Bank

Wygenerowano przez Doxygen 1.8.17

Rozdział 1

Indeks klas

1.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

Rozdział 2

Indeks plików

2.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

Rozdział 3

Dokumentacja klas

3.1 Dokumentacja struktury Account

`#include <Account.h>`

Diagram współpracy dla Account:

Metody publiczne

- Account (long long accountNumber, string name, string surname, double balance, double debit, TransactionList *&transactions)
- ~Account ()
- void accountInfo ()
- void withdrawal (double amount)
- void deposit (double amount)
- void transfer (Account *receiver, double amount)
- void accountStatement (string startDate, string stopDate)
- string getCurrentDate ()
- string formatBalance ()
- double roundAmount (double amount)
- bool checkDatesOrder (string youngDate, string oldDate)
- bool checkDateInRange (string checkedDate, string startDate, string stopDate)

Atrybuty publiczne

- long long accountNumber = 0

  numer konta użytkownika
- string name = ""

  imię użytkownika
- string surname = ""

  nazwisko użytkownika
- double balance = 0

  stan konta użytkownika
- double debit = 1000

  debet posiadany przez użytkownika
- TransactionList * transactions

  wskaźnik na listę transakcji
- Account * nextA

  wskaźnik na następny element listy kont użytkowników

3.1.1 Opis szczegółowy

Struktura konta użytkownika.

3.1.2 Dokumentacja konstruktora i destruktora

3.1.2.1 Account()

```
Account::Account (
            long long accountNumber,
            string name,
            string surname,
            double balance,
            double debit,
            TransactionList *& transactions )
```

Konstruktor.

Parametry

accountNumber	numer konta użytkownika
name	imię użytkownika
surname	nazwisko użytkownika
balance	stan konta użytkownika
debit	debet posiadany przez użytkownika
transactions	wskaźnik na listę transakcji

3.1.2.2 ~Account()

```
Account::~Account ( )
```

Destruktor, usuwa konto wywołany poleceniem delete.

3.1.3 Dokumentacja funkcji składowych

3.1.3.1 accountInfo()

```
void Account::accountInfo ( )
```

Funkcja wyświetla informacje o koncie wraz z wyciągiem transakcji.

3.1.3.2 accountStatement()

```
void Account::accountStatement (
            string startDate,
            string stopDate )
```

Funkcja wyświetla wyciąg z podsumowaniem danego konta posortowany według daty z podanego przedziału.

Parametry

startDate	data otwierająca zakres
stopDate	data zamykająca zakres

3.1.3.3 checkDateInRange()

```
bool Account::checkDateInRange (
            string checkedDate,
            string startDate,
            string stopDate )
```

Funkcja sprawdza czy podana data mieści się w zakresie dwóch podanych dat.

Parametry

checkedDate	sprawdzana data
startDate	data otwierająca zakres
stopDate	data zamykająca zakres

Zwraca

prawda gdy data mieści się w podanym zakresie w przeciwnym wypadku fałsz

3.1.3.4 checkDatesOrder()

```
bool Account::checkDatesOrder (
            string youngDate,
            string oldDate )
```

Funkcja sprawdza czy podane daty są w odpowiedniej kolejności młodsza-starsza.

Parametry

youngDate	pierwsza data do sprawdzenia, która powinna być młodsza
oldDate	druga data do sprawdzenia, która powinna być starsza

Zwraca

prawda gdy daty są w odpowiedniej kolejności w przeciwnym wypadku fałsz

3.1.3.5 deposit()

```
void Account::deposit (
            double amount )
```

Funkcja wpłaca kwotę na konto danego użytkownika.

Parametry

amount	kwota do wpłaty

3.1.3.6 formatBalance()

```
string Account::formatBalance ( )
```

Funkcja zwraca liczbę w odpowiednim formacie to znaczy: ze znakiem "+" lub "-" i dokładnością do dwóch miejsc po przecinku.

Parametry

amount	liczba do sformatowania

Zwraca

odpowiednio sformatowana liczba jako ciąg znaków

3.1.3.7 getCurrentDate()

```
string Account::getCurrentDate ( )
```

Funkcja pobiera z systemu aktualną datę w sposób polecany przez stronę Micorsoft.

Zwraca

aktualna data wraz z godziną w formacie ISO 8601 (YYYY-MM-DDThh:mm)

3.1.3.8 roundAmount()

```
double Account::roundAmount (
            double amount )
```

Funkcja zaokrągla podaną liczbę w dół do dwóch miejsc po przecinku.

Parametry

amount	liczba do zaokrąglenia w formacie double

Zwraca

zaokrąglona liczba

3.1.3.9 transfer()

```
void Account::transfer (
            Account * receiver,
            double amount )
```

Funkcja przelewa kwotę na konto odbiorcy, o ile pozwalają na to środki danego użytkownika.

Parametry

receiver	wskaźnik na konto odbiorcy
amount	kwota przelewu

3.1.3.10 withdrawal()

```
void Account::withdrawal (
            double amount )
```

Funkcja wypłaca kwotę z konta o ile pozwalają na to środki posiadane przez danego użytkownika.

Parametry

amount	kwota do wypłaty

Dokumentacja dla tej struktury została wygenerowana z plików:

- Account.h
- Account.cpp

3.2 Dokumentacja struktury AccountList

```
#include <AccountList.h>
```

Diagram współpracy dla AccountList:

Metody publiczne

- AccountList ()
- ∼AccountList ()
- void addAccount (Account ∗account)
- Account ∗ findAccount (long long accountNumber)
- void showAccounts ()
- void readData (string fileName)
- void saveData ()

Atrybuty publiczne

- Account ∗ aHead

  wskaźnik na głowę listy kont użytkowników
- string fileName = ""

  nazwa pliku do odczytu
- string separator = "##"

  separator

3.2.1 Opis szczegółowy

Struktura listy kont użytkowników.

3.2.2 Dokumentacja konstruktora i destruktora

3.2.2.1 AccountList()

```
AccountList::AccountList ( )
```

Konstruktor,

Zwraca

3.2.2.2 ∼AccountList()

```
AccountList::~AccountList ( )
```

Destruktor, usuwa listę kont wywołany poleceniem delete.

Zwraca

usunięta lista kont

3.2.3 Dokumentacja funkcji składowych

3.2.3.1 addAccount()

```
void AccountList::addAccount (
            Account * account )
```

Funkcja dodaje nowe konto do listy kont.

Parametry

account	wskaźnik na konto

3.2.3.2 findAccount()

```
Account * AccountList::findAccount (
            long long accountNumber )
```

Funkcja wyszukuje konto w liście na podstawie jego numeru.

Parametry

accountNumber	numer szukanego konta

Zwraca

wskaźnik na znalezione konto o ile istnieje w przeciwnym przypadku NULL

3.2.3.3 readData()

```
void AccountList::readData (
            string fileName )
```

Funkcja odczytuje dane z pliku i na ich podstawie tworzy listę kont.

Parametry

fileName	nazwa odczytywanego pliku

3.2.3.4 saveData()

```
void AccountList::saveData ( )
```

Funkcja nadpisuje w wcześniej odczytanym pliku dane z listy kont.

3.2.3.5 showAccounts()

```
void AccountList::showAccounts ( )
```

Funkcja wyświetla informacje o kontach z listy.

Dokumentacja dla tej struktury została wygenerowana z plików:

- AccountList.h
- AccountList.cpp

3.3 Dokumentacja struktury Raport

`#include <Raport.h>`

Diagram współpracy dla Raport:

Metody publiczne

- Raport (long long accountNumber, string date, double amount)
- Raport (long long accountNumber, string surname)

Atrybuty publiczne

- long long accountNumber = 0

  numer konta użytkownika
- string surname = ""

  nazwisko użytkownika
- string date

  data transakcji
- double amount = 0

  kwota transakcji
- Raport * nextR

  wskaźnik na następny element listy raportu

3.3.1 Opis szczegółowy

Struktura elementu listy raportu.

3.3.2 Dokumentacja konstruktora i destruktora

3.3.2.1 Raport() [1/2]

```
Raport::Raport (
            long long accountNumber,
            string date,
            double amount )
```

Konstruktor.

Parametry

accountNumber	numer konta użytkownika
date	data transakcji
amount	kwota transakcji

3.3.2.2 Raport() [2/2]

```
Raport::Raport (
            long long accountNumber,
            string surname )
```

Konstruktor.

Parametry

accountNumber	numer konta użytkownika
surname	nazwisko użytkownika

Dokumentacja dla tej struktury została wygenerowana z plików:

- Raport.h
- Raport.cpp

3.4 Dokumentacja struktury RaportList

`#include <RaportList.h>`

Diagram współpracy dla RaportList:

Metody publiczne

- RaportList ()
- ~RaportList ()
- void addItem (Raport *item)
- void showRaportTransaction ()
- void showRaportDebit ()
- void raportTransactions (AccountList *list, string startDate, string stopDate, int sortType)
- void raportDebitUsers (AccountList *list, int sortType)
- void sortRaportTransaction (int sortType)
- void sortRaportDebit (int sortType)
- void sortByDate ()
- void sortByAmount ()
- void sortByAccount ()
- void sortBySurname ()
- void swap (Raport *a, Raport *b)
- string formatAmount (double amount)
- string strinfToLower (string word)
- bool checkDatesOrder (string youngDate, string oldDate)

Atrybuty publiczne

- Raport * rHead

  wskaźnik na głowę listy raportu

3.4.1 Opis szczegółowy

Struktura listy raportu.

3.4.2 Dokumentacja konstruktora i destruktora

3.4.2.1 RaportList()

```
RaportList::RaportList ( )
```

Konstruktor.

3.4.2.2 ~RaportList()

```
RaportList::~RaportList ( )
```

Destruktor, usuwa listę raportu wywołany poleceniem delete.

3.4.3 Dokumentacja funkcji składowych

3.4.3.1 addItem()

```
void RaportList::addItem (
            Raport * item )
```

Funkcja dodaje nowy element do listy raportu.

Parametry

date	data transakcji
amount	kwota transakcji

3.4.3.2 checkDatesOrder()

```
bool RaportList::checkDatesOrder (
            string youngDate,
            string oldDate )
```

Funkcja sprawdza czy podana data mieści się w zakresie dwóch podanych dat.

Parametry

checkedDate	sprawdzana data
startDate	data otwierająca zakres
stopDate	data zamykająca zakres

Zwraca

prawda gdy data znajduje się w podanym zakresie w przeciwnym wypadku fałsz

3.4.3.3 formatAmount()

```
string RaportList::formatAmount (
            double amount )
```

Funkcja zwraca liczbę w odpowiednim formacie to znaczy: ze znakiem "+" lub "-" i dokładnością do dwóch miejsc po przecinku.

Parametry

amount	liczba do sformatowania

Zwraca

odpowiednio sformatowana liczba jako ciąg znaków

3.4.3.4 raportDebitUsers()

```
void RaportList::raportDebitUsers (
            AccountList * list,
            int sortType )
```

Funkcja do obsługi raportu użytkowników pyta o odnośnik do sortowania (nazwisko, numer konta).

Parametry

list	wskaźnik na listę kont
sortType	odnośnik do sortowania [(1)nazwisko, (2)numer konta]

3.4.3.5 raportTransactions()

```
void RaportList::raportTransactions (
            AccountList * list,
            string startDate,
            string stopDate,
            int sortType )
```

Funkcja do obsługi raportu transakcji.

Parametry

list	wskaźnik na listę kont
startDate	data otwierająca zakres
stopDate	data zamykająca zakres
sortType	odnośnik do sortowania [(1)data, (2)kwota transakcji, (3)numer konta]

3.4.3.6 showRaportDebit()

```
void RaportList::showRaportDebit ( )
```

Funkcja wyświetla wszystkie elementy listy raportu w formie numer konta i nazwisko użytkownika.

3.4.3.7 showRaportTransaction()

```
void RaportList::showRaportTransaction ( )
```

Funkcja wyświetla wszystkie elementy listy raportu w formie numer konta, data i kwota transakcji.

3.4.3.8 sortByAccount()

```
void RaportList::sortByAccount ( )
```

Funkcja sortuje listę raportu rosnąco według numerów kont metodą bąbelkową.

3.4.3.9 sortByAmount()

```
void RaportList::sortByAmount ( )
```

Funkcja sortuje listę raportu rosnąco według kwot transakcji metodą bąbelkową.

3.4.3.10 sortByDate()

```
void RaportList::sortByDate ( )
```

Funkcja sortuje listę raportu rosnąco według daty metodą bąbelkową.

3.4.3.11 sortBySurname()

```
void RaportList::sortBySurname ( )
```

Funkcja sortuje listę raportu rosnąco według nazwisk metodą bąbelkową.

3.4.3.12 sortRaportDebit()

```
void RaportList::sortRaportDebit (
            int sortType )
```

Funkcja do obsługi sortowania raportu użytkowników.

Parametry

sortType	numer odnośnika do sortowania

3.4.3.13 sortRaportTransaction()

```
void RaportList::sortRaportTransaction (
            int sortType )
```

Funkcja do obsługi sortowania raportu transakcji.

Parametry

sortType	numer odnośnika do sortowania

3.4.3.14 strinfToLower()

```
string RaportList::strinfToLower (
            string word )
```

Funkcja zamienia wszystkie duże litery podanego słowa na małe.

Parametry

word	słowo do sformatowania

Zwraca

odpowienio sformatowan słowo jako ciąg znaków

3.4.3.15 swap()

```
void RaportList::swap (
            Raport * a,
            Raport * b )
```

Funkcja zamienia kolejnością dwie transakcję.

Parametry

a	wskaźnik na pierwszą transakcję do zamiany
b	wskaźnik na drugą transakcję do zamiany

Dokumentacja dla tej struktury została wygenerowana z plików:

- RaportList.h
- RaportList.cpp

3.5 Dokumentacja struktury Transaction

`#include <Transaction.h>`

Diagram współpracy dla Transaction:

Metody publiczne

- Transaction ()
- void showTrans ()
- void showSignedTrans ()
- string formatAmount ()

Atrybuty publiczne

- string date = ""
  data transakcji
- double amount = 0
  kwota transakcji
- Transaction * nextT
  wskaźnik na następny element listy transakcji

3.5.1 Opis szczegółowy

Struktura transakcji.

3.5.2 Dokumentacja konstruktora i destruktora

3.5.2.1 Transaction()

```
Transaction::Transaction ( )
```

Konstruktor.

3.5.3 Dokumentacja funkcji składowych

3.5.3.1 formatAmount()

```
string Transaction::formatAmount ( )
```

Funkcja zwraca liczbę w odpowiednim formacie to znaczy: ze znakiem "+" lub "-" i dokładnością do dwóch miejsc po przecinku.

Parametry

amount	liczba do sformatowania

Zwraca

odpowiednio sformatowana liczba jako ciąg znaków

3.5.3.2 showSignedTrans()

```
void Transaction::showSignedTrans ( )
```

Funkcja wyświetla podpisaną datę i kwotę transakcji.

3.5.3.3 showTrans()

```
void Transaction::showTrans ( )
```

Funkcja wyświetla datę i kwotę transakcji.

Dokumentacja dla tej struktury została wygenerowana z plików:

- Transaction.h
- Transaction.cpp

3.6 Dokumentacja struktury TransactionList

`#include <TransactionList.h>`

Diagram współpracy dla TransactionList:

Metody publiczne

- TransactionList ()
- ∼TransactionList ()
- void addTransaction (string date, double amount)
- Transaction ∗ findTransaction (string date, double amount)
- void deleteTransaction (Transaction ∗transaction)
- void showTransactions ()
- void dateSort ()
- void swap (Transaction ∗a, Transaction ∗b)
- bool checkDatesOrder (string youngDate, string oldDate)

Atrybuty publiczne

- Transaction ∗ tHead
  wskaźnik na głowę listy transakcji

3.6.1 Opis szczegółowy

Struktura listy transakcji.

3.6.2 Dokumentacja konstruktora i destruktora

3.6.2.1 TransactionList()

```
TransactionList::TransactionList ( )
```

Konstruktor.

3.6.2.2 ∼TransactionList()

```
TransactionList::~TransactionList ( )
```

Destruktor, usuwa liste transakcji wywołany poleceniem delete.

3.6.3 Dokumentacja funkcji składowych

3.6.3.1 addTransaction()

```
void TransactionList::addTransaction (
            string date,
            double amount )
```

Funkcja dodaje nową transakcję do listy transakcji.

Parametry

date	data transakcji
amount	kwota transakcji

3.6.3.2 checkDatesOrder()

```
bool TransactionList::checkDatesOrder (
            string youngDate,
            string oldDate )
```

Funkcja sprawdza czy podane daty są w odpowiedniej kolejności młodsza-starsza.

Parametry

youngDate	pierwsza data do sprawdzenia, która powinna być młodsza
oldDate	druga data do sprawdzenia, która powinna być starsza

Zwraca

prawda lub fałsz w zależności od tego czy daty są w odpowiedniej kolejności

3.6.3.3 dateSort()

```
void TransactionList::dateSort ( )
```

Funkcja sortuje listę transakcji rosnąco według daty metodą bąbelkową

3.6.3.4 deleteTransaction()

```
void TransactionList::deleteTransaction (
            Transaction * transaction )
```

Funkcja usuwa transakcję z listy transakcji.

Parametry

transaction	wskaźnik na transakcję przeznaczoną do usunięcia

3.6.3.5 findTransaction()

```
Transaction * TransactionList::findTransaction (
            string date,
            double amount )
```

Funkcja wyszukuje transakcję na podstawie daty i kwoty.

Parametry

date	data transakcji
amount	kwota transakcji

Zwraca

wskaźnik na znalezioną transakcję o ile istnieje w przeciwnym przypadku NULL

3.6.3.6 showTransactions()

```
void TransactionList::showTransactions ( )
```

Funkcja wyświetla wszystkie transakcje z listy transakcji.

3.6.3.7 swap()

```
void TransactionList::swap (
            Transaction * a,
            Transaction * b )
```

Funkcja zamienia kolejnością dwie transakcję

Parametry

a	wskaźnik na pierwszą transakcję do zamiany
b	wskaźnik na drugą transakcję do zamiany

Dokumentacja dla tej struktury została wygenerowana z plików:

- TransactionList.h
- TransactionList.cpp

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku Account.cpp

```
#include "Account.h"
```
Wykres zależności załączania dla Account.cpp:

4.2 Dokumentacja pliku Account.h

```
#include <iostream>
#include <string>
#include <time.h>
#include "TransactionList.h"
#include "Transaction.h"
```
Wykres zależności załączania dla Account.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- struct Account

4.3 Dokumentacja pliku AccountList.cpp

```
#include "AccountList.h"
```
Wykres zależności załączania dla AccountList.cpp:

4.4 Dokumentacja pliku AccountList.h

```
#include <iostream>
#include <fstream>
#include <string>
#include "Account.h"
#include "TransactionList.h"
```
Wykres zależności załączania dla AccountList.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- struct AccountList

4.5 Dokumentacja pliku Bank.cpp

```
#include <iostream>
#include <fstream>
#include <string>
#include "AccountList.h"
#include "RaportList.h"
```

Wykres zależności załączania dla Bank.cpp:

Funkcje

- int **main** (int argc, char ∗argv[ ])

4.6 Dokumentacja pliku Raport.h

```
#include <string>
```

Wykres zależności załączania dla Raport.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- struct Raport

4.7 Dokumentacja pliku RaportList.cpp

```
#include "RaportList.h"
```

Wykres zależności załączania dla RaportList.cpp:

4.8 Dokumentacja pliku RaportList.h

```
#include <iostream>
#include <string>
#include "Raport.h"
#include "AccountList.h"
#include "Account.h"
#include "Transaction.h"
```

Wykres zależności załączania dla RaportList.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- struct RaportList

4.9 Dokumentacja pliku Transaction.cpp

```
#include "Transaction.h"
```
Wykres zależności załączania dla Transaction.cpp:

4.10 Dokumentacja pliku Transaction.h

```
#include <iostream>
#include <string>
```
Wykres zależności załączania dla Transaction.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- struct Transaction

4.11 Dokumentacja pliku TransactionList.cpp

```
#include "TransactionList.h"
```
Wykres zależności załączania dla TransactionList.cpp:

4.12 Dokumentacja pliku TransactionList.h

```
#include <iostream>
#include <string>
#include "Transaction.h"
```
Wykres zależności załączania dla TransactionList.h: Ten wykres pokazuje, które pliki bezpośrednio lub pośrednio załączają ten plik:

Komponenty

- struct TransactionList