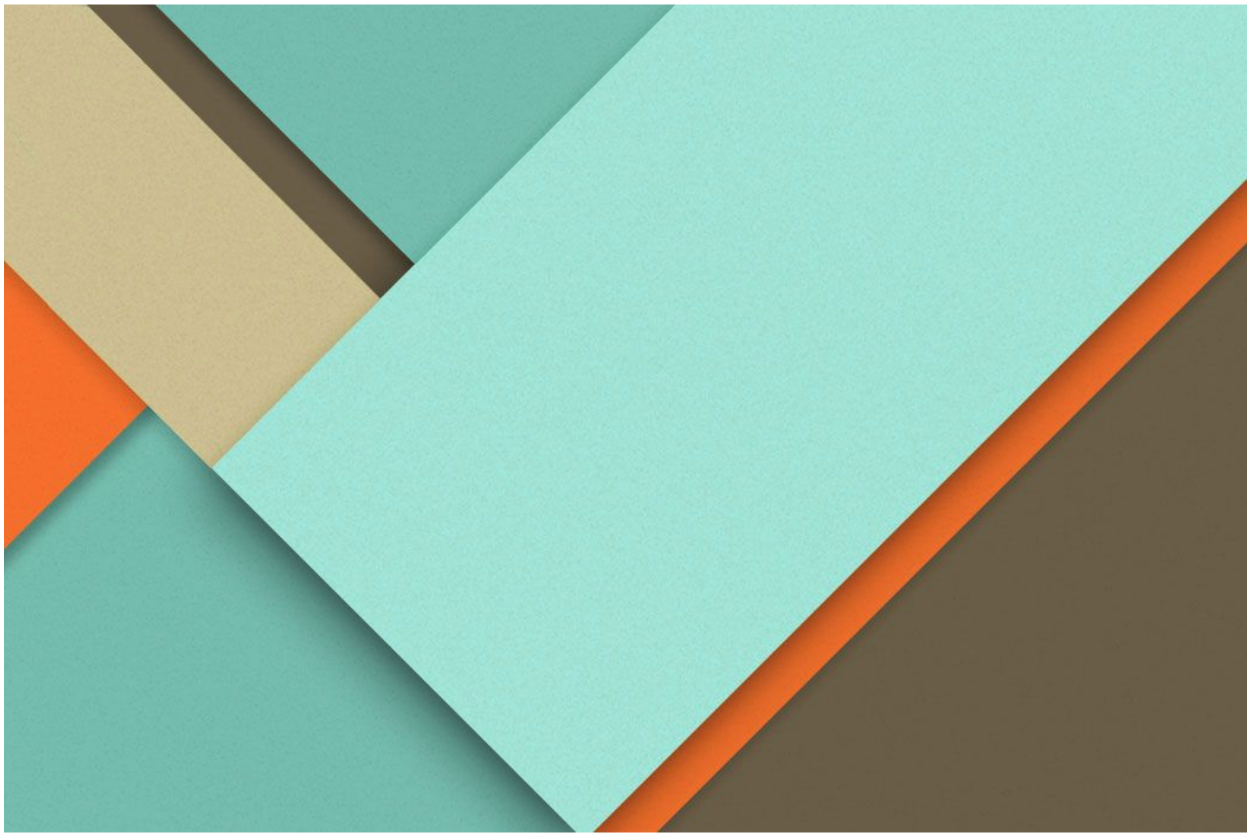

Προγραμματισμός Συστήματος Εργασία 1

Ονοματεπώνυμο: Μπαρμπάρ Ηλίας-Ελιάς
ΑΜ: 1115201200118

Data Structures Used

Wallets: Simple, dynamic, linked list.

SenderHashTable: Hash table with buckets with records assigned dynamically and each record connects to a dynamic, linked list for each record's transactions.

ReceiverHashTable: Same as the senderHashTable.

BitCoinTree: A linked list where each node is also the head of a tree, those trees use a linked list that holds all the leaves since most functions use the leaves as opposed to the whole tree .

Files

main.cpp:

Our main reads the necessary info from cmd, then proceeds to read the input file and after that the application starts.

reads:

Reads input files and fills our data structures also reads from a file when requested by '/requestTransactions inputfile'.

application:

This function interacts with the data structures and the user to implement all following functions:

/requestTransaction… (fills a transaction and if possible informs the necessary data structures)

/requestTransactions… (same as the above)

/findEarnings… (prints all transactions received by a user)

/findPayments… (prints all transactions sent by a user)

/walletStatus… (prints the wallet status of a user)

/bitCoinStatus… (prints the amount of transactions the bitcoin was involved in as well as the current unspent amount)

/traceCoin… (prints all transactions in which the bitcoin was involved)

/exit… (exits the program freeing all structures from memory)

wallets:

Consists of the wallet linked list and the bitcoin linked list every wallet has.

class walletList functions:

void add: Given a cstring containing the userID, creates and initializes and adds a new wallet/user on the linked list.

void addBitcoin: Given a cstring containing the bitcoinID, creates, initializes and adds a new bitcoin on the last user. This function only works with the bitCoinBalancesFile.

wallet *findUser: Given a cstring containing a userID, it finds the user with such ID an returns an address to that wallet. If not it returns NULL.

void print: a simple print function for checks (not used by the program).

struct wallet functions:

int checkBalance: a bool function that returnes whether the balance of a user is enough to cover a given amount

int spendBitCoins/receiveBitCoins: interacts with senderHashTable to remove the necessary amount from a user and give it to the user receiving the amount (this only happens on the sender hash table, so the receiver hash table doesn't have to do anything).

bitCoinTree:

add/traAdd: Initializes a bitcoin tree and adds a node to the dynamic list of treeheads. With traAdd we make sure to split the tree nodes every time a transaction takes place.

printStatus: Uses the leaves for the '/bitCoinStatus' function.

printCoin: Parses a bitcoinTree and prints that transactions that took place.

receiverHashtable/senderHashtable:

addRecord: Using the input files adds records to our hash tables

printEarnings/printPayments: through the receiver/sender hash table finds a user and then uses their dynamic transaction list to print all the transactions they took a part in.

transaction/functions/bucketNode:

Complementary files that have simple functions and structs used throughout the program.

Compilation

We compile using a makefile so all the user has to do is type ‘make’ which has the following form:

```
bitcoin:
	g++ main.cpp application.cpp receiverHashtable.cpp bitCoinTree.cpp wallets.cpp senderHashtable.cpp transaction.cpp reads.cpp bucketNode.cpp functions.cpp -o bitcoin

.PHONY: bitcoin
```

Resources

https://www.dreamincode.net/forums/topic/10157-data-structures-in-c-tutorial/

https://stackoverflow.com/questions/11168519/fscanf-or-fgets-reading-a-file-line-after-line

https://stackoverflow.com/questions/3501338/c-read-file-line-by-line

https://stackoverflow.com/questions/1442116/how-to-get-the-date-and-time-values-in-a-c-program

https://stackoverflow.com/questions/11765301/how-do-i-get-the-unix-timestamp-in-c-as-an-int

https://stackoverflow.com/questions/6417158/c-how-to-free-nodes-in-the-linked-list