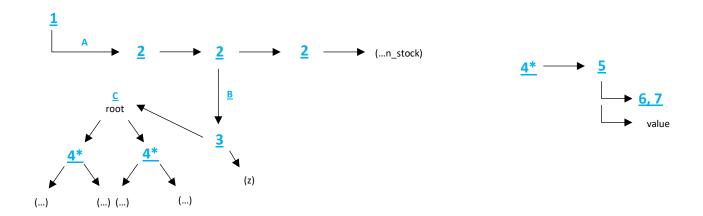
Lab12

```
quotation_BST{
                                                                                                                                                                                 6
              ar name[MAX_L];
                                                                          a BST link root:
            int n_transaction;
stock_link next;
                                                                                                                              struct quotation BST *BST;
                                                                                                               В
                                                                     struct BST_node{
                                                                                                                     typedef struct quotation{
               stock collection{
                                                                         quotation q;
q_BST_link L_next;
q_BST_link R_next;
int N;
<u>1</u>
                                                                                                                         date q_date;
double value;
           int n_stock;
stock_link head;
                                                                                                                      quotation;
           ruct stock* stock link:
        struct stock_collection* stock_list;
```

Schema delle strutture utilizzzate:



Descrizioni algoritmi utilizzati:

- 1. Acquisizione da nuovo file
- 2. Ricerca titolo azionario
- 3. Ricerca quotazione in data\n
- 4. Ricerca quotazione Min e Max, in intervallo date
- 5. Ricerca quotazione quotazione Min e Max totale
- 6. Bilanciamento albero

1. new_file():

La funzione legge il file di input e ritorna un puntatore a lista di stock.

Il calcolo della quotazione giornaliera è stato effettuato tramite una media ponderata, la quale viene aggiornata ad ogni lettura di una nuova transizione nella data.

 $\mathbf{M_{i}}$: media ponderata dopo i transazioni

N_i: numero totale di transizioni in data dopo *i* transazioni

q_i: valore del titolo *i* transazioni, **n**_i: numero di quote acquistate con quotazione **q**_i

Se viene letta una nuova transizione in quella data: $M_{i+1} = (M_i * N_i + q_{i+1} * n_{i+1}) / (N_i + n_{i+1})$

 \mathbf{M}_{i+1} viene calcolata solo dopo che viene letta una data diversa da quella precedente, tramite la funzione is_different_date() (riga 155 "main.c").

Andrea Pellegrino s268631

- Una volta inserito il titolo di una azione, viene cercato all'interno della lista di titoli, tramite uno scorrimento in lista
 (SEARCH_in_list() restituisce il puntatore al titolo azionario con il nome dell'azione).
 Tramite la PRINT_stock(), vengono stampate le informazioni di quel titolo azionario, tramite una visita IN_ORDER della
 BST_print() (riga 88 "quotation.c").
- 5. E' stato sviluppato come caso particolare del punto 3. In particolare vengono create 2 date fittizie (data1: 0000/00/00, data2: 9999/99/99) entro cui stampare la quotazione corrente qi solo se la data di qi è compresa tra data1 e data2. In questo le informazioni di un titolo verranno sempre stampate (la condizione è sempre vera)
- 6. Per bilanciare l'albero, inizialmente vengono calcolate le lunghezze dei cammini MAX e MIN nell'albero, tramite la funzione ricorsiva BST_lenght() (riga 72 "quotation.c")
 La funzione è in grado di calcolare sia MAX che MIN, tramite il parametro M (M=0: calcolo minimo, M=1 calcolo massimo)
 Essa confronta ricorsivamente la lunghezza del sottoalbero SX con quella del sottoalbero DX di un nodo ritornando, in base a M, il MIN o il MAX tra i 2 sottoalberi.

Viene poi effettuato il confronto *max/min* > *S* (soglia), se risultante vero viene effettuato il bilanciamento tramite la funzione **balanceR()** (riga 166 "quotation.c").

Essa opera tramite un bilanciamento ricorsivo di un sottoalbero, se l'albero non è già bilanciato (max/min > S (soglia)). Il bilanciamento viene effettuato facendo uso delle funzioni di partizionamento partR() (riga 142 "quotation.c") e, all'interno di essa, rotazioni a SX e DX rotL(), rotR() (riga 132, 122 "quotation.c).

Commenti generici:

La funzione **key_cpm**(date **k1**, date **k2**) opera un confronto tra date, la quale ritorna un valore in base a quanto differiscono le 2 date:

- return -1: se k1 < k2
- return 6: se k1 == k2 (data e orario coincidono)
- return 0<N<6: se k1 > k2, N aumenta se la differenza di tempo tra k1 e k2 diminuisce