Documentatie

(problema 5, lab 3-5)

**Nume si prenume:** Mihalache Florin

**Grupa:** 214

# Cerinta

**Michael** is the administrator of a bloc and wants to manage the monthly expenses for each apartment in the bloc. In order to complete this task, Michael needs an application to store, for a certain month, the expenses for each apartment. Each expense will be stored in the application through the following elements: amount, type of the expense (the administrator wants to group the expenses in several predefined categories: such as water, heating, illuminating, gas, others). Michael needs an application in order to repeatedly execute the following functionalities (each functionality is exemplified):

1. Add a new transaction into the list.

* insert 100, type at 25 – inserts at apartment 25 an expense of 100 RON having the given type

2. Modify expenses from the list.

* remove 15 – removes all the expenses at apartment 15
* remove from 5 to 10 – removes all the expenses from apartment 5 to apartment 10
* remove type – removes all the expenses having the indicated type, from all the apartments
* replace 12, type with 200 – replaces the amount for the expense having the specified type at apartment 12 with 200 RON

3. Write the expenses having different properties.

* greater than 100 - writes all the apartments with an overall expense greater than 100RON
* less than 100 before 15 - writes all the apartments with an overall expense less than 100
* for apartments from 1 to 15
* all type – writes all the expenses having the specified type.
* Sold 10 – computes the sold (total amount) for apartment 10

4. Obtain different characteristics of expenses.

* + sum type– writes the total amount for the expenses having the specified type.
  + max 25 – writes the maximum expense per type for apartment 25
  + asc sort apt – sorts the total expenses/apartment in an ascending order
  + desc sort type - sorts the total expenses per type in a descending order

5. Filter.

* + filter type – retains only the expenses having the specified type.
  + filter 300 – retains only the overall expenses greater than 300 RON

6. Undo the last operation.

* + undo – the last operation that has modified the list of expenses is cancelled.

# Prezentare fisiere de tip header

alist.h

**struct** Apartment {

**struct** Expense {

**float** amount;

string type;

};

**int** number; // apartment number

Expense\* eList; // expenses list

**int** eSize;

**int** eCapacity;

/\* Initializes the apartment

\* Post: eSize == 0

\* number == -1

\*/

**Apartment**();

/\* Destroys the apartment \*/

**~Apartment**();

/\* Returns the size of expenses list \*/

**int** **size**();

/\* Adds expense to the list \*/

**void** **add**(string type, **float** amount);

/\* Removes expense by index \*/

**void** **removeByID**(**int** id);

/\* Removes all expenses by type \*/

**void** **removeByType**(string type);

/\* Sorts expenses in an ascending order \*/

**void** **sort**();

};

**struct** aptList {

Apartment\* aList; // apartments list

**int** aSize;

**int** aCapacity;

Apartment\* cpList; // apartments list, for undo

**int** cpSize;

**int** cpCapacity;

/\* Initializes the apartments list

\* Post: aSize == 0

\*/

**aptList**();

/\* Destroys the apartments list \*/

**~aptList**();

/\* Returns the size of apartments list \*/

**int** **size**();

/\* Adds apartment to the list \*/

**void** **add**(**int** number);

/\* Inserts expense at given apartment \*/

**void** **insert**(**int** aptNumber, string type, **float** amount);

/\* Removes all the expenses at given apartment \*/

**void** **remove**(**int** aptNumber);

/\* Removes all the expenses from a range of apartments \*/

**void** **remove**(**int** fromAptNumber, **int** toAptNumber);

/\* Returns a pointer to apartment, by number \*/

Apartment\* **getAptByNumber**(**int** aptNumber);

/\* Removes all expenses of a given type \*/

**void** **removeAllByType**(string type);

/\* Retains only the expenses having the specified type \*/

**void** **filterByType**(string type);

/\* Retains only the overall expenses greater than given amount \*/

**void** **filterGreaterThan**(**float** amount);

/\* Saves a copy of the list, for undo \*/

**void** **saveCopy**();

/\* The last operation that has modified the list of expenses is cancelled. \*/

**void** **undo**();

};

ui.h

/\* Writes all the apartments with an overall expense greater than a given amount \*/

**void** **writeGreaterThan**(aptList\* apts, **float** amount);

/\* Writes apartments from 1 to a given apartment number with an overall expense less than a given amount \*/

**void** **writeLessThan**(aptList\* apts, **float** amount, **int** aptNumber);

/\* Writes all the expenses having the specified type \*/

**void** **writeAllType**(aptList\* apts, string type);

/\* Writes the sold (total amount of expenses) for given apartment \*/

**void** **writeSold**(aptList\* apts, **int** aptNumber);

/\* Writes the total amount for the expenses having the specified type \*/

**void** **writeSumType**(aptList\* apts, string type);

/\* Writes the maximum expense per type for given apartment \*/

**void** **writeMaxType**(aptList\* apts, **int** aptNumber);

/\* Prints menu options \*/

**void** **printMenu**();

/\* The menu \*/

**void** **menu**(aptList\* apts);

/\* Shows all the expenses \*/

**void** **showAll**(aptList\* apts);

/\* UI for adding expense \*/

**void** **addExpense**(aptList\* apts);

/\* UI for modifying expenses \*/

**void** **modifyExpenses**(aptList\* apts);

/\* UI for removing expenses at given apartment \*/

**void** **removeExpAptUi**(aptList\* apts);

/\* UI for removing expenses for a range of apartments \*/

**void** **removeExpRangeUi**(aptList\* apts);

/\* UI for removing all expenses of a given type \*/

**void** **removeAllByTypeUi**(aptList\* apts);

/\* UI for writing expenses having different properties \*/

**void** **writePropertiesUi**(aptList\* apts);

/\* UI for obtaining different characteristics of expenses \*/

**void** **obtainChUi**(aptList\* apts);

/\* UI for filtering the list of expenses \*/

**void** **filterListUi**(aptList\* apts);

/\* UI for undo \*/

**void** **undoUi**(aptList\* apts);

tests.h

**void** **testAll**();

**void** **testInsertion**();

**void** **testRemove**();

**void** **testWritingProperties**();

**void** **testSorting**();

**void** **testFilterByType**();

**void** **testFilterGreaterThan**();