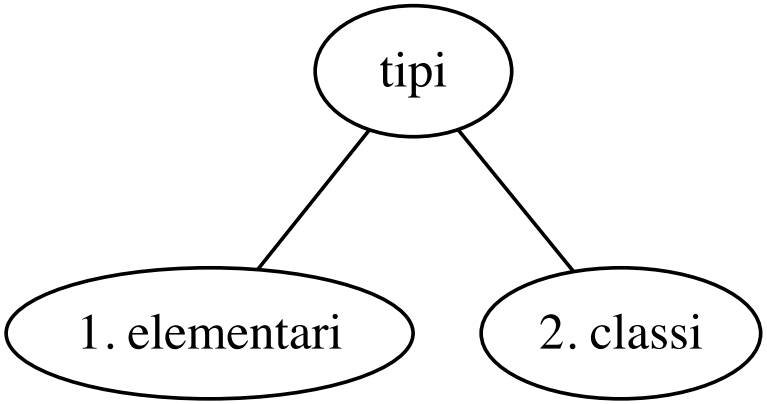
Due tipi di tipo



1. Gli uni sono in dotazione fissa del linguaggio
   — sono ‘retaggio del passato’

(Attenti al doppio senso di ‘classe’)

Due tipi di tipo



1. Gli uni sono in dotazione fissa del linguaggio
   — sono ‘retaggio del passato’
2. Gli altri si possono arricchire indefinitamente
   — serviranno a ‘istanziare oggetti’ sempre nuovi

(Attenti al doppio senso di ‘classe’)

Quali sono i tipi 'elementari'? I **tipi primitivi** di Java sono:

Nome	*Valore*	*Intervallo dei valori*
byte	intero	da -128 a 127
short	intero	da -32768 a 32767
int	intero	da -2147483648 a 2147483647
long	intero	da -9223372036854775808 a 9223372036854775807
float	numero in virgola mobile	...
double	numero in virgola mobile	...
char	carattere	tutti i val. Unicode da 0 a 65535
boolean	val. di verità	**true** / **false**

(Tipi ben piú elaborati verranno associati agli oggetti)

Attenti alle mistificazioni !

Non si possono assimilare (anche se spesso lo si fa) agli

interi i valori di tipo **int**, né ai

reali i valori di tipo **double**.

Attenti alle mistificazioni !

Non si possono assimilare (anche se spesso lo si fa) agli

interi i valori di tipo **int**, né ai

reali i valori di tipo **double**.

In effetti, quando un matematico considera gli insiemi

$$\begin{aligned} \mathbb{Z} &= \{0, \pm 1, \pm 2, \dots\} \\ \mathbb{N} &= \{0, 1, 2, \dots\} \end{aligned}$$

degli interi con e senza segno, assume che siano infiniti!

Attenti alle mistificazioni !

Non si possono assimilare (anche se spesso lo si fa) agli

interi i valori di tipo **int**, né ai

reali i valori di tipo **double**.

In effetti, quando un matematico considera gli insiemi

$$\begin{aligned} \mathbb{Z} &= \{0, \pm 1, \pm 2, \dots\} \\ \mathbb{N} &= \{0, 1, 2, \dots\} \end{aligned}$$

degli interi con e senza segno, assume che siano infiniti! Per di piú, gli insiemi $\mathbb{Q}$ ed $\mathbb{R}$ dei razionali e dei reali sono densi.

Attenti alle mistificazioni !

Non si possono assimilare (anche se spesso lo si fa) agli

interi i valori di tipo **int**, né ai

reali i valori di tipo **double**.

In effetti, quando un matematico considera gli insiemi

$$\begin{aligned} \mathbb{Z} &= \{0, \pm 1, \pm 2, \dots\} \\ \mathbb{N} &= \{0, 1, 2, \dots\} \end{aligned}$$

degli interi con e senza segno, assume che siano infiniti! Per di piú, gli insiemi $\mathbb{Q}$ ed $\mathbb{R}$ dei razionali e dei reali sono densi.

Nella programmazione le cose vanno ben diversamente. . .

Attenti alle mistificazioni !

Non si possono assimilare (anche se spesso lo si fa) agli

interi i valori di tipo **int**, né ai

reali i valori di tipo **double**.

In effetti, quando un matematico considera gli insiemi

$$\begin{aligned} \mathbb{Z} &= \{0, \pm 1, \pm 2, \dots\} \\ \mathbb{N} &= \{0, 1, 2, \dots\} \end{aligned}$$

degli interi con e senza segno, assume che siano infiniti! Per di piú, gli insiemi $\mathbb{Q}$ ed $\mathbb{R}$ dei razionali e dei reali sono densi.

Nella programmazione le cose vanno ben diversamente. . .

. . . Inoltre Java non ha un tipo primitivo per gli *interi senza segno*.

Volete almeno un esempio di classe ?

Con i caratteri (**char**) possiamo formare parole. . .

Volete almeno un esempio di classe ?

Con i caratteri (**char**) possiamo formare parole. . .
. . . anche parole lunghissime, come

Volete almeno un esempio di classe ?

Con i caratteri (**char**) possiamo formare parole. . .
. . . anche parole lunghissime, come
"`precipitevolissimevolmente`"

Volete almeno un esempio di classe ?

Con i caratteri (**char**) possiamo formare parole...
...anche parole lunghissime, come
"`precipitevolissimevolmente`"

Con gli stessi caratteri possiamo formare testi...

Volete almeno un esempio di classe ?

Con i caratteri (**char**) possiamo formare parole...
...anche parole lunghissime, come
"`precipitevolissimevolmente`"

Con gli stessi caratteri possiamo formare testi...
...anche testi lunghissimi, come il monologo di Molly Bloom

Volete almeno un esempio di classe ?

Con i caratteri (**char**) possiamo formare parole...
...anche parole lunghissime, come
"`precipitevolissimevolmente`"

Con gli stessi caratteri possiamo formare testi...
...anche testi lunghissimi, come il monologo di Molly Bloom

Nella dotazione iniziale di Java troveremo la classe `String`

Volete almeno un esempio di classe ?

Con i caratteri (**char**) possiamo formare parole...
...anche parole lunghissime, come
"`precipitevolissimevolmente`"

Con gli stessi caratteri possiamo formare testi...
...anche testi lunghissimi, come il monologo di Molly Bloom

Nella dotazione iniziale di Java troveremo la classe `String`

Come potremo far stare un'istanza di `String` dentro una variabile
(di taglia fissa)?

Realizzazione di programmi

Fasi della messa in opera (Situazione idealizzata):

Realizzazione di programmi

Fasi della messa in opera (Situazione idealizzata):

individuazione dei problemi

Realizzazione di programmi

Fasi della messa in opera (Situazione idealizzata):

individuazione dei problemi

ideazione degli algoritmi (procedimenti rigorosi) per risolverli

Realizzazione di programmi

Fasi della messa in opera (Situazione idealizzata):

individuazione dei problemi

ideazione degli algoritmi (procedimenti rigorosi) per risolverli

programmazione i.e. implementazione di tali algoritmi tramite classi Java (metodi formalizzati)

$\cdots \cdots$ `edit` $\longrightarrow$ `Progr.java`

Realizzazione di programmi

Fasi della messa in opera (Situazione idealizzata):

individuazione dei problemi

ideazione degli algoritmi (procedimenti rigorosi) per risolverli

programmazione i.e. implementazione di tali algoritmi tramite classi Java (metodi formalizzati)

$$\cdots \cdots \boxed{\texttt{edit}} \longrightarrow \texttt{Progr.java}$$

compilazione i.e. traduzione di tali programmi in istruzioni di una macchina 'virtuale'

$$\texttt{Progr.java} \longrightarrow \boxed{\texttt{javac}} \longrightarrow \texttt{Progr.class}$$

Realizzazione di programmi

Fasi della messa in opera (Situazione idealizzata):

individuazione dei problemi

ideazione degli algoritmi (procedimenti rigorosi) per risolverli

programmazione i.e. implementazione di tali algoritmi tramite classi Java (metodi formalizzati)

$$\cdots \cdots \boxed{\texttt{edit}} \longrightarrow \texttt{Progr.java}$$

compilazione i.e. traduzione di tali programmi in istruzioni di una macchina 'virtuale'

$$\texttt{Progr.java} \longrightarrow \boxed{\texttt{javac}} \longrightarrow \texttt{Progr.class}$$

esecuzione i.e. emulazione della macchina astratta sul proprio microprocessore

$$\texttt{Progr.class} \longrightarrow \boxed{\texttt{java}} \longrightarrow \cdots \cdots$$

Installazioni preliminari

Prima di poter utilizzare Java, occorre che abbiate sul PC:

- JDK (Java Development Kit):
  compilatore,
  emulatore della Java Virtual Machine,
  una **dotazione iniziale** di classi Java
- un **ambiente** di sviluppo

Installazioni preliminari

Prima di poter utilizzare Java, occorre che abbiate sul PC:

- JDK (Java Development Kit):
  compilatore,
  emulatore della Java Virtual Machine,
  una **dotazione iniziale** di classi Java
- un **ambiente** di sviluppo

L'ambiente di sviluppo è meno essenziale del JDK e per ora *non vi conviene* prenderne uno professionale; forse vi

- confonderebbe
- comporterebbe una spesa

Installazioni preliminari

Prima di poter utilizzare Java, occorre che abbiate sul PC:

- JDK (Java Development Kit):
  compilatore,
  emulatore della Java Virtual Machine,
  una **dotazione iniziale** di classi Java
- un **ambiente** di sviluppo

L'ambiente di sviluppo è meno essenziale del JDK e per ora *non vi conviene* prenderne uno professionale; forse vi

- confonderebbe
- comporterebbe una spesa

Ma non voglio sconfinare nella materia del 'laboratorio' di Programmazione. . .

Sitografia

it.wikipedia.org/wiki/Rompicapo_delle_otto_regine

Sitografia

it.wikipedia.org/wiki/Rompicapo_delle_otto_regine
it.wikipedia.org/wiki/Identità_dei_quattro_quadrati_di_Eulero