Scuola di Ingegneria Industriale

Laurea in Ingegneria Energetica

Laurea in Ingegneria Meccanica

POLITECNICO DI MILANO

Dipartimento di Elettronica, Informazione e Bioingegneria





Informatica B

Sezione D

Franchi Alessio Mauro, PhD *alessiomauro.franchi@polimi.it*

Esercizio 15

Scrivere un programma che, dato un numero intero inserito da tastiera, stampi a video tutti i suoi divisori;

Esercizio 15 - Soluzione

Il programma deve ispezionare tutti (o quasi) i numeri minori del numero N inserito dall'utente da tastiera;

Iterazione, il ciclo for

Chiedo all'utente un numero intero positivo; ricordatevi di controllare gli input da tastiera!

```
int num,i;
printf("Inserisci un numero intero positivo: ");
scanf("%d",&num);
if(num>0)
{
```

Esercizio 15 - Soluzione

Adesso devo verificare, per ogni intero positivo minore del numero appena letto, se sia un divisore

```
printf("I divisori di %d sono: ",num);
for(i=1;i<=num/2;i++){
        if(num%i==0)
                printf("%d - ",i);
}
```

Condizioni di entrata, di uscita e di incremento del ciclo for

Corpo del ciclo for

for (exp.Iniz; cond; exp.Incr)

Valore iniziale della variabile contatore
i = 0;

Condizione di uscita
i < num/2

Espressione di incremento
i++

Esercizio 15 - Soluzione

Qualche indicazione sul ciclo for:

- Il ciclo ha una **variabile contatore** (in genere numerica che parte da zero e viene incrementata): conta il numero di volte che si è eseguito il ciclo;
- Devo sapere a priori il **numero di iterazioni**: si deve specificare il **valore iniziale** della variabile contatore e il **valore finale**, espresso come **condizione**;
- **L'incremento** della variabile contatore può essere di qualsiasi entità (step), anche negativo (il contatore viene decrementato)
- In linea di principio qualunque algoritmo che impieghi un ciclo for può essere trascritto in una forma che usa solo il ciclo while, e viceversa.

Esercizio 16

Scrivere un programma che stampi a video, dato un numero intero x in ingresso, tutti i numeri interi da x a 0;

Esercizio 16 - Soluzione

Anche in questo caso sappiamo a priori quanti cicli il mio programma deve effettuare; usiamo quindi un ciclo for!

Attenzione: dobbiamo stampare i numeri da x a 0, in ordine inverso! Quindi nel ciclo for dobbiamo **decrementare** la variabile contatore!

```
int max,i;
printf("Inserisci un numero intero positivo: ");
scanf("%d",&max);
if(max>0)
for(i=max;i>=0;i--)
        printf("%d - ",i);
else
        printf("Numero inserito non valido\n");
return 0;
```

Inizializzo la variabile contatore **i** pari al numero inserito e ad ogni ciclo la decremento di 1 (**i--**); devo ciclare fino a quanto **i è positiva**.

Esercizio 17

Scrivere un programma che stampi a video la somma dei primi N numeri naturali, dove N è un intero positivo inserito da tastiera;

Esercizio 18

Scrivere un programma che chieda all'utente di inserire numeri interi da tastiera e li ristampi immediatamente a video; il programma termina quando l'utente inserisce uno zero (lo zero non deve essere stampato a video). (ATTENZIONE: lo abbiamo appena visto con il while, qui lo facciamo con il DO-WHILE!)

Esercizio 18 - Soluzione

Il programma deve continuamente chiedere all'utente di inserire un numero e stamparlo a video. Termina quando l'utente inserisce 0.

Il ciclo do-while

Nessuna condizione all'ingresso del ciclo

```
int num;
do{
        printf("Inserisci il prossimo numero: ");
        scanf("%d",&num);
        if(num!=0)
                printf("Hai inserito %d.\n",num);
}while(num!=0);

return 0;
```

Il corpo del ciclo

La condizione di permamenza è verificata alla fine del ciclo!

Esercizio 18 - Soluzione

Ciclo while e do-while a confronto

```
int num;
do{
    printf("Inserisci il prossimo numero: ");
    scanf("%d",&num);
    if(num!=0)
        printf("Hai inserito %d.\n",num);
}while(num!=0);
return 0;
```

Non serve!

La condizione si sposta

```
int num;
printf("Inserisci un numero: ");
scanf("%d",&num);
while(num!=0){
        printf("Hai inserito %d.\n",num);
        printf("Inserisci il prossimo numero: ");
        scanf("%d",&num);
}
return 0;
```

Il ciclo while può non essere mai eseguito, il do-while viene eseguito una o più volte!

Esercizio 19

Scrivere un programma che sommi tutti i numeri positivi inseriti dall'utente da tastiera; l'esecuzione termina quando l'utente inserisce 0; I numeri negativi inseriti non devono essere sommati.

Esercizio 19 - Soluzione

Anche in questo caso il programma deve continuamente chiedere all'utente di inserire un numero.

Serve un ciclo. Quale?

- Sappiamo a priori quanti cicli devo eseguire? NO, quindi non useremo il ciclo for
- Il ciclo deve essere eseguito almeno una volta? SI, quindi useremo un ciclo do-while!

```
int main()
{
    int num;
    int sum=0;
        do{
            …
            ...
        }while(num!=0);
        printf("La somma è %d",sum);
        return 0;
}
```

Condizione di permamenza nel ciclo: il numero inserito diverso da 0!

Esercizio 19 - Soluzione

Nel corpo del ciclo devo chiedere all'utente il numero e farne la somma con i precedenti!

Attenzione! I numeri negativi non devo considerarli! Come si fa?

```
{
printf("Inserisci un numero: ");
scanf("%d",&num);
if(num<0)
        continue;
sum+=num;
}
```

Se il numero è negativo, “continue”

Continue e break

L'istruzione **continue** interrompe l'iterazione corrente del ciclo e da inizio a quella successiva;

L'istruzione **break** fa uscire del tutto dal ciclo.

Si possono usare in tutti i tipi di ciclo (for, while, do-while)

Esercizio 21

Scrivere un programma che, a seconda del voto inserito (A,B,C,D,E) da tastiera, stampi a video un commento

(ad esempio, inserendo “A” da tastiera il programma potrebbe scrivere in output “Eccellente!”, e cosi via!).

Esercizio 21 - Soluzione

Si può fare in due modi:

Con una serie (a volte luuuunga) di if-else if-else, uno per ogni possibile input da tastiera;

Con il costrutto **switch.**

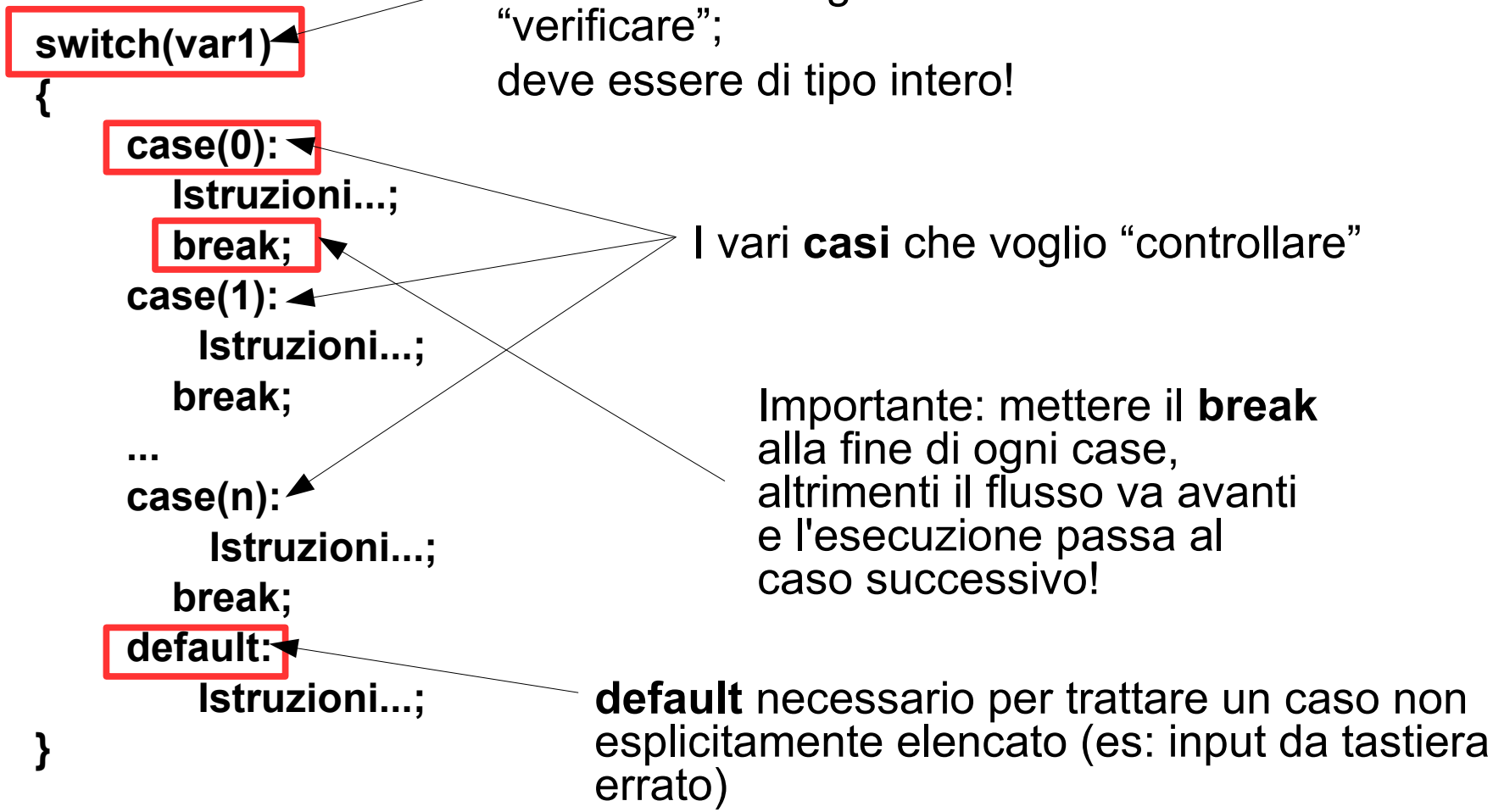
```
scanf("%c",&grade);

if (grade == 'a')
        printf("Eccellente!\n");
else if (grade == 'b');
        printf("Ottimo!\n");
else if (grade == 'c')
        printf("Insoma...!\n");
else if (grade == 'd')
        printf("Lasciamo perdere!\n");
else
        printf("Voto non corretto!\n");
```

```
scanf("%c",&grade);
switch(grade)
{
    case('a'):
        printf("Eccellente!\n");
        break;
    case('b'):
        printf("Ottimo!\n");
        break;
    case('c'):
        printf("Insoma...!\n");
        break;
    case('d'):
        printf("Lasciamo perdere!\n");
        break;
    default:
        printf("Voto non corretto!\n");
}
```

Esercizio 21 - Soluzione

Il costrutto switch in generale:



Esercizio 21 - Soluzione

Quali sono i vantaggi dello switch rispetto ad un if - else if - else?

1) **Leggibilità del codice!**

Immaginate di avere 100 casi, quale dei due codici sarà più chiaro?

2) **Performance!**

Con gli if – else if – else il computer verifica ogni condizione, con lo switch il computer sa già a priori cosa deve fare, senza confronti!

Esercizio 22

Scrivere un programma che acquisisca due numeri decimali da tastiera e svolga quindi le seguenti operazioni a seconda dell'input da tastiera:

-Se l'utente inserisco 0 termina;

-Se l'utente inserisce 1 ne stampa a video la somma;

-Se l'utente inserisce 2 ne stampa a video la differenza;

-Se l'utente inserisce 3 stampa a video il maggiore.

-In tutti gli altri casi stampi a video “Comando non riconosciuto” ed esca.

I tipi di dato in C

Il c è un linguaggio (debolmente) tipizzato: ogni variabile deve avere un tipo!

char %c	1 byte	-128 to 127 o 0 to 255 (a,b,c,d,...)
int %d	4 byte	-2147483648 to 2147483647
float %f	4 byte	$\pm 1.18 \times 10^{-38}$ to $\pm 3.4 \times 10^{38}$
double %f	8 byte	$\pm 2.23 \times 10^{-308}$ to $\pm 1.80 \times 10^{308}$
string %s	-	Sequenza di caratteri terminata da '\0'

Modificatori:

- short : meno bit
- long : più bit
- signed : con segno
- unsigned : senza segno

Provate ad eseguire questo codice:

```
int k = 2147483630;
printf("%d\n", k);
k = 2147483670;
printf("%d\n", k);
```

I tipi di dato in C

Esistono le **variabili enumerative**:

enum { gennaio, febbraio, marzo, aprile, maggio, giugno, luglio, agosto, settembre, ottobre, novembre, dicembre } mese;

mese = aprile; ⟶ mese = 3;

printf("%d",mese); ⟶ output: 5

I valori nelle enum sono **interi**; le stringhe non sono accessibili!
(Non potete scrivere questo: printf("%s",mese);

Esercizio 13

Scrivere un programma che stampi a video l'alfabeto “a-z”

(AIUTO: ricordatevi che in c il tipo char è equivalente al tipo intero, con la conversione data dalla tabella ASCII)

Esercizio 13 - soluzione

In c il tipo char è in realtà per il programma un numero intero!

ASCII: American Standard Code for Information Interchange

0 – carattere NULL ('\0')
32 – spazio (' ')
Da 48 a 57 – numeri da 0 a 9
Da 65 a 90 – lettere maiuscole 'A'-'Z'
Da 97 a 122 – lettere maiuscole 'a'-'z'
(97 – 65 = 32, ricordatevi la differenza tra minuscole e maiuscole!)

```
char c ='A';

printf ("%c", c); // output A
printf ("%d", c); // output 65
printf ("%c", c + 32); // output a
printf ("%d", 'B' – 'A')); // output 1
```

Esercizio 13 - soluzione

Guardando la tabella ASCII caratteri minuscoli dell'alfabeto equivalgono agli interi compresi tra 97 e 122. Quindi ci basta stampare come caratteri gli interi compresi tra 97 e 122.

```
int main()
{
        int i;
        for (i=97;i<123;i++)
                printf("%c\n",i);

        return 0;
}
```

Esercizio 14

Scrivere un programma che legge da tastiera un carattere minuscolo e ne stampa il codice ASCII; il programma deve continuare sino a quanto l'utente inserisce un carattere non minuscolo.

Esercizio 14 - Soluzione

Per prima cosa chiediamo all'utente il carattere e lo salviamo in una variabile di tipo char

```
char c;
int i;
printf("Inserisci un carattere minuscolo: ");
scanf("%c", &c);
```

Cicliamo fino a quanto il carattere inserito è “maggiore” di 'a' e “minore” di 'z'.
Notate l'uso di operatori matematici con I caratteri

```
while( c >= 'a'&& c <= 'z') {
        printf("valore ASCII per %c risulta %d\n", c, c);
        printf("scrivi un car. minuscolo (ogni altro per finire)\n");
        scanf("%c", &c);
        scanf("%c", &c);
}
```

Esercizio 14 - Soluzione

Per prima cosa chiediamo all'utente il carattere e lo salviamo in una variabile di tipo char

```
char c;
int i;
printf("Inserisci un carattere minuscolo: ");
scanf("%c", &c);
```

Cicliamo fino a quanto il carattere inserito è “maggiore” di 'a' e “minore” di 'z'.
Notate l'uso di operatori matematici con I caratteri

```
while( c >= 'a' && c <= 'z') {
        printf("valore ASCII per %c risulta %d\n", c, c);
        printf("Inserisci un carattere minuscolo: ");
        scanf("%c", &c);
        scanf("%c", &c);
}
```

Perchè abbimo messo una doppia scanf(“%c”,&c)?

Esercizio 23

Chiedere all'utente di inserire un numero intero positivo da tastiera; chiedere poi all'utente in quale base (binario, ottale, esadecimale) vuole convertirlo e stampare a video il numero convertito di conseguenza.

Aggiunta: continuare l'esecuzione del programma sino a quando l'utente inserisce un numero negativo da tastiera

Esercizio 14 - Soluzione

Per prima cosa chiediamo all'utente il carattere e lo salviamo in una variabile di tipo char

```
char c;
int i;
printf("Inserisci un carattere minuscolo: ");
scanf("%c", &c);
```

Cicliamo fino a quanto il carattere inserito è “maggiore” di 'a' e “minore” di 'z'.
Notate l'uso di operatori matematici con I caratteri

```
while( c >= 'a' && c <= 'z') {
        printf("valore ASCII per %c risulta %d\n", c, c);
        printf("Inserisci un carattere minuscolo: ");
        scanf("%c", &c);
        scanf("%c", &c);
}
```

Perchè abbimo messo una doppia scanf(“%c”,&c)?

Esercizio 14 - Soluzione

Per prima cosa chiediamo all'utente il carattere e lo salviamo in una variabile di tipo char

```
char c;
int i;
printf("Inserisci un carattere minuscolo: ");
scanf("%c", &c);
```

Cicliamo fino a quanto il carattere inserito è “maggiore” di 'a' e “minore” di 'z'.
Notate l'uso di operatori matematici con I caratteri

```
while( c >= 'a' && c <= 'z') {
        printf("valore ASCII per %c risulta %d\n", c, c);
        printf("Inserisci un carattere minuscolo: ");
        scanf("%c", &c);
        scanf("%c", &c);
}
```

Perchè abbimo messo una doppia scanf(“%c”,&c)?