

008AA – ALGORITMICA E LABORATORIO

Verifica dell'8 aprile 2010

Cognome Nome:

N. Matricola:

Anno:

Superato Progr. 1: SI - NO

Esercizio 1.

Dato un array A di n interi scrivere in C una funzione `min()` che, *ricorsivamente*, restituisce il minimo di A in tempo $O(n)$. Verificare tale complessità tramite l'impostazione e la risoluzione, mediante metodo *iterativo*, della relativa equazione di ricorrenza.

Riportare anche la funzione `main()` che invoca `min`.

```
#include<stdio.h>

int min (int a[], int i, int j){
    int minL;

    int size = j-i+1;
    printf("%d %d %d\n",i, j, size);
    if (j<i) return;
    if (j==i) return a[i];
    int m = size/2;
    int low = min(a,i,i+m-1);
    int up = min(a,i+m,j);
    if (low<up) minL=low;
        else minL=up;
    return minL;
}

main(){
int a[8]={27,32,-13,222,5,98,102,-13098};
printf("%d\n",min(a,0,7));
}
```

Cognome Nome:

N.Matr:

Esercizio 2.

Sia data la funzione ricorsiva:

```
Pippo( n ) {
    if (n < 50) return (n+1);
    x = 0;
    for( i = 0; i < n; i = i+1 ) {
        y = n;
        while (y > 1) {
            x = x + y;
            y = floor(y/2);
        }
    }
    return (x + Pippo(n/2));
}
```

dove `floor(z)` restituisce l'approssimazione inferiore dell'intero z passato come argomento.

- Scrivere la relazione di ricorrenza $T(n)$ per la complessità in tempo al caso pessimo della funzione Pippo.
- Trovare la soluzione per $T(n)$ in forma chiusa.

Cognome Nome:

N.Matr:

Esercizio 3.

Sia dato un vettore A di n interi, alcuni dei quali possono essere ripetuti, ordinato in modo non decrescente. Si progetti un algoritmo *efficiente* che, ricevuto in ingresso A e un intero k , conta il numero *occ* di occorrenze di k in A .

Cognome Nome:

N.Matr:

Esercizio 4.

Data la sequenza di chiavi intere 20, 30, 80, 40, 50, 35, 16, 5, costruire un albero AVL per inserzioni successive, disegnando l'albero dopo l'inserzione di ogni chiave e indicando, prima di ogni rotazione, il nodo critico e il tipo di rotazione eseguita.