Esercitazioni di Algoritmi e Strutture Dati

I esercitazione, 2/03/2016

Tong Liu

OBIETTIVI DEGLI ESERCITAZIONI

- Comprendere meglio i concetti fondamentali
- Presentare gli argomenti rilevanti
- Arrivare al livello richiesto dall'esame

CARATTERISTICHE DI ESAME

- Solo lo scritto (3 ore) - 12 CFU
- Pseudocodice e complessità
- Testo esame: solo descrizione poco suggerimento (Padronanza delle tecniche di algoritmi richiesta)

LINGUAGGI DI PROGRAMMAZIONE

- Fortran 1957, Lisp 1959, Simula 1967, Pascal 1970, C 1972, Prolog 1972, Objective-c 1983, C++ 1986,Python 1989,Java 1995, Javascript 1995, php 1995, Swift 2014 …

LINGUAGGI DI PROGRAMMAZIONE

- **Caratteristiche comuni:**
    - **variabili (tipi, puntatori, reference)**
    - **istruzioni (ciclo fetch-execute)**
    - **espressione: combinazione di variabili, costanti, operatori**
    - **strutture di controllo: permettono di governare il flusso dell'esecuzione del programma**
    - **sottoprogramma - sottoprocedura - funzioni - classi**
    - **strutture dati: array, trees, trees bilanciati, heap, vector, hash tables, associaton lists, dictionaries, sets**
    - **funzionalità aggiunte: input, output, ...**

PSEUDOCODICE

- Descrizione di alto livello sul procedimento di esecuzione.
- Struttura convenzionale (lingua naturale, notazioni compatte matematici. Omette i dettagli come dichiarazione variabili, sotto procedure …)
- Comprensibilità e leggibilità per gli umani
- Utilizzato per la bozza dell'idea prima dello sviluppo effettivo.
- No standard assoluto

PSEUDOCODICE: OPERAZIONI

Tipi di operazioni	**Simboli**	**Esempi**
Assegnamento	← or :=	$c \leftarrow 2\pi r$, $c := 2\pi r$
Confronto	<, >, =, ≠, ≤, ≥	
Aritmetica	+, −, ×, /, mod, !, sin()	
Floor/ceiling	$\lfloor$, $\rfloor$, $\lceil$, $\rceil$	$a \leftarrow \lfloor b \rfloor + \lceil c \rceil$
Logica	**and**, **or, &&, ll**	
Sommatoria, Prodotto	∑ ∏	$h \leftarrow \sum_{a \in A} 1/a$

PSEUDOCODICE: COSTRUTTI

- **if … then else**
- **while … do**
- **for … to … do**
- **for … downto … do**
- **iif(cond,v_2,v_2) equivale a cond ? v_1 : v_2**
- **foreach … do**

```
integer sum(integer[] A, integer i, integer j)
    integer sum ← 0
    for k ← i to j do
        sum ← sum + A[k]
    return sum
```

commenti %
tipi di dato primitivo: integer, real, boolean
tipo variabili dichiarate al primo utilizzo

PSEUDOCODICE:TIPI DI DATI

array (multidimensionali) con partenza da 1 o A[0..n-1]

struttura

. : per accedere ai campi (p.x, p.y , e.g. p punto in piano cartesiano)
new : per crearle - possono essere definite funzioni ad-hoc
delete :per cancellarle (delete A)
nil : per il valore nullo (A[x] := nil)

ESERCIZIO 1: ALGORITMO DI EUCLIDE

```
int Euclid_MCD_Subtraction(int a, int b)
{
    a = abs(a);
    b = abs(b);
    if (a != 0)
    {
        while (b != 0)
        {
            if (a > b)
            {
                a = a - b;
            }
            else
            {
                b = b - a;
            }
        }
    }
    return a;
}
```

MCD (24, 14) = 2

$24 = 2^3 \cdot 3$
$14 = 2 \cdot 7$

Algoritmo Sottrazioni successive

$24 - 14 = 10$
$14 - 10 = 4$
$10 - 4 = 6$
$6 - 4 = 2$
$4 - 2 = 2$
$2 - 2 = 0$

quindi MCD(24, 14) = 2

ESERCIZIO 1: ALGORITMO DI EUCLIDE

```
int Euclid_MCD_Division(int a, int b)
{
     int temp;
     while(b!=0)
     {
          temp = b;
          b = a % b;
          a = temp;
     }
     return abs(a);
}
```

MCD (24, 14) = 2 = 2

ricorda che:
m.c.m. (24, 13) = $2^3 \cdot 3 \cdot 7$

Algoritmo Divisioni successive

24 : **14** = 1 resto **10**
14 : **10** = 1 resto **4**
10 : **4** = 2 resto 2
4 : **2** = 2 **resto 0**

Quindi MCD(24, 14) = 2

ESERCIZIO 1: ALGORITMO DI EUCLIDE

Come viene rappresentato in pseudocodice?

ESERCIZIO 2: SOMMA MASSIMALE

Input: un vettore di interi A[1…n]

Output: la somma del sottovettore A[i…j] che ha somma massimale, ovvero il sottovettore la cui somma degli elementi (tutti gli elementi e consecutivi) è più grande o uguale alla somma degli elementi di qualunque altro sottovettore

Si scriva lo pseudo-codice di un algoritmo che risolve il problema e si invita di giustificare anche la complessità.

E.g. : -6, -9, 2, 5, -5, 8, 7, 3, -9, 2 ⟶ -6, -9, **2, 5, -5, 8, 7, 3**, -9, 2

fa 20

ESERCIZIO 2: SOMMA MASSIMALE

Soluzione 1

```
integer maxsum(integer[] A, integer n)
    integer max ← 0
    for i ← 1 to n do
        for j ← i to n do
            integer sum ← sum(A, i, j)
            max ← max(sum, max)
    return max
```

```
integer sum(integer[] A, integer i, integer j)
    integer sum ← 0
    for k ← i to j do
        sum ← sum + A[k]
    return sum
```

ESERCIZIO 2: SOMMA MASSIMALE

Soluzione 2

```
integer maxsum(integer[] A, integer n)
    integer max ← 0                                  % Massimo valore trovato
    for i ← 1 to n do
        integer sum ← 0                              % Somma sottovettore
        for j ← i to n do
            sum ← sum + A[j]
            if sum > max then
                max ← sum
    return max
```

Un miglioramento è assegnare alla variabile *max* meno infinito o il primo elemento dell'array A per adattare l'algoritmo al vettore con elementi tutti negativi (suggerimento da un vostro collega in lezione).

ESERCIZIO 3: SEARCH

Definire e inizializzare un array di N numeri interi *distinti* (la scelta di N e degli elementi dell'array è arbitraria).

Definire una funzione search che, presi in input un array A e un numero intero x, restituisca:

- i, se l'(i+1)-esimo elemento di A è x (cioè, A[i] == x);
- -1, se l'elemento x non occorre in A (cioè, per ogni 0 <= i < N, A[i] != x).

Domande?

Comunicazione: L'esercitazione di Giovedì 10 Marzo è spostata a Lunedì 14 Marzo presso l'Aula Ercolani 2, dalle 16:30 alle 18:30

ALCUNI RIFERIMENTI

- Pseudocode: https://en.wikipedia.org/wiki/Pseudocode
- Algoritmo di Euclide(ITA): https://it.wikipedia.org/wiki/Algoritmo_di_Euclide
- Algoritmo di Euclide(EN): https://en.wikipedia.org/wiki/Euclidean_algorithm