**Aplicatie pentru cheltuieli de familie**

**Enunt:**

Scrieți o aplicație care gestionează cheltuielile de familie efectuate într-o lună.

Pentru o cheltuială se vor retine ziua (din luna), suma, tipul cheltuielii (5 categorii:

mâncare, întreținere, îmbrăcăminte, telefon, altele). Aplicația să permită

efectuarea repetată a următoarelor operații:

**Lista de functionalitati:**

1. *Adaugă cheltuială*

• adaugă o nouă cheltuială (se specifică ziua, suma, tipul)

• actualizează cheltuială (se specifică ziua, suma, tipul)

2. *Ștergere*.

• Șterge toate cheltuielile pentru ziua dată

• Șterge cheltuielile pentru un interval de timp (se dă ziua de început și

sfârșit, se șterg toate cheltuielile pentru perioada dată)

• Șterge toate cheltuielile de un anumit tip

3. *Căutări*.

• Tipărește toate cheltuielile mai mari decât o sumă dată

• Tipărește toate cheltuielile efectuate înainte de o zi dată și mai mici

decât o sumă (se dă suma și ziua, se tipăresc toate cheltuielile mai

mici ca suma dată și efectuate înainte de ziua specificată)

• tipărește toate cheltuielile de un anumit tip.

4. *Rapoarte*.

• Tipărește suma totală pentru un anumit tip de cheltuială

• Găsește ziua în care suma cheltuită e maximă

• Tipărește toate cheltuielile ce au o anumită sumă

• Tipărește cheltuielile sortate după tip

5. *Filtrare*.

• Elimină toate cheltuielile de un anumit tip

• Elimină toate cheltuielile mai mici decât o sumă dată

6. *Undo*

• Reface ultima operație (lista de cheltuieli revine ce exista înainte de

ultima operație care a modificat lista)

**Plan de iteratii:**

***Iteratie 1*** *- Lab 4:*

* Construirea functiilor de baza (citire, actualizare)
* Fragmentarea programului pe module in functie de tipul de functii (**Main.py**-partea main a programului, **TaskSelect.py** -modul care contine functia principala de selectie a functionalitatilor , **Interface.py**-modul care contine partea de mesaje si interfata a programului , **TaskFunctions.py**-modul care cuntine functiile de calcul ale programului, **TaskTesters.py**-modul alcatuit din functii de testare ale functiilor task-urilor)
* Crearea functiilor corespunzatoare cerintelor de la punctele 3 si 5, si a functiilor adiacente sau de baza pentru program

***Iteratie 2*** *- Lab 5:*

* Crearea functiilor corespunzatoare cerintelor de la punctele 2 si 4, impreuna cu functiile de test corespunzatoare

***Iteratie 3*** *- Lab 6:*

* Crearea functiilor corespunzatoare cerintei de la punctul 6, impreuna cu functiile de test aferente

**Scenariu de rulare:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Nr.** | **Utilizator** | **Program** | **Descriere** |
| 1. |  | -initializeaza aplicatie | -Programul apeleaza functia de interfata care prezinta functionalitatile aplicatiei si asteapta input-ul de la utilizator  -inputul este unul din codurile printate anterior, sugerand codul fiecarei functionalitati; daca acesta e gresit, se afiseaza mesaj de eroare si programul asteapta ca input alt cod |
| 2. | In: 1a | -apeleaza prin functia *taskSelecter()* functia *citire()* | - se apeleaza functia de baza de citire si se asteapta de la utilizator inserarea unui input de forma: {**[zi], [suma de bani], [tip\_cheltuiala]**}  - zi- nr natural mai mic decat 32  - suma- nr natural  - tip\_cheltuiala- string din multimea ["mancare", "intretinere", "imbracaminte", "telefon"]  - daca nu se regaseste in acea multime, programul considera ca tipul este echivalent cu tipul “altele”  - in final, variabila *data* va deveni o lista de liste care contin datele cheltuielilor |
| 3. | In: 1b | -apeleaza prin functia *taskSelecter()* functia *actualizare()* | - se apeleaza functia de actualizare a unei cheltuieli si asteapta un input asemanator cu cel de la functia *citire* |
| 4. | In: x |  | - programul se opreste |
| 5 | In: 6 |  | - programul anuleaza ultima operatie executata |

**Nota:** Toate comenzile introduse de user vor trece prin *taskSelecter()* pentru a se apela functiile corespondente fiecarui cod inserat.

Pentru urmatoarele comenzi introduse, se apeleaza functiile corespunzatoare rezolvarii cerintelor 2,3,4,5, (care folosesc la randul lor metodele necesare indeplinirii sarcinii):

* **2a, 2b, 2c** - (*sterge\_cheltuieli\_dupa\_zi(),* respectiv *sterge\_cheltuieli\_din\_perioada()*) pentru cerintele de la subpunctul 2 (OBSERVATIE: pentru codul **2c** se refolosesc functiile folosite la **5a**)
* **3a, 3b, 3c** - (*gaseste\_cheltuieli\_mai\_mari\_decat\_suma ()*, *gaseste\_cheltuieli\_mai\_mici\_decat\_suma\_mai\_vechi\_decat\_zi ()*, respectiv *gaseste\_cheltuieli\_dupa\_tip ()*) pentru cerintele de la subpunctul 3
* **4a, 4b, 4c, 4d** - (*suma\_totala\_tip()*, *zile\_suma\_totala\_max()*, *cheltuieli\_sume\_egale()*, respectiv *sortare\_dupa\_tip()*) pentru cerintele de la subpunctul 4
* **5a, 5b** - (*elimina\_cheltuieli\_dupa\_tip (),* respectiv *elimina\_cheltuieli\_mai\_mici\_decat\_suma ()*) pentru cerintele de la subpunctul 5