Laboratorio di programmazione

Lezione V

Tatiana Zolo

tatiana.zolo@libero.it

SWITCH

- Metodo alternativo ad istruzioni `if-else` annidate per scegliere fra un insieme di opzioni mutuamente esclusive.

```
switch (espressione) {
case C1:
  S1;
  break;
case C2:
  S2;
  break;
...
default:
  Sk;
  break;
}
```

- Attenzione: l'esecuzione *comincia* dall'etichetta `case` corrispondente allo `switch` e continua attraverso le successive $\Longrightarrow$ usare `break`!

SWITCH

- Se due o più valori devono essere gestiti dalla stessa sequenza di azioni, si può scrivere:

```
switch (espressione) {
case C1:
case C2:
case C3:
  S1;
  break;
...
default:
  Sk;
  break;
}
```

SWITCH

- L'`espressione` che viene valutata ed utilizzata per la selezione può essere una qualsiasi espressione C++ che restituisce un valore intero.
- I valori specificati nei `case`, che saranno confrontati con l'`espressione`, devono invece essere costanti (noti a tempo di compilazione): non possono essere espressioni che fanno riferimento a variabili.
- es.`mesi.cpp`.

STRINGHE DI CARATTERI STILE C

Definizione

- Costanti stringa: insiemi di caratteri racchiusi tra apici doppi.
- Array di `char`: per trattare come una sola unità un insieme di caratteri.

```
char str1[10];
char str2[] = ''come stai?'';
```

`str1` è un array di 10 caratteri; `str2` ha un numero di elementi determinato dalla quantità di caratteri presenti tra doppi apici più uno: il carattere *null* (`\0`) che chiude la stringa.

`\0` $\Longrightarrow$ permette di usare le stringhe senza conoscerne a priori la dimensione.

es. `prova_str.cpp`.

STRINGHE DI CARATTERI STILE C

Lettura di una stringa da tastiera

- `cin`: attenzione ai caratteri di spaziatura (spazi, tab e newline)!
- `getline()`: sintassi

```
cin.getline(string, size, delimiter = '\n');
```

  - `string` è l'array di caratteri in cui sono posti i caratteri letti;
  - `size` è il max numero di caratteri da leggere dallo stream;
  - `delimiter` indica un carattere delimitatore (per default è l'invio).

  Continua a leggere caratteri finché sono stati letti `size-1` caratteri oppure non si è incontrato il carattere delimitatore.

  `gcount()`: restituisce il numero di caratteri letto dall'ultima chiamata di `getline()`.

- es. `prova_str.cpp`, `prova_str1.cpp`, `prova_str2.cpp`.

STRINGHE DI CARATTERI STILE C

Funzioni di libreria: includere il file header C associato `#include <cstring>` (attenzione `<string>` è un'altra cosa: si riferisce al tipo di dato astratto `string`!).

Tra le funzioni più comuni:

- `strcpy(r, s)`: copia il contenuto della stringa `s` in `r`.
- `strcat(r, s)`: aggiunge `s` alla fine di `r`.
- `strcmp(r, s)`: confronta `s` e `r`. Se sono uguali restituisce 0; se `s` è maggiore di `r` lessicograficamente restituisce un valore positivo, altrimenti un valore negativo.
- `strlen(s)`: restituisce la lunghezza della stringa `s`.
- es.`copia.cpp`.

ESERCIZI

1. switch: considerare due numeri complessi rappresentati come array di 2 elementi (prima cella per la parte reale e seconda per la parte immaginaria). Immettere da tastiera parte reale e parte immaginaria di ciascun numero ed il simbolo di operazione binaria che si vuole compiere con i due complessi. Stabilire quale operazione fare e visualizzarne il risultato (`operaz_complex.cpp`).
2. stringhe: stabilire una password e memorizzarla in una stringa. Scrivere un programma che chieda all'utente di inserire la password e che ne confermi o meno l'esattezza. Permettere al più tre tentativi prima di negargli la possibilita' di riprovare (`password.cpp`).

ESERCIZI

3. stringhe: conversione stringhe –> numeri interi. Scrivere un programma che legga una sequenza di caratteri terminata da "a capo" rappresentante una costante intera (eventualmente con segno '-'). Se la sequenza e' sintatticamente corretta memorizzare il corrispondente numero intero in una variabile di tipo 'int' e visualizzarlo; altrimenti visualizzare un opportuno messaggio (`str_num.cpp`). Es. L'utente inserisce i caratteri `-`, `1`, `5` e il programma deve riconoscere che si tratta dell'intero -15.
4. stringhe: conversione stringhe –> numeri interi. Ripetere l'esercizio precedente dove, invece di considerare una sequenza di caratteri singoli, si considera in input una stringa rappresentante una costante intera (`str_num2.cpp`). Es. L'utente inserisce la stringa `''1082 ''` e il programma deve riconoscere che si tratta dell'intero 1082.

APPENDICE: OPERAZIONE TRA NUMERI COMPLESSI

$z \in \mathbb{C}$: $z = a + ib$, dove $a, b \in \mathbb{R}$ e i è l'unità immaginaria.

Possiamo considerare il numero complesso come una coppia ordinata (a, b) di numeri reali. Esempi:

$$
\begin{aligned}
-3 + 2i &\rightsquigarrow (-3, 2), \\
4 &\rightsquigarrow (4, 0), && \text{(numero complesso reale)}, \\
-5i &\rightsquigarrow (0, -5), && \text{(numero complesso puramente immaginario)}, \\
i &\rightsquigarrow (0, 1).
\end{aligned}
$$

- Addizione: $(a, b) + (c, d) = (a + c, b + d)$.
- Sottrazione: $(a, b) - (c, d) = (a - c, b - d)$.
- Moltiplicazione: $(a, b) \cdot (c, d) = (ac - bd, ad + bc)$.
- Quoziente: $\dfrac{(a, b)}{(c, d)} = \left(\dfrac{ac + bd}{c^2 + d^2}, \dfrac{bc - ad}{c^2 + d^2} \right)$.