UNIVERSITÀ
DI TORINO

Laboratorio di Programmazione I

Lezione n. 3: Condizioni booleane

Alessandro Mazzei

Slides: prof. Elvio Amparore

Espressioni booleane

Quali valori assumono queste espressioni logiche:

- `3>5 || 10 == 7+3` — **true**
- `3 != 5 && (6 < 2 || 5+2 == 10 - 3)` — **true**
- `3 < 5 < 7` — **NO!** (3<5)<7 ⇒ 1<7 vero ma per il motivo sbagliato!
- `4 > 3 > 2` — **NO!** (4>3)>2 ⇒ 1>2 ⇒ FALSO!
- `3 < 5 && 5 < 7` — **true**
- `3 < 5 && 7 < 5` — **false**
- `true && false` — **false**
- `true && 5 < 7` — **true**
- `false || 5 < 10` — **true**

Booleani

Leggere il codice del file **booleani.c** per vedere vari esempi di operazioni booleane in C.

Procedere con gli esercizi successivi solo dopo aver capito tutte le operazioni di questo codice.

NOTA: le leggi di De Morgan stabiliscono relazioni di equivalenza tra gli operatori di congiunzione e disgiunzione logica. Si possono sintetizzare come

$$\neg(a \wedge b) = \neg a \vee \neg b$$

$$\neg(a \vee b) = \neg a \wedge \neg b$$

Identificare problemi con gcc

Gli avvertimenti del compilatore (**warnings**) sono messaggi diagnostici visualizzati per certi costrutti che non sono inerentemente errati, ma che sono potenzialmente sbagliati o indici di errori nel codice.

Per abilitare i warnings, abbiamo due opzioni:

- **-Wall** riporta gli avvertimenti
- **-Werror** se ci sono avvertimenti il programma non viene compilato.

NOTA: per compilare abilitando i warnings usiamo:

- Unix: **gcc *-Wall*** *aritmetica.c* **-o** *aritmetica*
- Windows: **gcc *-Wall*** *aritmetica.c* **-o** *aritmetica.exe*

NOTA: in sede di esame useremo: **-Wall -Werror**

Esercizi booleani da completare

Aprire il codice di **esercizi_booleani.c**, e implementare i 6 punti indicati:

1. dati i due interi **a** e **b**, stampa **1** se il primo intero è multiplo del secondo, **0** altrimenti;
2. dato l'intero **a** (un voto), stampa "**true**" se **a** è compreso fra 1 e 30 (inclusi), "**false**" altrimenti;
3. dati i due booleani **b1** e **b2**, stampa a video il booleano "**false**" se **b1** e **b2** sono entrambi **veri**, "**true**" altrimenti;
4. dati i due interi **a** e **b**, stampa a video il massimo; se sono uguali stampa a video "**I due valori sono uguali**".
5. dati tre interi **a**, **b** e **c**, stampa a video il massimo (si usi una variabile di supporto max);
6. dati tre interi **a**, **b** e **c**, stampa a video "**ordinati**" se questi sono ordinati in modo crescente, altrimenti non stampare nulla.

Saluti!

Scrivere un programma **saluta.c** che chieda l'ora all'utente (come numero intero 0-23) e stampi a terminale il saluto più appropriato tra:

- “Buongiorno”
- “Buon pomeriggio”
- “Buonasera”
- “Buona notte”

Definite voi quali intervalli usare per decidere il saluto più appropriato.

Scelte su interi

Sulla pagina Moodle trovate un esercizio con nome

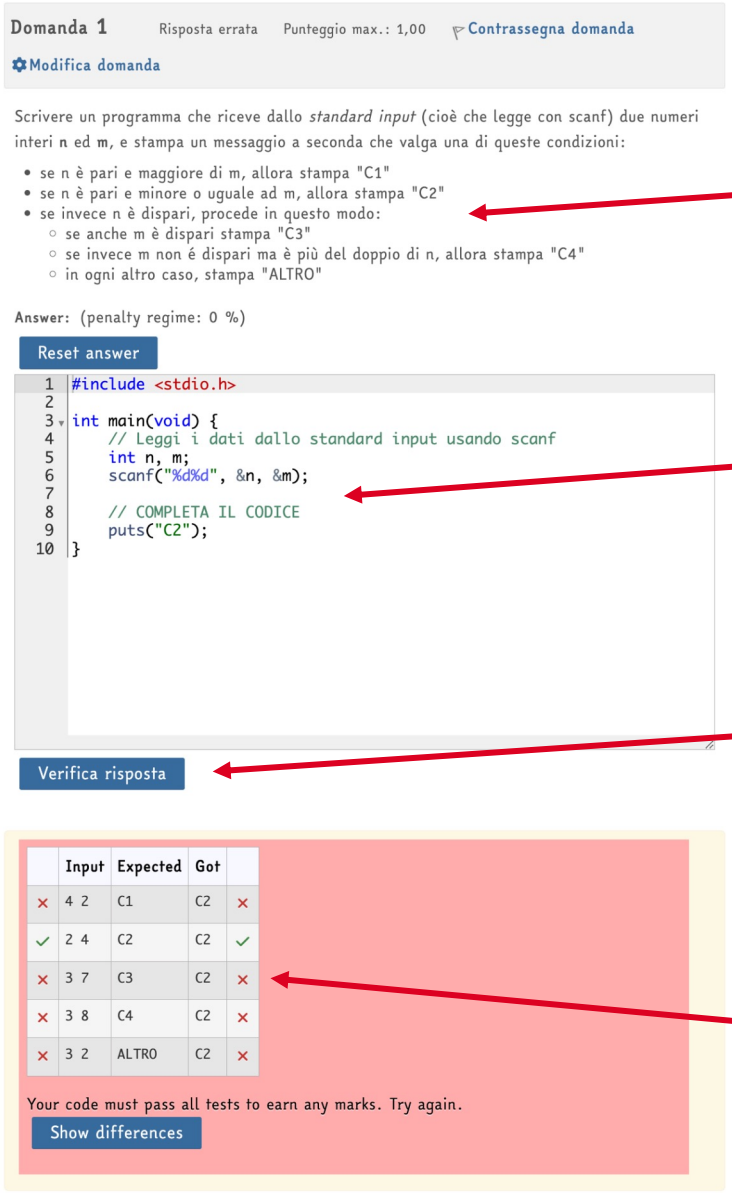
Lab03-Es1 Scelte su Interi

Aprite il link e premete il tasto

Tenta il quiz adesso

per avviare l'esercizio.

Come funziona Coderunner



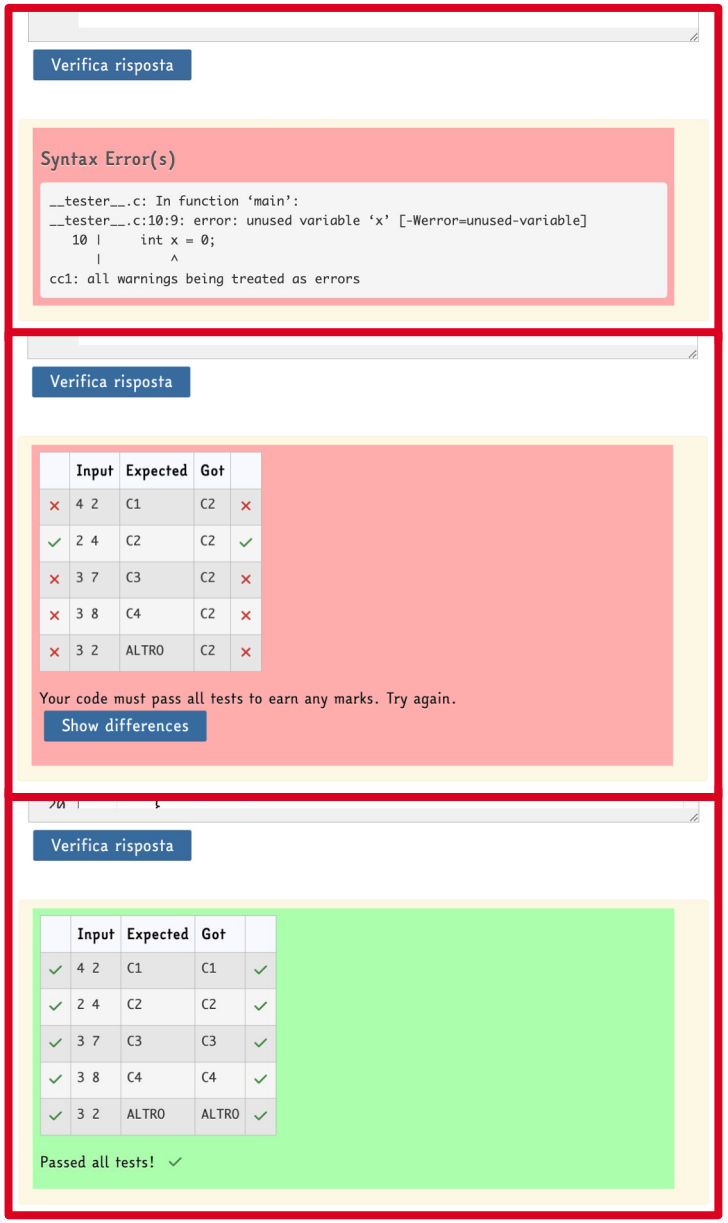
Testo dell'esercizio
⚠Leggere con cura prima di iniziare a scrivere il codice!

Area di lavoro in cui scrivere il codice C del programma.

Compila ed esegui.
Potete verificare il codice quante volte volete.

Il programma viene eseguito per diversi input, e viene controllato che le stampe generate con printf corrispondano con quelle attese

Esiti della verifica della risposta



Errore di compilazione!

NOTA: viene compilato con: **-Wall -Werror**

Il programma compila ed esegue, ma **non passa tutti i test** definiti, cioè non produce in output il testo atteso.

Il programma compila, esegue, e **passa tutti i test**, cioè produce tutti gli output attesi.

Niente panico!

- La compilazione e l'esecuzione con degli input predefiniti (**test**) è automatica: l'obiettivo è scrivere un codice robusto, che risponda correttamente a input diversi.
- Gli input proposti sono formulati per testare varie casistiche.

Non è l'esame, l'esercizio non viene valutato!

NOTA: però l'interfaccia web riporta in alcuni punti dei messaggi su punteggi, che potete ignorare.

Errori 1/2

Quali errori vedete in questo codice?

```c
#include <stdio.h>

int main() {
    int voto;
    scanf("%d", &voto);
    if (voto >= 18);
        printf("Esame passato.\n");
}
```

Errori 1/2

Quali errori vedete in questo codice?

```c
#include <stdio.h>

int main() {
    int voto;
    scanf("%d", &voto);
    if (voto >= 18);
        printf("Esame passato.\n");
}
```

🟥 È un errore particolarmente insidioso perché il codice compila senza segnalare problemi. Per il linguaggio C è come se avessimo scritto:

```c
    if (voto >= 18)
        ;       // istruzione vuota
    printf("Esame passato.\n");
```

Errori 2/2

Quali errori vedete in questo codice?

```c
#include <stdio.h>
int main() {
    int voto;
    scanf("%d", &voto);
    if (voto >= 18)
        printf("Esame passato.\n");
    else
        printf("Esame non passato.\n");
        printf("Ripresentarsi al prossimo appello.\n");
}
```

Errori 2/2

Quali errori vedete in questo codice?

```
#include <stdio.h>
int main() {
    int voto;
    scanf("%d", &voto);
    if (voto >= 18)
        printf("Esame passato.\n");
    else {
        printf("Esame non passato.\n");
        printf("Ripresentarsi al prossimo appello.\n");
    }
}
```

Mancano le graffe per un blocco **else** con più di un'istruzione! Anche questo è un errore particolarmente insidioso perché il codice compila senza segnalare problemi. Per il linguaggio C solo la prima istruzione dopo la parola chiave **else** è eseguita nel blocco condizionale. L'ultima printf è quindi eseguita sempre!

Errori 2/2

È sempre meglio mettere i blocchi condizionali tra parentesi graffe, a meno che non siano molto brevi ed evidenti.

```c
#include <stdio.h>

int main() {
    int voto;
    scanf("%d", &voto);
    if (voto >= 18) {
        printf("Esame passato.\n");
    }
    else {
        printf("Esame non passato.\n");
        printf("Ripresentarsi al prossimo appello.\n");
    }
}
```

Emersione del massimo con 4 variabili

Sulla pagina Moodle trovate un esercizio con nome

Lab03-Es2 Emersione del Massimo

Avete visto a lezione un esercizio simile (ma più semplice). Completate il programma, assicurandovi che passi tutti i test proposti.

Selezione operazioni

Scrivere un programma **selezione_operazioni.c** che:

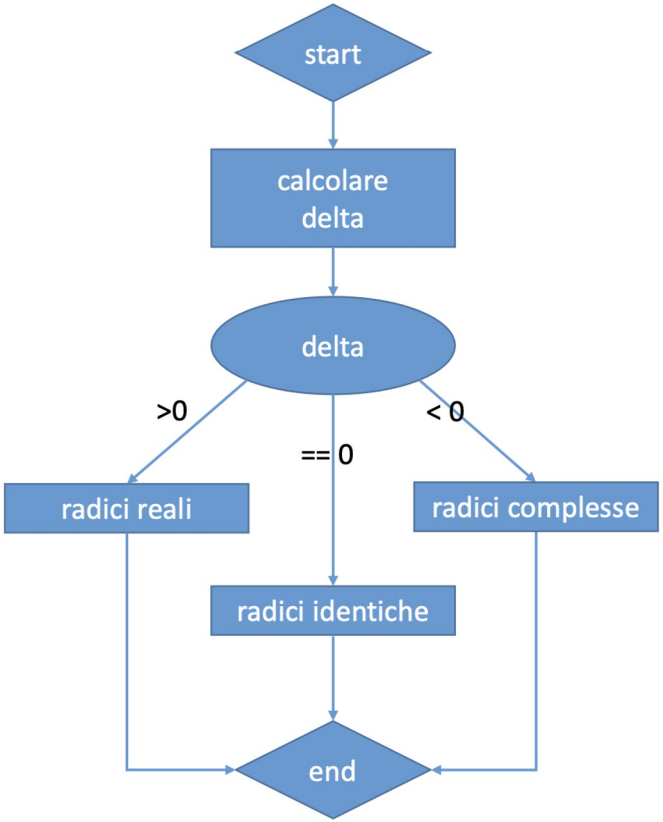
1. Chiede all'utente di inserire tre numeri interi **x1**, **x2** e **x3**.
2. Stampa il seguente testo:
   a. "scrivi **1** per calcolare la somma"
   b. "scrivi **2** per calcolare il prodotto"
   c. "scrivi **3** per calcolare la media"
3. Chiede all'utente di inserire un numero (tra **1**, **2** o **3**) corrispondente ad un'**operazione**.
4. Se l'**operazione** selezionata non è tra quelle proposte, deve essere stampato a video un messaggio di errore appropriato.
5. Altrimenti effettua l'**operazione** selezionata sugli interi **x1**, **x2** e **x3**, e visualizza il risultato ottenuto.

Risoluzione equazioni di secondo grado

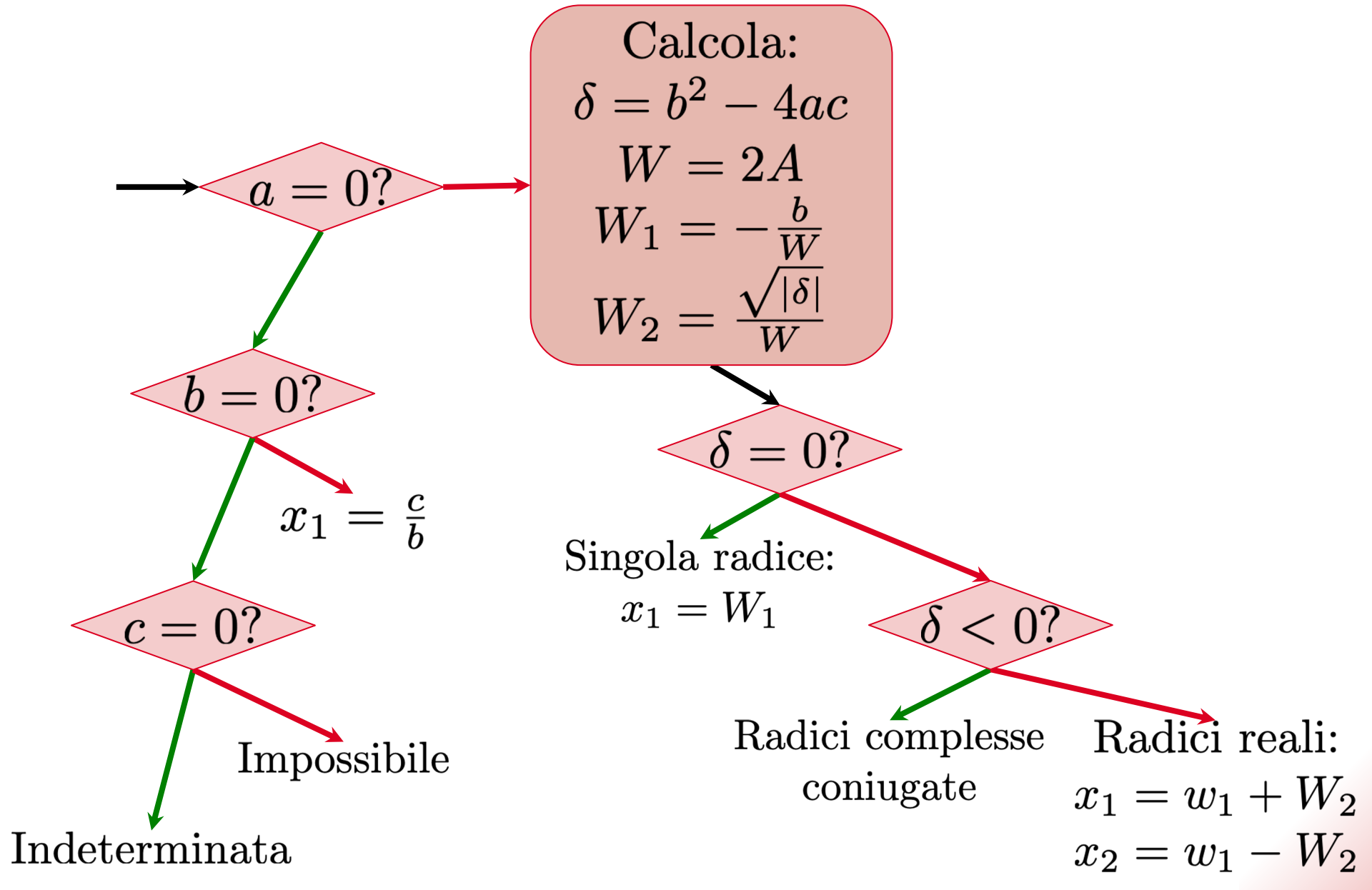
Scrivere un programma **equaz_secondo.c** che prende in ingresso i tre coefficienti **a**, **b** e **c**, e calcola, ove possibile, le radici dell'equazione di secondo grado

$$ax^2 + bx + c = 0$$

e le stampa. Quando non possibile, stampa invece un messaggio di errore appropriato, e termina.



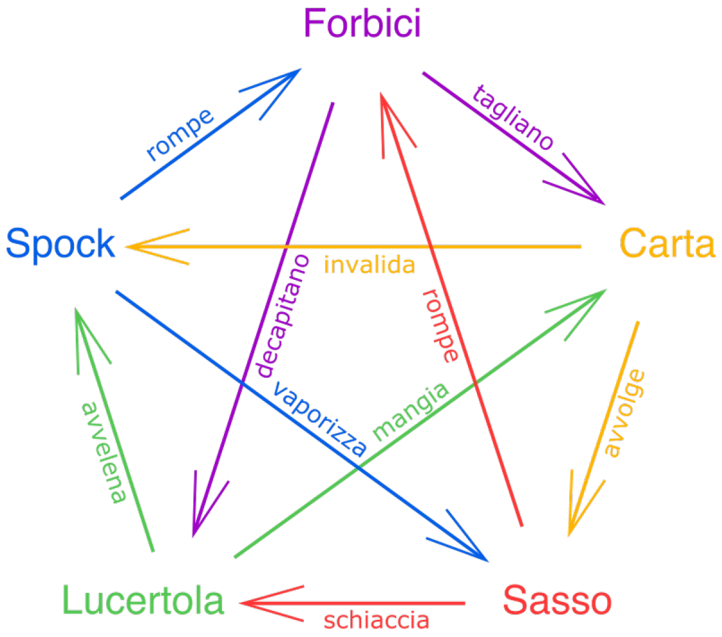
Flusso risoluzione: $ax^2 + bx + c = 0$

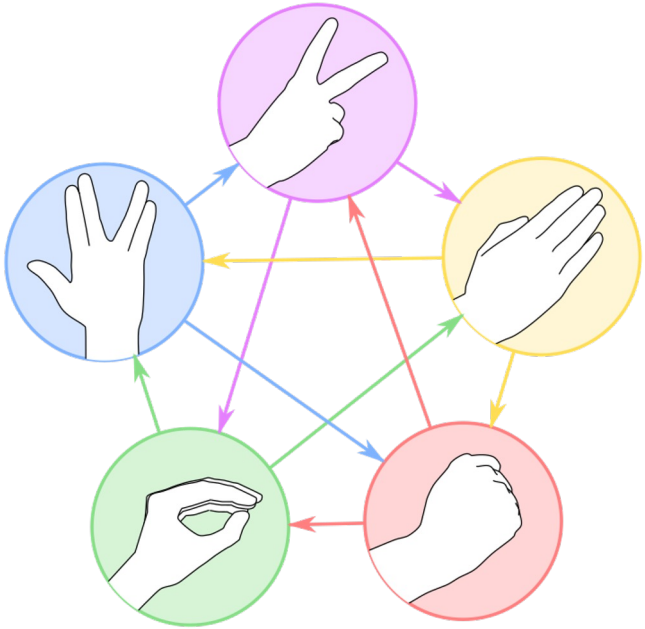


Sasso-carta-forbici-lucertola-Spock

ESERCIZIO EXTRA

- **Sasso-carta-forbici-lucertola-Spock** (*rock-paper-scissors-lizard-Spock*) è una variante fantasiosa inventata da due studenti statunitensi, Sam Kass e Karen Bryla, e resa successivamente famosa nel telefilm The Big Bang Theory[1].
- È un'estensione a cinque stati della Morra Cinese, in cui vengono aggiunti la lucertola e Spock (celebre personaggio della saga Star Trek).





[1] Da Wikipedia Italia. https://it.wikipedia.org/wiki/Morra_cinese

Sasso-carta-forbici-lucertola-Spock

ESERCIZIO EXTRA

Sulla pagina Moodle trovate un esercizio con nome

Lab03-Es3 Sasso-carta-forbici-lucertola-Spock

Seguite il testo e completate l'esercizio.

NOTA: come fare a determinare se "*gli input non sono stati letti*"? Avete visto a lezione che la funzione **scanf** ritorna il numero di specificatori di conversione che sono stati letti con successo dallo standard input.