Isabella Priano s258356

Esercizio n. 1 - Laboratorio 11

Il mio programma vuole avere un `main` asciutto senza inclusioni di `<stdlib.h>` e `<stdio.h>` e per tale scopo ho previsto l'utilizzo di `"utility.h"` dove ho raccolto funzioni idonee all'apertura del file scelto e soprattutto per la gestione del client contenente i comandi utili ad ottenere in output i dati desiderati.

Per facilitarmi la scrittura del codice, nello `switch` invece di utilizzare i soliti `case 0, case 1` ecc. ho preferito usare un `typedef enum` dove ho definito i "casi" con gli effettivi nomi delle funzioni richieste.

Dopo aver individuato le operazioni da svolgere nei vari casi dello `switch`, ho provveduto a dare "corpo" al mio programma attraverso le opportune funzioni di base viste in teoria adattandole opportunamente per risolvere il problema proposto nello specifico esercizio.
Più precisamente, seguendo le indicazioni dell'esercizio riguardo la costruzione di una struttura dati a due livelli, ho deciso di "spezzare" i due problemi proposti in `collezione_giornaliera.c` e `collezioni_titoli.c` volendone semplificarne la stesura del codice per la loro gestione. Era comunque possibile unire i due file `.c` in uno solo chiamandolo per esempio `collezioni.c` .

In `collezione_giornaliera.c` si può notare che `COLLEZIONE_Qinit` è molto simile a `void ST_init(){head =(z = NEW(NULLitem, 0, 0, 0));};` dove `NEW` è

```
link NEW( Item item, link l, link r, int N){
 link x = malloc(sizeof(*x));
 x->item = item; x->l = l; x->r = r; x->N = N;
 return x; }
```

l'unica differenza è che io ho preferito tipizzare `NEW` con `static link` mentre `COLLEZIONE_Qinit` con `Collezione`.

Anche poi l'inserimento `Insert(Quotazione q, link h, link z)` (inserimento ricorsivo) ricalca molto la funzione di base

```
insertT(link h, Item item){
key v = key(item);
if(h == z) return NEW(item, z, z, 1);
if(less(v, key(h->item))){
h = l = insertT(h->l, item); h = rotR(h);
else {h->r = insertT(h->r, item); h = rotL(h);}
return h;}
```

Le differenze si possono notare dove vi è l'utilizzo di `less` ; io uso `DATAlower` e l'aggiunta di `ChangeMinMax` al posto di `rotR` e `rotL` che uso poi in `static link partition(link h, int r)`.

Nell'ambito dell'utilizzo delle strutture seguo ciò che viene richiesto:

- un quasi ADT per data e ora in `data.h` come `typedef struct { int g, m, a, min; float ore;}Data;`
- un ADT di I classe per il titolo e uno per la collezione di titoli che si trovano rispettivamente in `classe_titoli.c` come `struct classeTitolo{char nome[N]; collezioni c; };` e in `classe_titoli.h` definito da `typedef struct classeTitolo *ClasseTitolo;`
- un quasi ADT per la quotazione giornaliera e un ADT di I classe per collezione di quotazioni giornaliere rispettivamente si trovano in `quotazione_giornaliera.h` definita come `typedef struct {Data d; float q; int n; float min, max;} Quotazione;` messa nel `.h` e non nel `.c` perché usando come tipo delle funzioni Quotazione mi serve definirlo in quel punto, e `collezione_giornaliere.h` come `typedef struct collezione *Collezione;`