Laboratorio di Calcolo

Variabili e memoria

DIPARTIMENTO DI FISICA

SAPIENZA
UNIVERSITÀ DI ROMA

Variabili e costanti

```
a = 2;
b = 3.5;
c = 2.;
```

0xffffa000	
0xffffa001	
0xffffa002	00000010
0xffffa003	
0xffffa004	
0xffffa005	
0xffffa006	
0xffffa007	

Variabili e costanti

```
a = 2;
b = 3.5;
c = 2.;
```

Nota: per brevità le variabili intere sono rappresentate usando solo 8 bit

0xffffa000	
0xffffa001	
0xffffa002	00000010
0xffffa003	
0xffffa004	
0xffffa005	
0xffffa006	
0xffffa007	

Variabili e costanti

```
a = 2;
b = 3.5;
c = 2.;
```

$3.5 = (1 + 0.75)*2^1$

0xffffa000	
0xffffa001	
0xffffa002	00000010
0xffffa003	01000000
0xffffa004	01100000
0xffffa005	00000000
0xffffa006	00000000
0xffffa007	

Variabili e costanti

```
a = 2;
b = 3.5;
c = 2.;
```

$2. = (1 + 0)*2^1$

0xffffa002	00000010
0xffffa003	01000000
0xffffa004	01100000
0xffffa005	00000000
0xffffa006	00000000
0xffffa007	01000000
0xffffa008	00000000
0xffffa009	00000000
0xffffa00a	00000000

Variabili e costanti

```
a = 2;
b = 3.5;
c = 2.;
d = 'f';
```

f ↔ 102 (66_{16})

0xffffa003	01000000
0xffffa004	01100000
0xffffa005	00000000
0xffffa006	00000000
0xffffa007	01000000
0xffffa008	00000000
0xffffa009	00000000
0xffffa00a	00000000
0xffffa00b	01100110

Operatori matematici

```
c = a + b;
```

0xffffa002	00000010
0xffffa003	01000000
0xffffa004	01100000
0xffffa005	00000000
0xffffa006	00000000
0xffffa007	01000000
0xffffa008	00000000
0xffffa009	00000000
0xffffa00a	00000000

Operatori matematici

```
c = a + b;
```

Nota: nel registro della CPU il numero 2 viene "promosso" a `float`

0xffffa002	00000010
0xffffa003	01000000
0xffffa004	01100000
0xffffa005	00000000
0xffffa006	00000000
0xffffa007	01000000
0xffffa008	00000000
0xffffa009	00000000
0xffffa00a	00000000

Operatori matematici

```
c = a + b;
```

0xffffa002	00000010
0xffffa003	01000000
0xffffa004	01100000
0xffffa005	00000000
0xffffa006	00000000
0xffffa007	01000000
0xffffa008	00000000
0xffffa009	00000000
0xffffa00a	00000000

Operatori matematici

```
c = a + b;
```

$5.5 = (1 + 0.375)*2^2$

0xffffa002	00000010
0xffffa003	01000000
0xffffa004	01100000
0xffffa005	00000000
0xffffa006	00000000
0xffffa007	01000000
0xffffa008	10110000
0xffffa009	00000000
0xffffa00a	00000000

Operatori speciali

```
a = 2;
b = ++a;
```

0xffffa002	00000010
0xffffa003	

Operatori speciali

```
a = 2;
b = ++a;
```

0xffffa002	00000011
0xffffa003	00000011

Operatori speciali

```
a = 2;
b = ++a;
```

0xffffa002	00000010
0xffffa003	

```
a = 2;
b = a++;
```

Operatori speciali

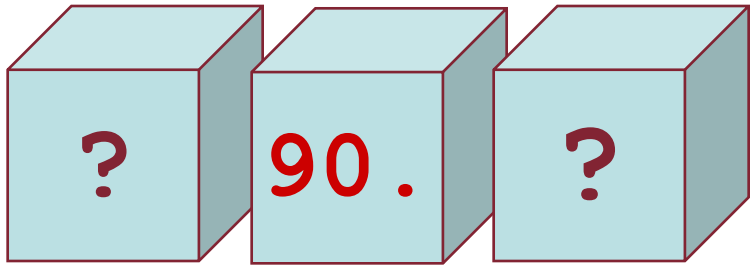
```
a = 2;
b = ++a;
```

0xffffa002	00000011
0xffffa003	00000010

```
a = 2;
b = a++;
```

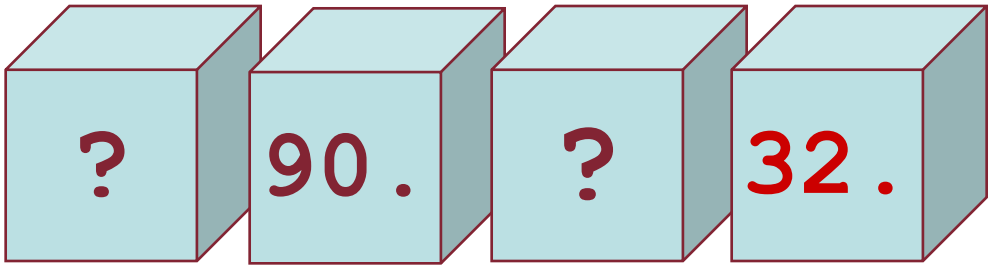
Programmi e variabili

```
main() {
    float tc, tf = 90., conv;
    float offset = 32.;
    conv = 5./9.;
    tc = (tf - offset) * conv;
}
```



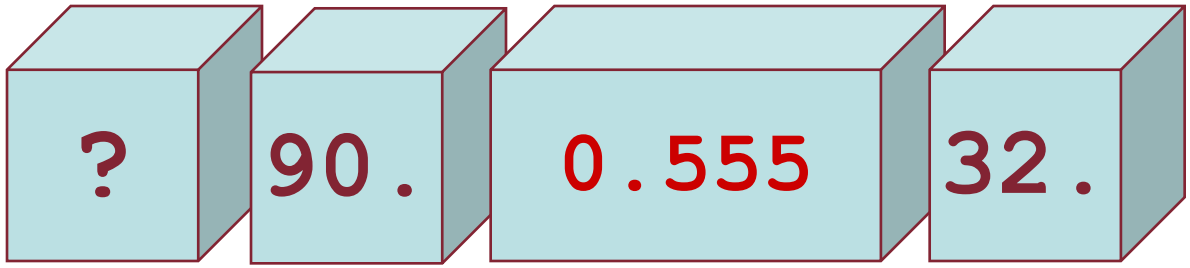
Programmi e variabili

```
main() {
    float tc, tf = 90., conv;
    float offset = 32.;
    conv = 5./9.;
    tc = (tf - offset) * conv;
}
```



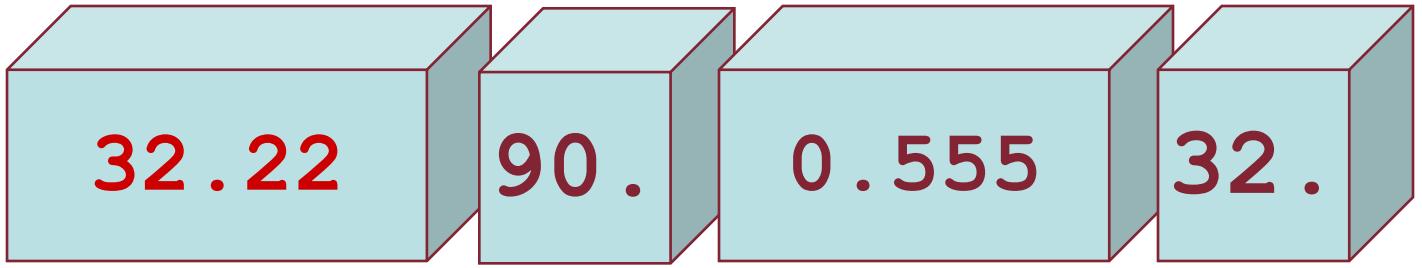
Programmi e variabili

```
main() {
    float tc, tf = 90., conv;
    float offset = 32.;
    conv = 5./9.;
    tc = (tf - offset) * conv;
}
```



Programmi e variabili

```
main() {
    float tc, tf = 90., conv;
    float offset = 32.;
    conv = 5./9.;
    tc = (tf - offset) * conv;
}
```



Test

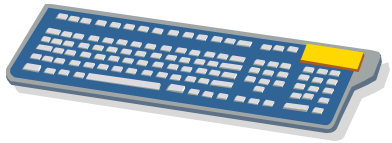
- Cosa accade se eseguo il programma?
- Nulla, perché non ci sono istruzioni di *output*!

Input/Output

```
scanf("%f", &tf);
```

Input/Output

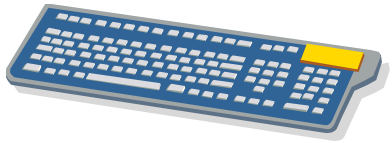
```
scanf("%f", &tf);
```



90

Input/Output

```
scanf("%lf", &tf);
```

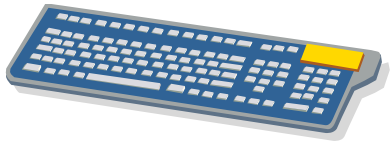


$90 = (1+0.40625)2^6$

01000010110100000000000...

Input/Output

```
scanf("%lf", &tf);
```



$90 = (1+0.40625)2^6$

01000010110100000000000...

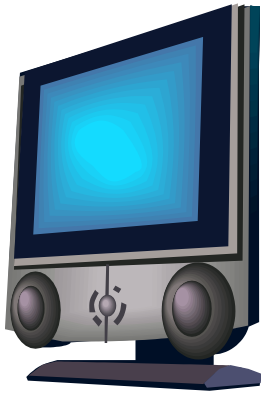
90.

Input/Output

```
printf("%f F = %f C\n", tf, tc);
```

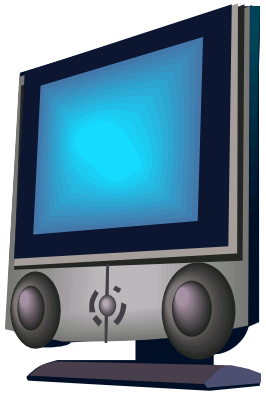
Input/Output

```
printf("%f F = %f C\n", tf, tc);
```



Input/Output

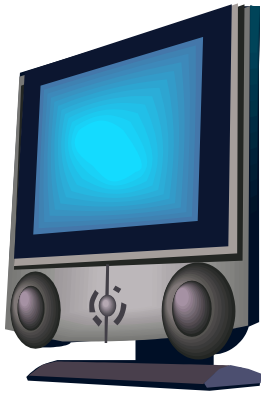
```
printf("%f F = %f C\n", tf, tc);
```



```
90.000000 F = ...
```

Input/Output

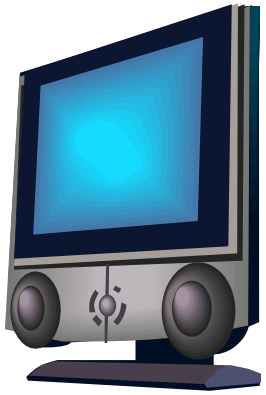
```
printf("%f F = %f C\n", tf, tc);
```



```
90.000000 F = 32.222222 C...
```

Input/Output

```
printf("%f F = %f C\n", tf, tc);
```



```
90.000000 F = 32.222222 C
>
```

Importanza della rappresentazione

```
(tc – offset) * 5./ 9.;
(tc – offset) * 5./ 9;
(tc – offset) * 5 / 9;

5 / 9 * (tf – offset);
```

Direttive

`#include`

Consente di includere il contenuto di un altro file

`#define`

Definisce un simbolo (Attenzione!!!)

`#ifdef, #ifndef, #else`

Compilazione condizionale