

Fondamenti di Informatica T-1

Modulo 2

Espressioni eterogenee

Cosa succede se in una espressione uso tipi diversi?

... laddove possibile, una espressione eterogenea viene risolta applicando la *promotion* dei tipi, e considerando *l'overloading* degli operatori...

Cosa succede se assegno ad un tipo *inferiore* un valore rappresentato tramite un tipo *superiore*?

Come si effettua la conversione esplicita da un tipo ad un altro?

Esercizio 1

(espressioni)

```
#include <stdio.h>
int main(void)
{
    int i, k;
    float j;

    i = 20;
    k = i % 3;
    i = i / 3;

    k = i / 4.0F;
    j = i / 4.0F;

    return (0);
}
```

1. Copiare, compilare ed eseguire il seguente programma...
2. Utilizzando il debug e le finestre di “watch”/“locals”, rispondere alle seguenti domande:
 - a) Quanto valgono *i* e *k* dopo il primo blocco di assegnamenti?
 - b) Quanto valgono *k* e *j* dopo il secondo blocco di assegnamenti?
 - c) Se *k* e *j* al termine del programma hanno valori diversi, perchè?
3. In fase di compilazione il programma potrebbe aver generato dei *warnings*...
 - a) Leggete i warning e spiegate il loro significato
 - b) Correggete il codice al fine di far scomparire i warning (compilate sempre con “rebuild all”)

Operatori: priorità ed associatività

| Priorità | Operatore | Simbolo | Associatività |
|----------|---|--|---------------|
| 1 (max) | chiamate a funzione selezioni | () [] -> . | a sinistra |
| 2 | operatori unari: op. negazione op. aritmetici unari op. incr. / decr. op. indir. e deref. op. sizeof | ! ~ + - ++ -- & * sizeof | a destra |
| 3 | op. moltiplicativi | * / % | a sinistra |
| 4 | op. additivi | + - | a sinistra |

Operatori: priorità ed associatività

| Priorità | Operatore | Simbolo | Associatività |
|----------|------------------------------------|---|---------------|
| 5 | op. di shift | >> << | a sinistra |
| 6 | op. relazionali | < <= > >= | a sinistra |
| 7 | op. uguaglianza | == != | a sinistra |
| 8 | op. di AND bit a bit | & | a sinistra |
| 9 | op. di XOR bit a bit | ^ | a sinistra |
| 10 | op. di OR bit a bit | | a sinistra |
| 11 | op. di AND logico | && | a sinistra |
| 12 | op. di OR logico | | a sinistra |
| 13 | op. condizionale | ? . . . : | a destra |
| 14 | op. assegnamento e sue varianti | = += -= *= /= %= &= ^= = <<= >>= | a destra |
| 15 (min) | op. concatenazione | , | a sinistra |

Esercizio 2

(espressioni e priorità degli operatori)

Scova l' errore...

Aldo, Giovanni e Giacomo hanno comprato in pasticceria una torta già tagliata in 12 fette, e poi si sono incontrati per mangiarla assieme.

E' andata più o meno così:

- Aldo ha mangiato 2 fette.
- Giovanni ne ha prese 5 nel piatto, ma poi si è ricordato di essere a dieta e quindi ne ha restituite 3.
- Giacomo ne ha prese 6, ma poi dopo averne mangiate 2 ha restituito le altre... salvo che di nascosto ne ha mangiato un' ulteriore fetta...


Hanno scritto quindi un programma che calcola il numero di fette rimaste nel piatto... al numero iniziale di fette hanno sottratto le fette mangiate da ogni membro della famiglia...

Esercizio 2

(espressioni e priorità degli operatori)

```
#include <stdio.h>
int main(void)
{
    int torta_i = 12;
    int torta_f;

    torta_f = torta_i - 2 - 5-3 - 6-4-1;
    return (0);
}
```



Il programma è **sbagliato**... perchè?

1. Correggete l'espressione inserendo le parentesi nei punti giusti, e senza sostituire all'operatore '-' l'operatore '+' ...
2. Verificate tramite il debugger che il numero di fette finale calcolato sia effettivamente quello giusto...

Esercizio 3

(espressioni, casting esplicito)

Federico, Carlo ed Eugenio sono andati a pranzo al ristorante, ed hanno ordinato:

- Federico: tortellini in panna, €9.00, contorno di verdure, €5.00, caffè, €1.20
- Carlo: gramigna con salsiccia, €8.00, contorno di verdure, € 5.00, caffè, €1.20
- Eugenio: Insalata Caesar, €13.00, dolce, €6.00, caffè, €1.20

I tre amici decidono di pagare dividendo in parti uguali.

Si implementi un programma che, a partire dalle cifre sopra esposte:

- a) Calcoli il totale dovuto da ognuno degli amici usando solo degli interi, che rappresenteranno quindi gli importi in centesimi di euro
- b) Stampi quanto dovuto da ogni amico, in termini di euro e centesimi di euro
- c) Calcoli il totale, divida per tre quote differenti, e stampi quanto dovuto esattamente da ogni amico. In particolare si stampi sia il valore esatto (con molte cifre decimali), che l'arrotondamento al centesimo di euro.

Qualora non sia possibile dividere in modo esatto, Federico dovrà pagare qualche centesimo più degli altri.

Esercizio 3 - Soluzione

(espressioni, casting esplicito)

```
#include <stdio.h>

int main() {
    int spesaF, spesaC, spesaE;
    int totale;
    int quota;
    spesaF = 900 + 500 + 120;
    spesaC = 800 + 500 + 120;
    spesaE = 1300 + 600 + 120;
    printf("Federico: %d.%d\n", spesaF/100, spesaF%100);
    printf("Carlo: %d.%d\n", spesaC/100, spesaC%100);
    printf("Eugenio: %d.%d\n", spesaE/100, spesaE%100);

    totale = spesaF + spesaC + spesaE;
    printf("Quota esatta: %f\n", totale / 3.0);

    quota = totale/3;
    printf("Quote arrotondate:\n");
    printf("Carlo: %d.%d\n", quota/100, quota%100);
    printf("Eugenio: %d.%d\n", quota/100, quota%100);
    printf("Federico: %d.%d\n", (totale-2*quota)/100, (totale-2*quota)%100);

    return 0;
}
```

Input e output in C

- Input con formato:

`scanf("stringa formato", lista variabili);`

- Output con formato:

`printf("stringa formato", lista variabili);`

- Tramite la stringa di formato, si specifica "come"
- Tramite i parametri successivi, si specifica "che cosa"

Esempio

(input output)

Esempio – Echo di un numero intero

- Realizzare (cioè scrivere e compilare) un programma che legga da tastiera un numero intero e ne stampi il valore a video (echo)
- pseudo-algoritmo:
 - Leggo da input un numero intero
 - Salvo il numero letto in una variabile apposita
 - Stampo a video il valore della variabile

Esempio

```
#include <stdio.h>
int main() {

    int value;

    scanf_s("%d", &value);
    printf("Valore letto:%d\n", value);
    return 0;
}
```

Esercizio 1

(input/output)

Esercizio 1 – Echo di caratteri

- Realizzare un programma che legga da tastiera tre caratteri e ne stampi il valore a video (echo)

Esercizio 1 – Soluzione

(input/output)

```
#include <stdio.h>
int main()  {
    char c1, c2, c3;

    scanf_s("%c%c%c", &c1, 1, &c2, 1, &c3, 1);
    printf("Caratteri: %c %c %c\n", c1, c2,
c3);
    return 0;
}
```

Esercizio 2

(IO, semplice programma)

- Realizzare un programma che legga da input tre numeri interi e stampi a video la loro somma e la media.

Esercizio 2 - Soluzione

(IO, semplice programma)

```
#include <stdio.h>

int main() {
    int num1, num2, num3, somma;
    float media;

    scanf_s("%d%d%d", &num1, &num2, &num3);

    somma = num1 + num2 + num3;
    media = somma / 3.0F;
    printf("Somma: %d\n", somma);
    printf("Media: %f\n", media);
    return 0;
}
```


Esempio

(espressioni condizionali)

Esempio – Stabilire il massimo tra due valori

- Realizzare un programma che legga da input due numeri reali, e ne stampi a video il valore massimo
- Al fine di determinare il massimo, si utilizzino le sole espressioni condizionali

Esempio

(espressioni condizionali)

```
#include <stdio.h>
int main() {
    float num1, num2, max;

    scanf_s("%f %f", &num1, &num2);

    max = (num1 > num2) ? num1 : num2;
    printf("Max: %f\n", max);
    return 0;
}
```

Esercizio 1

(espressioni condizionali)

Elaborazione di numeri reali

- Realizzare un programma che legga da input un numero reale, e stampi a video:
 1. il suo valore assoluto
 2. La parte intera del suo valore assoluto

Esercizio 1 – Soluzione

(espressioni condizionali)

```
#include <stdio.h>
int main() {
    float num1, abs_real;
    int abs_int;

    scanf_s("%f", &num1);

    abs_real = ((num1 > 0) ? num1 : - num1);
    abs_int = (int) abs_real;
    printf("Absolute value: %f\n", abs_real);
    printf("Absolute integer value: %d\n", abs_int);
    return 0;
}
```

Esercizio 2

(espressioni condizionali)

Stampa di caratteri in ordine alfabetico

- Realizzare un programma che legga da input tre caratteri e li stampi in ordine alfabetico.

A tal scopo, si rammenti la rappresentazione dei caratteri in linguaggio C...

- Si utilizzino solo le espressioni condizionali (e non l'istruzione if... ad esempio)

Esercizio 2 – Soluzione 1

(espressioni condizionali)

```
#include <stdio.h>
int main() {
    char c1, c2, c3;
    char first, second, third;
    int temp;

    scanf_s("%c%c%c", &c1, 1, &c2, 1, &c3, 1);
    // scanf("%c%c%c", &c1, &c2, &c3);

    first = ((c1 < c2)? c1 : c2);
    first = ((first < c3)? first : c3);

    third = ((c1 > c2)? c1 : c2);
    third = ((third > c3)? third : c3);

    temp = c1 + c2 + c3 - first - third;
    second = (char) temp;

    printf("Characters: %c %c %c\n", first, second, third);
    return 0;
}
```

Esercizio 2 – Soluzione 2

(espressioni condizionali)

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

```
int main() {
    char c1, c2, c3;
    char first, second, third;
    char temp1, temp2;

    scanf_s("%c%c%c", &c1, 1, &c2, 1, &c3, 1);
    // scanf("%c%c%c", &c1, &c2, &c3);

    first = ((c1<c2)? (temp1=c2, c1) : (temp1=c1, c2));
    first = ((first<c3)? (temp2=c3, first) : (temp2=first, c3));
    second = ((temp1<temp2)? (third=temp2, temp1) : (third=temp1, temp2));

    printf("Characters: %c %c %c\n", first, second, third);
    return 0;
}
```

Esercizio 2 – Soluzione 3

(espressioni condizionali)

```
#include <stdio.h>

int main () {
    char c1, c2, c3, tmp;

    printf("Enter 3 chars: ");
    scanf_s("%c%c%c", &c1, 1, &c2, 1, &c3, 1);
    // scanf("%c%c%c", &c1, &c2, &c3);

    (c1 < c2)? (1) : (tmp = c1, c1 = c2, c2 = tmp);
    (c2 < c3)? (1) : (tmp = c2, c2 = c3, c3 = tmp);
    (c1 < c2)? (1) : (tmp = c1, c1 = c2, c2 = tmp);

    printf("Sorted chars: %c %c %c\n", c1, c2, c3);
    return 0;
}
```