

## 008AA – ALGORITMICA E LABORATORIO

Appello del 22 febbraio 2011

### SOLUZIONI

#### Esercizio 1.

Dato un array  $a$  contenente  $n$  interi positivi, si deve stabilire se la somma di tutti gli elementi di  $a$  sia pari o dispari senza eseguire addizioni, ma applicando l'operatore aritmetico modulo e gli operatori booleani.

1. Progettare un algoritmo divide et impera per risolvere il problema.
2. Indicare e risolvere la relazione di ricorrenza associata alla sua complessità in tempo.

#### Soluzione

```
SommaPari(a, sx, dx) //restituisce TRUE se la somma è pari, FALSE altrimenti
    if (sx == dx) return (a[sx] % 2 == 0);
    centro = (sx + dx) / 2;
    RisSx = SommaPari(a, sx, centro);
    RisDx = SommaPari(a, centro + 1, dx);
    return RisSx == RisDx;
```

#### Costo in tempo:

$$T(n) = 2 T(n/2) + O(1),$$

Teorema Principale, primo caso:  $T(n) = O(n)$

#### Esercizio 2.

Fornire la definizione di albero binario di ricerca AVL, e descrivere in pseudocodice l'algoritmo che calcola la chiave massima in esso contenuta, discutendone la complessità al caso pessimo.

#### Soluzione

*Si consulti il libro di testo.*

#### Esercizio 3.

Date le stringhe  $A = \text{GUARDA}$  e  $B = \text{UGANDA}$ , simulare l'algoritmo di programmazione dinamica per il calcolo della distanza tra due stringhe mostrando

1. il contenuto della tabella che l'algoritmo riempie dinamicamente;
2. la soluzione trovata dall'algoritmo.

## Soluzione

1. Tabella di programmazione dinamica:

	∅	U	G	A	N	D	A
∅	0	1	2	3	4	5	6
G	1	1	1	2	3	4	5
U	2	1	2	2	3	4	5
A	3	2	2	2	3	4	4
R	4	3	3	3	3	4	5
D	5	4	4	4	4	3	4
A	6	5	5	4	5	4	<b>3</b>

2. La distanza tra le due stringhe calcolata dall'algoritmo è 3.

## Esercizio 4.

Scrivere una funzione C che controlla se un intero inserito a tempo d'esecuzione è primo oppure no.

## Soluzione

```
#include <stdio.h>
#include <math.h>
#define FALSE 0
#define TRUE 1

/*Controlla se un numero e' primo*/

main()
{
    int num, div_cor = 2, e_primo = TRUE;

    printf("Dare il numero da controllare e poi andare a capo\n");
    scanf("%d",&num);

    if(num < 0)
        printf("Il numero doveva essere positivo\n");
    else{if(num >1)
        while(div_cor < num && e_primo)
            if(num % div_cor != 0)
                div_cor++;
            else
                e_primo = FALSE;
        printf("Il numero %d %s e\' primo,\n",num,e_primo ? "" : "non");
    };
}
```