#include "list.h"

#include "item.h"

/\*

specifica sintattica:

conta elementi (LISTA, ITEM) -> INT

specifica semantica:

int conta\_elementi(list L, item K)

precondizioni:

lista L non nulla

item K non nullo

L non contiene elementi ripetuti

L = <a0, a1, a2, ..., an>

postcondizioni:

contatore = numero di elementi di L strettamente maggiori di K

\*/

/\*

PROGETTAZIONE:

1.0 per ogni elemento della lista

1.1 controllo se i-esimo elemento è maggiore di k

1.2 allora aumento il contatore

\*/

int conta\_elementi(list L, item K) {

int contatore = 0;

int size = sizeList(L);

for (int i = 0 ; i < size; i++) {

item el = getItem(L, i);

if (maggiore(el, K))

contatore++;

}

return contatore;

}

/\*

specifica sintattica

conta elementi (LIST, ITEM) -> INT

specifica semantica

int conta\_elementi(list L, item K)

precondizioni

lista L non nulla

item K non nullo

L = <a0, a1, a2, ..., an>

postcondizioni

contatore = numero di elementi distinti di L

strettamente maggiori di K (in caso di ripetizioni verranno conteggiati una sola volta)

\*/

/\*

progettazione:

1.0 creo una nuova lista "usciti" che conterrà gli elementi maggiori di K già

controllati

2.0 per ogni elemento della lista l

2.1 se l'i-esimo elemento è maggiore di K e non è presente nella lista degli item già usciti

2.2.1 incremento il contatore

2.2.2 inserisco questo elemento nella lista usciti

3.0 dealloco usciti

4.0 ritorno il contatore

\*/

int conta\_elementi(list L, item K) {

list usciti = newList();

int size = sizeList(L);

int contatore = 0;

for (int i = 0; i < size; i++) {

item el = getItem(L, i);

if (maggiore(el, K) && !searchItem(usciti, el)) {

contatore++;

insertList(usciti, sizeList(usciti), el); // aggiungo alla fine della lista

}

}

freeList(usciti);

return contatore;

}