Esercitazione 4.10.2019

Programmazione 1

Esercizio 1

- Si scriva un programma in linguaggio C per verificare se una data immessa da tastiera nel formato *ggmmaaaa* è corretta (secondo il calendario gregoriano).
- Il programma deve leggere un valore intero *data*, determinare se corrisponde ad una data corretta, e stampare un messaggio che dica "data corretta" oppure "data non valida" secondo i casi.

Nel calendario gregoriano, introdotto a partire dall'**ottobre 1582**, un anno normale è composto da 365 giorni. Poiché la durata effettiva di un anno siderale, ovvero il tempo impiegato dalla Terra per compiere una rivoluzione completa intorno al Sole, è in realtà di 365,25635 giorni, ogni quattro anni viene introdotto un "anno bisestile" di 366 giorni che consente di compensare il margine di errore introdotto da tre anni normali. In linea di massima, qualsiasi anno perfettamente divisibile per 4 è un anno bisestile. Ad esempio, 1988, 1992 e 1996 sono anni bisestili.

Ma….

- Rimane tuttavia un altro piccolo errore da considerare. Per eliminare questo errore, il calendario gregoriano prevede che ogni anno perfettamente divisibile per 100, ad esempio 1900, sia un anno bisestile solo se perfettamente divisibile anche per 400.
- Per questa ragione, gli anni elencati di seguito non sono anni bisestili:
- 1700, 1800, 1900, 2100, 2200, 2300, 2500, 2600
- in quanto perfettamente divisibili per 100 ma non per 400.
- Mentre gli anni elencati di seguito sono anni bisestili:
- 1600, 2000, 2400
- in quanto perfettamente divisibili per 100 e anche per 400.

Analisi e specifica

- **Dati di ingresso: *data***
- **Precondizione: *data*>= 1101582 e *data*<= 31129999**
- **Dati di uscita: *valido***
- **Postcondizione: se *data* è un intero corrispondente ad una data corretta, *valido* ha valore VERO, altrimenti *valido* ha valore FALSO**

Attributo	*Tipo*	*Descrizione*
data	intero	Dato di ingresso nel formato *ggmmaaaa*
valido	booleano	Dato in uscita che indica la correttezza della data

Progettazione Algoritmo (1)

- Prendi in input un intero *data*
- Se la precondizione 1101582 <= *data* <= 31129999 è verificata

  anno = **aaaa (*data*)**

  mese = **mm (data)**

  giorno = **gg (data)**

  valido = **verifica (anno, mese, giorno)**
- Se la precondizione non è verificata, *valido* = FALSO

- Se *valido* ha valore VERO, stampa "la data è corretta", altrimenti stampa "la data non è valida"

Progettazione Algoritmo (2)

Raffiniamo il passo: ***anno* = aaaa (*data*)**

Analisi e specifica della funzione aaaa( )

- **Dati di ingresso: *data***
- **Precondizione: *data*>= 1101582 e *data*<= 31129999 (già verificata)**
- **Dati di uscita: *anno***
- **Postcondizione: *anno* è un intero corrispondente alle ultime 4 cifre di *data***

Progettazione Algoritmo (3)

Raffiniamo il passo: ***valido* = verifica (*anno, mese, giorno)***

Analisi e specifica della funzione verifica( )

- **Dati di ingresso: 3 interi *anno, mese, giorno***
- **Precondizione: nessuna (già verificata)**
- **Dati di uscita: *valido***
- **Postcondizione: *valido* è VERO se valgono tutte le seguenti 3 condizioni:**
  - ***a)*** ***anno* è compreso fra 1582 e 9999**
  - ***b)*** ***mese* è compreso fra 1 e 12**
  - ***c)*** ***giorno* è compreso fra 1 e:**
    - **i. 31 se mese è uguale a 1, 3, 5, 7, 8, 10, 12**
    - **ii. 30 se mese è uguale a 4, 6, 9, 11**
    - **iii. 28 se mese è uguale a 2 oppure 29 se l'anno è bisestile**

altrimenti *valido* è FALSO

Progettazione Algoritmo (4)

Progettazione della funzione verifica( )

- **verifica( ) prende in ingresso 3 interi *anno, mese, giorno***
- **Se valgono tutte le seguenti 3 condizioni:**

  ***a) anno* è compreso fra 1582 e 9999**

  ***b) mese* è compreso fra 1 e 12**

  ***c) giorno* è compreso fra 1 e:**

  i. **31 se mese è uguale a 1, 3, 5, 7, 8, 10, 12**
  ii. **30 se mese è uguale a 4, 6, 9, 11**
  iii. **28 se mese è uguale a 2 oppure 29 se bisestile(*anno*) è VERO**

  assegna a *valido* il valore VERO altrimenti assegna FALSO
- **Restituisci *valido***

Progettazione Algoritmo (5)

Raffiniamo il passo: ***se* bisestile(*anno)* è VERO**

Analisi e specifica della funzione bisestile( )

- **Dati di ingresso: un intero *anno***
- **Precondizione: nessuna (già verificata)**
- **Dati di uscita: *bis***
- **Postcondizione: *bis* è VERO se vale una delle seguenti 2 condizioni:**
  - ***a)*** ***anno* è divisibile per 4 ma non per 100**
  - ***b)*** ***anno* è divisibile per 400**

  altrimenti *bis* è FALSO

Progettazione Algoritmo (6)

Progettazione della funzione bisestile( )

- **bisestile( ) prende in ingresso un intero *anno***

- **Se vale una delle seguenti 2 condizioni:**

  ***a) anno* è divisibile per 4 ma non per 100**

  ***b) anno* è divisibile per 400**

  assegna a *bis* il valore VERO altrimenti assegna FALSO

- **Restituisci *bis***

Completare la analisi, specifica e progettazione delle funzioni rimanenti:

- **mm (*data*)**

- **gg (*data*)**

…. e adesso:

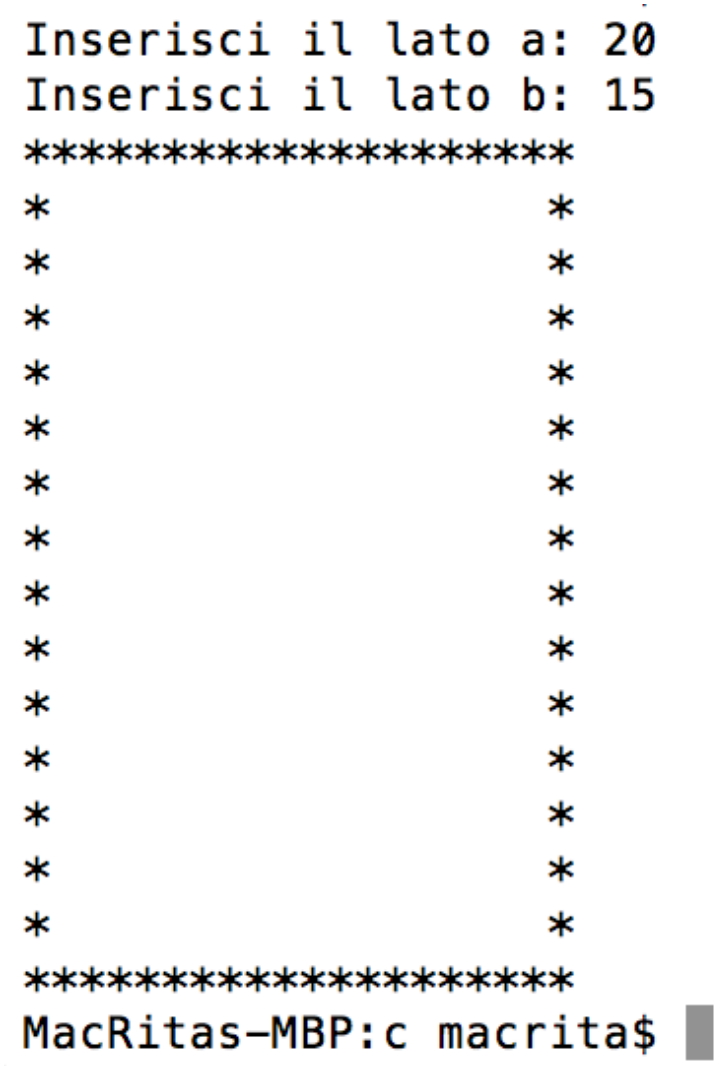
CODIFICA

ESERCIZIO 2 (per casa)

- Scrivere un programma C che riceva in input una sequenza di N numeri, N>0 inserito dall'utente, rappresentanti anni e decida se ogni anno sia o meno bisestile.

Esercizio 3 (per casa)

Chiedere 2 num. interi (a,b) e disegnare un rettangolo di dimensioni a*b usando il carattere " * ".



Esercizio 4 (per casa)

- Scrivere un programma C che legga N numeri, N>1, e stabilisca se la sequenza dei numeri inseriti sia crescente, decrescente oppure né crescente né decrescente.
- *Suggerimento.* Una sequenza è crescente se ogni numero è maggiore del precedente, decrescente se ogni numero è minore del precedente, né crescente né decrescente in tutti gli altri casi.

Altri esercizi per casa

- Scrivere un programma C che legga in input la lunghezza della base di un triangolo e lo visualizzi come in figura.

- Estendere il

programma precedente

Per generare un rombo