Домашно 1

Честито! Вие сте посетител номер 1000000 на тази страница! На вас се пада уникалната възможност да се включите в създаването на най-успешния проект включващ не-block-chain технологията! От вас се иска да направите програма, която ще съхранява и обработва данни за потребители на новата не-анонимна, не-крипто-валута FMICoin, само за студенти!

От вашата програма ще се изисква да може да създава портфейли при подаване на команда `add-wallet **fiatMoney** **name**`, където **`fiatMoney`** е началният баланс (инвестиция) на портфейла в истински пари (левове), а **`name`** е името на притежателя на портфейла, което не надхвърля 255 символа. При създаването на портфейл, трябва да му генерирате уникален идентификатор, който да е цяло *положително* число, по малко от 2^{32} с цел неговата идентификация. Данните за създадените портфейли трябва да съхранявате в бинарен файл с името *wallets.dat*, и при стартиране на вашата програма ако съществува такъв файл, трябва да зареждате от него вече запазената информация.

Трябва да поддържате и списък с извършени трансакции между различните портфейли запазени в програмата като запазвате информацията за тези трансакции в бинарен файл с име *transactions.dat*. Данните за една трансакция са **`senderId`** - уникалният номер на портфейла на изпращача, **`receiverId`** - уникалният номер на портфейла на получателя, **`fmiCoins`** - количеството трансферирани FMICoin-и с тип реално число и **`time`** - цяло число с големина 8 байта, което е броя секунди, изминали от 01.01.1970г 00:00ч. За да стимулираме развитието на иновативната валута, при регистрацията на всеки нов потребител ще захранваме неговата сметка с начално количество FMICoin-и равни на стойността на парите които той инвестира (количеството **`fiatMoney`** подадено на програмата при създаване на портфейл), разделени на 375 (число генерирано от честен генератор на случайни числа, и същевременно определящо константният курс на FMICoin монетите спрямо истинските пари). Тоест при създаване на портфейл, трябва и да създадем една трансакция, която има получател новосъздадения портфейл и количеството изчислени монети, а за изпращач ще записваме номера на системния портфейл - 4294967295 (инцидентно този номер не е валиден за който и да е създаден портфейл).

Няма как да направим успешна не-крипто-валута без начин за търгуване. За това вашата програма трябва да може да съхранява информация за нареждания за покупка и продажба на монети, направени от потребителя. Информацията за едно нареждане съдържа три полета: **`type`** - тип на нареждането, а именно **`SELL`** или **`BUY`**, **`walletId`** - номер на портфейла, който създава нареждането и **`fmiCoins`** - количество на монетите, които са предложени за продажба или покупка. Трябва да можете да създавате нареждания с команда `make-order **type** **fmiCoins** **walletId**`, където **`type`** е едно от SELL или BUY, **`fmiCoins`** е реално число, представляващо количество монети и **`walletId`** е номерът на портфейла на наредителя. Информацията за всички създадени нареждания трябва да съхранявате в бинарен файл с име *orders.dat*.

Имате следните условия при създаване на нареждане за *продажба* на монети:

- Съществува портфейл за подадения номер
- Наредителят (**`walletId`**) трябва да има поне толкова монети, колкото иска да продаде
    - За да разберете какво количество монети притежава един портфейл то трябва да разгледате извършените трансакции, които афектират него (като имате предвид че

началното му състояние от монети идва от трансакцията от системният портфейл с номер 4294967295)
    - Също така трябва да се проверят създадените нареждания от този портфейл, тъй като те са вече “одобрени” и трябва да може да се изпълнят от следващо нареждане.
- За да изпълни текущата заявка за продажба, програмата трябва да провери дали има нареждания за купуване на монети и да удовлетвори някои или всички от тях в реда в който са били направени. Ако се случи така че има недостатъчно нареждания за покупка за да се удовлетвори текущото нареждане, то остатъка от текущото нареждане се записва във файла с гореспоменатия формат. Също така при изпълнение на нареждането съответните портфейли трябва да разполагат с нужното количество левове (за купувача) и монети (за продавача). Използваните нареждания за покупка се премахват от списъка с нареждания, и за всяко едно от тях се създава трансакция във файла за трансакции *transactions.dat* със съответните изпращач, получател и монети.
- За всяка изпълнена трансакция трябва да отразите и промяната на **fiatMoney** в портфейлите на участниците в трансакцията като използвате гореспоменатия курс за монети спрямо **fiatMonet**.

Условията за създаване на нареждане за *покупка* на монети са следните:

- Съществува портфейл за подадения номер
- Наредителят трябва да има достатъчно истински пари за да закупи желания брой монети, изчислен при гореспоменатия курс. Трябва да се проверят и наличните нареждания дали няма нареждания от същият портфейл тъй като те са вече “одобрени” и трябва да може да се извършат.
- За да изпълни текущата заявка, програмата трябва да провери дали има нареждания за продажба на монети и да удовлетвори някои или всички от тях в реда в който са били направени. Също така при изпълнение на нареждането съответните портфейли трябва да разполагат с нужното количество левове (за купувача) и монети (за продавача). Изпълняват се заявки за продажба на монети докато се изпълни цялата текуща заявка или докато няма повече подходящи заявки. Ако се окаже че няма достатъчно заявки за да може текущата да се удовлетвори напълно, то тогава записваме текущата заявка във файла за заявки *orders.dat*, но със съответно приспаднато количество монети. За всяка от извършените трансакции в този процес трябва да направите запис за трансакция със съответните получател, изпращач и монети. Също всяка изпълнена заявка се премахва от списъка заявки.
- За всяка изпълнена трансакция трябва да отразите и промяната на **fiatMoney** в портфейлите на участниците в трансакцията, като използвате гореспоменатия курс за монети спрямо **fiatMonet**.

При изпълнение на нареждане трябва да генерирате текстов файл с информация за всички изпълнени заявки, които са били предизвикани от създаването на това нареждане. В този текстов файл трябва да има по един ред за всяка трансакция, в който се включват имената на получателя, изпращача и количеството трансферирани монети. Накрая на файла трябва да се запише броя извършени трансакции и себестойността на трансакциите в истински пари. Името на този файл трябва да съдържа идентификационния номер на портфейла, който задава нареждането и времето в което се е случило нареждането.

За да стимулираме използването на валутата за търговски и благотворителни цели трябва да може да се извършват и директни трансакции между два портфейла. За целта трябва да се поддържа команда `transfer **senderId** **receiverId** **fmiCoins**`. За успешното извършване на трансакция по тази команда трябва да съществуват портфейлите на изпращача и получателя, а също изпращачът трябва да разполага със съответното количество монети.

Програмата ви трябва да поддържа също и следните три команди:

- `wallet-info **walletId**` - дава информация за портфейл с въведен номер, като извежда името, количеството пари в него, а също и количеството монети които притежава. Количеството монети които притежава този портфейл може да бъде изчислено от трансакциите в които участва портфейла.
- `attract-investors` - дава информация за 10-те най богати (спрямо количеството монети) потребителя в системата към момента. За всеки от тях трябва да показва количеството монети, броят извършени заявки и времето на първата и последната извършена заявка.
- `quit` спиране на програмата с гарантирано записване на всички данни в съответните файлове.

Забележки и пояснения

- При пускане на програмата бинарните файлове споменати по-горе може да съществуват от предишно изпълнение. При това положение, при изпълнението на всяка команда трябва да се съобразявате с данните, вече записани във файловете.
- Няма ограничение кои данни може да съхранявате в паметта на вашата програма или да държите само във файловете, но имайте предвид, че този проект ще бъде много успешен. Това означава че кодът който напишете сега, ще се ползва от многото ви колеги след успешният старт на проекта и съответно той трябва да спазва добрите практики в ООП и да бъде лесно преизползваем. Както **всеки** програмист, ценящ труда си, трябва да се погрижите, че всяка команда ще се изпълнява максимално бързо и ефикасно въпреки големият брой портфейли и трансакции.

Отделът за развойна дейност прилага и следният откъс код, който да ви даде идея как ще са организирани данните и цели да ви помогне в започване на задачата:

```
struct Wallet {
    char owner[256];
    unsigned id;
    double fiatMoney;
};

struct Transaction {
    long long time;
    unsigned senderId;
    unsigned receiverId;
    double fmiCoins;
};

struct Order {
    enum Type { SELL, BUY } type;
    unsigned walletId;
    double fmiCoins;
};
```