Teoria

Dato il seguente array bidimensionale:

```
int A[4][5];
```

Scrivere un'espressione che, usando l'aritmetica dei puntatori, acceda all'elemento `A[2][3]`.

Data la seguente funzione ricorsiva:

```
int f(int n) {
    if (n <= 1) return 1;
    return n * f(n-2);
}
```

- Qual è il parametro su cui viene fatta la ricorsione?
- Qual è la misura di complessità del problema?
- Spiegare perché questa misura decresce ad ogni chiamata ricorsiva.
- Determinare il fattore minimo di decrescita della misura di complessità ad ogni chiamata ricorsiva.

Considerata la seguente struttura:

```
struct punto {
    int x: 5;
    int y: 5;
    int z: 6;
};
```

Quanto vale sizeof(struct punto)?

Assumendo che un int occupi 4 byte, cosa stampa il seguente codice?

```
int arr[3][4] = {{1,2,3,4}, {5,6,7,8}, {9,10,11,12}};

printf("%d", *(*(arr+1)+2));
```

Data la funzione:

```
int T(int n) {
    if (n <= 2) return n;
    return T(n-1) + T(n-2) + T(n-3);
}
```

- Calcolare T(5)
- Qual è la complessità temporale di questa funzione in termini di notazione O-grande?

Dato il seguente array bidimensionale:

```
char str[3][4] = {"ABC", "DEF", "GHI"};
```

Scrivere un'espressione che, usando l'aritmetica dei puntatori, acceda all'elemento 'F'.

Considerando la seguente struttura:

```
struct packet {
    unsigned int type : 4;
    unsigned int priority : 3;
    unsigned int : 1;
    unsigned int data : 24;
};
```

Quanto vale sizeof(struct packet)?

Considerando il seguente codice:

```
#define MAX(a,b) ((a) > (b) ? (a) : (b))

int x = 5, y = 10;

printf("%d", MAX(x++, y++));
```

Cosa viene stampato? Quali sono i valori di x e y dopo l'esecuzione?

Considerare la seguente funzione:

```
void f(int *p, int n) {
    if (n <= 1) return;
    f(p+1, n-2);
    *p = *(p+n-1);
}
```

Cosa fa questa funzione? Descrivere brevemente il suo effetto su un array.

Data la seguente dichiarazione:

```
int (*f)(int, char*);
```

Cosa rappresenta f? Come si potrebbe assegnare una funzione a questo puntatore?

Considera il seguente codice:

```
int add(int a, int b) { return a + b; }
int subtract(int a, int b) { return a - b; }
int operate(int (*op)(int, int), int x, int y) {
    return op(x, y);
}
```

Come si chiamerebbe la funzione `operate` per eseguire un'addizione? E per una sottrazione?

Esercizi

Implementare una funzione che, data una lista concatenata e un valore pivot, riordini la lista in modo che tutti i nodi con valore minore del pivot vengano spostati all'inizio della lista, seguiti dai nodi con valore uguale al pivot, e infine dai nodi con valore maggiore del pivot. La funzione deve avere la seguente firma:

```
void riordina_lista_pivot(Lista **head, int pivot);
```

Dove Lista è definita come:

```
struct nodo {
    int valore;
    struct nodo *next;
};
typedef struct nodo Lista;
```

L'ordine relativo degli elementi all'interno di ciascun gruppo (minore, uguale, maggiore) deve essere mantenuto.

Esempio:

Input: 3 -> 5 -> 8 -> 2 -> 1 -> 9 -> 4 -> 5 -> 7, pivot = 5

Output: 3 -> 2 -> 1 -> 4 -> 5 -> 5 -> 8 -> 9 -> 7

Implementare la funzione in modo efficiente, discutendo la strategia utilizzata e analizzando la complessità computazionale.

Implementare una funzione ricorsiva che calcoli la somma dei quadrati dei numeri pari contenuti in un array di interi. La funzione deve avere la seguente firma:

```
int somma_quadrati_pari(int arr[], int size);
```

Dove:

- `arr` è l'array di interi
- `size` è la dimensione dell'array

La funzione deve restituire la somma dei quadrati dei soli numeri pari presenti nell'array.

Esempi:

- Per l'array [1, 2, 3, 4, 5], la funzione deve restituire 20 (2^2 + 4^2 = 4 + 16 = 20)
- Per l'array [1, 3, 5, 7], la funzione deve restituire 0 (nessun numero pari)

Scrivere la funzione in modo ricorsivo, specificando PRE e POST condizioni e discutendone brevemente la correttezza.

Implementare due funzioni:

a) Una funzione che ricostruisca un albero binario data la sua rappresentazione serializzata in forma di stringa. La serializzazione usa '(' per indicare l'inizio di un sottoalbero, ')' per la fine, e ',' come separatore. I valori sono interi.

```
BTree* deserialize(char* str);
```

b) Una funzione che serializzi un albero binario nella stessa forma.

```
char* serialize(BTree* root);
```

Esempio:

L'albero:

1

/

2 3

/

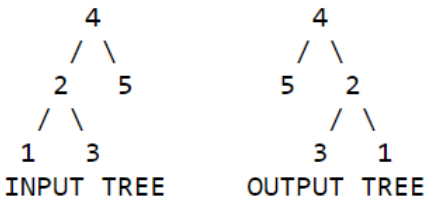
4 5

Sarà rappresentato come: "(1,(2),((3,(4),(5))))"

Scrivi una funzione ricorsiva con firma:

```
void mirror (BST* root)
```

che scambia i puntatori destro e sinistro di un albero. Gli alberi sottostanti riportano la situazioni corretta.



Scrivere una funzione `void min_max(int *A, int dim, int *i, int *j)` che, dato un array A di dim interi, trovi l'indice del minimo (*i) e del massimo (*j) valore di A.

Scrivere una funzione `void mossa_alfiere(int m[][8], int x, int y)` che, data una posizione (x,y) sulla scacchiera, assegni 1 alle posizioni della scacchiera raggiungibili da un alfiere posizionato in (x,y) e 0 alle altre.