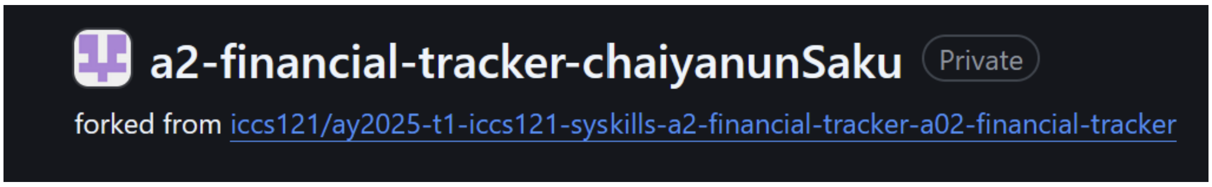

a2-financial-tracker-chaiyanunSaku Private

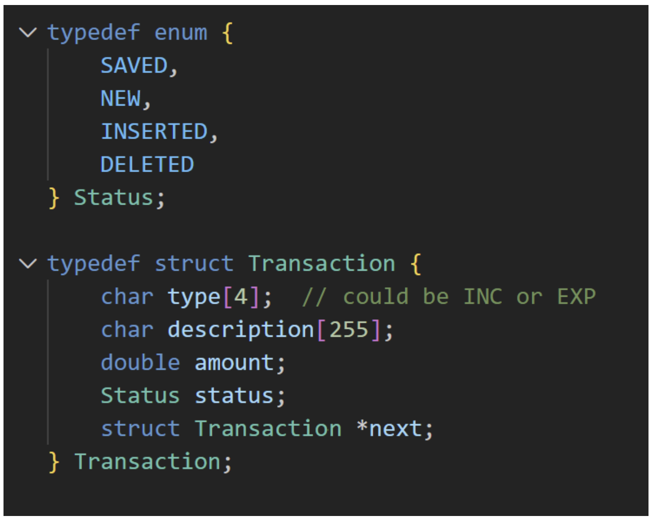
forked from iccs121/ay2025-t1-iccs121-syskills-a2-financial-tracker-a02-financial-tracker

ICCS121 System Skills and Low-level Programming Assignment 2

https://github.com/iccs121/a2-financial-tracker-chaiyanunSaku

Chaiyanun Sakulsaowapakkul 6681299

Linked list implementation



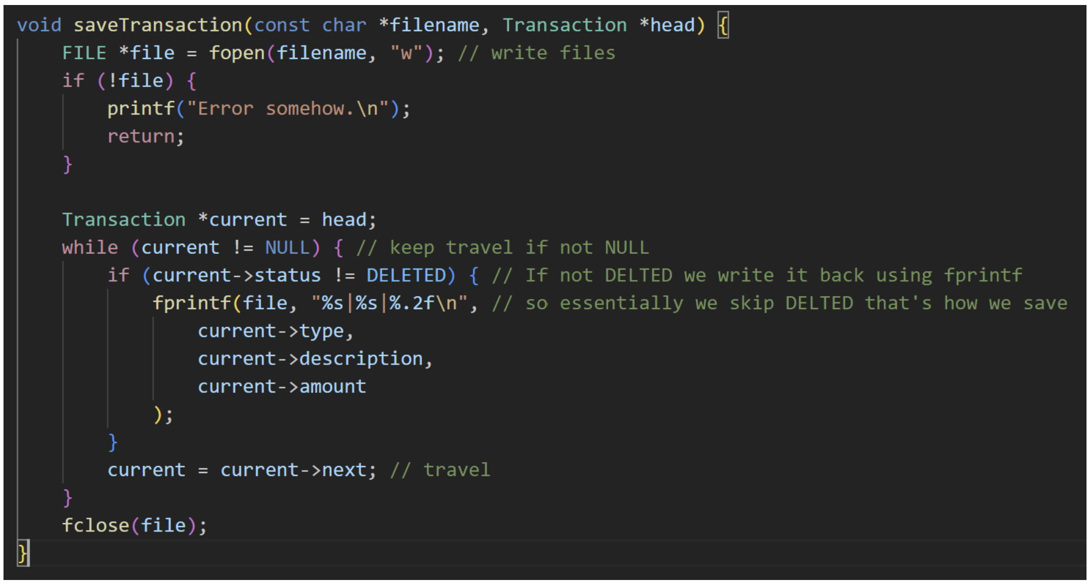
```c
typedef enum {
    SAVED,
    NEW,
    INSERTED,
    DELETED
} Status;

typedef struct Transaction {
    char type[4];  // could be INC or EXP
    char description[255];
    double amount;
    Status status;
    struct Transaction *next;
} Transaction;
```

We have a linked list called “Transaction” which has char type, char description, double amount, Status status, and a struct Transaction *next which is a pointer to the next node in the linked list.

enum Status makes it easier to track what kind of Transaction each node is.

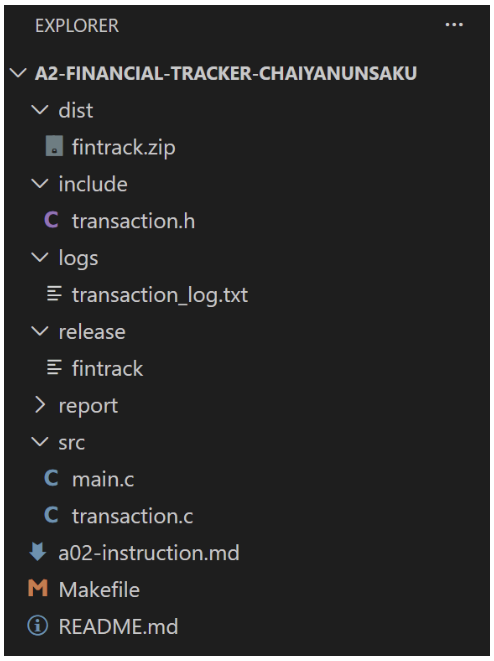
We use Linked list instead of something like an array because Linked list has dynamic size, we don’t know how many transactions the user will add. Linked list and grow and shrink using malloc without needing to reallocate large arrays which make inserting or deleting transactions in any location very easy. At most it is O(n) time.

Writing nodes back in the file is very easy because the “DELETED” node will just be skipped by the fprintf().   while (current->status != DELETED)

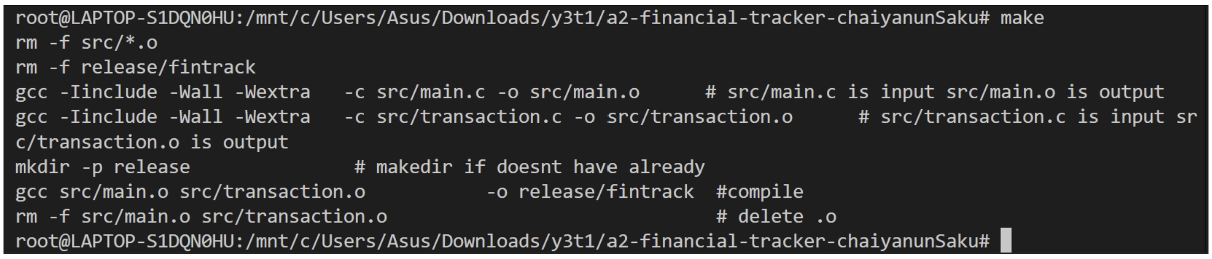


```c
void saveTransaction(const char *filename, Transaction *head) {
    FILE *file = fopen(filename, "w"); // write files
    if (!file) {
        printf("Error somehow.\n");
        return;
    }

    Transaction *current = head;
    while (current != NULL) { // keep travel if not NULL
        if (current->status != DELETED) { // If not DELTED we write it back using fprintf
            fprintf(file, "%s|%s|%.2f\n", // so essentially we skip DELTED that's how we save
                current->type,
                current->description,
                current->amount
            );
        }
        current = current->next; // travel
    }
    fclose(file);
}
```

Showcasing

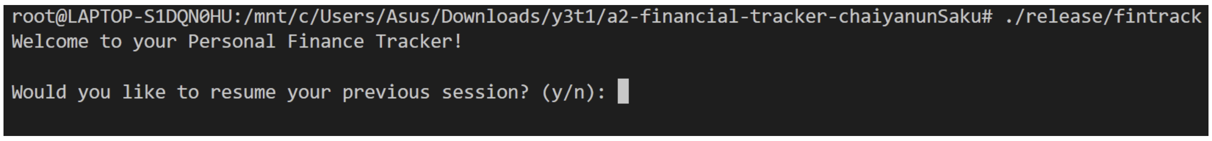


These are the directories and files for this project.
To begin, go ahead and type “make” in the terminal.



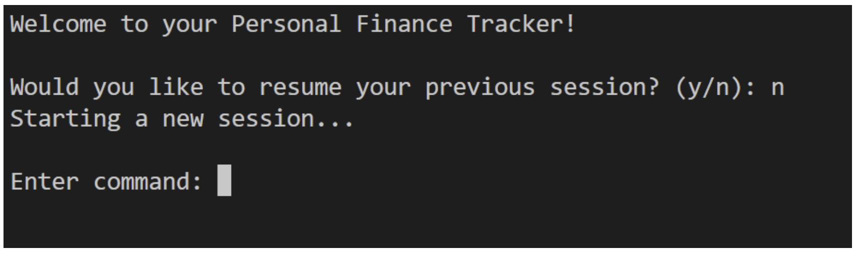
```
root@LAPTOP-S1DQN0HU:/mnt/c/Users/Asus/Downloads/y3t1/a2-financial-tracker-chaiyanunSaku# make
rm -f src/*.o
rm -f release/fintrack
gcc -Iinclude -Wall -Wextra   -c src/main.c -o src/main.o      # src/main.c is input src/main.o is output
gcc -Iinclude -Wall -Wextra   -c src/transaction.c -o src/transaction.o      # src/transaction.c is input sr
c/transaction.o is output
mkdir -p release               # makedir if doesnt have already
gcc src/main.o src/transaction.o           -o release/fintrack  #compile
rm -f src/main.o src/transaction.o                            # delete .o
root@LAPTOP-S1DQN0HU:/mnt/c/Users/Asus/Downloads/y3t1/a2-financial-tracker-chaiyanunSaku#
```

(all: clean $(EXECUTABLE) dist # make will do clean, compile, and zip)
This make command will clean (delete .o files), compile, and then zip into /dist/fintrack.zip.
The executable will be in /release/fintrack so to run the program we simply type “./release/fintrack”.



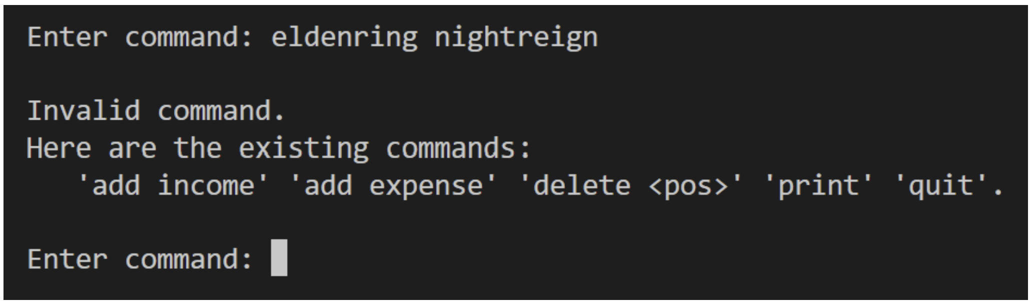
```
root@LAPTOP-S1DQN0HU:/mnt/c/Users/Asus/Downloads/y3t1/a2-financial-tracker-chaiyanunSaku# ./release/fintrack
Welcome to your Personal Finance Tracker!

Would you like to resume your previous session? (y/n):
```

We will see the welcome message and the program will ask if we want to resume from the previous sessions. Let’s begin with not resuming. Simply type “n”



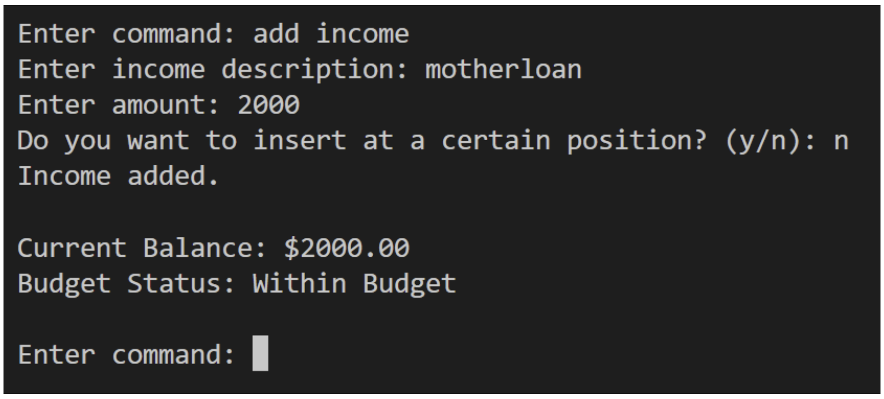
```
Welcome to your Personal Finance Tracker!

Would you like to resume your previous session? (y/n): n
Starting a new session...

Enter command: 
```

There are a total of 5 commands available. The program will keep asking for commands until we type “quit”. If we typed in an unknown command the program will tell us what are the existing commands to use.



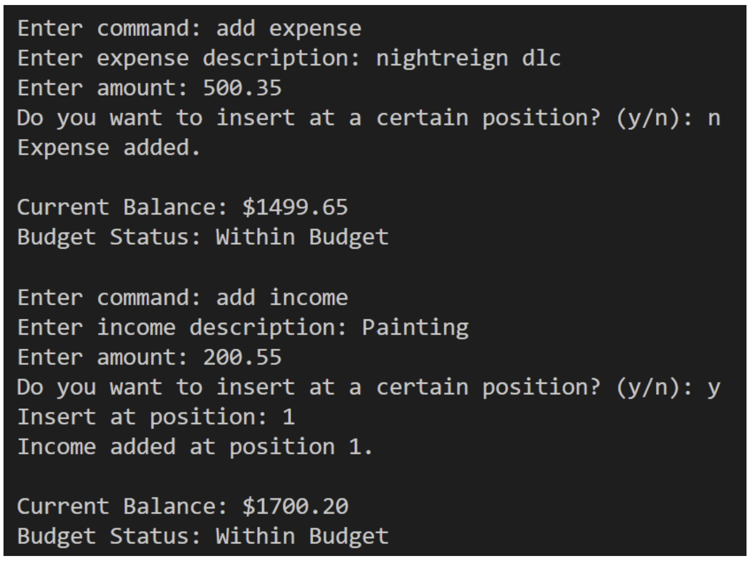
```
Enter command: eldenring nightreign

Invalid command.
Here are the existing commands:
   'add income' 'add expense' 'delete <pos>' 'print' 'quit'.

Enter command: 
```

Let’s try adding income without inserting



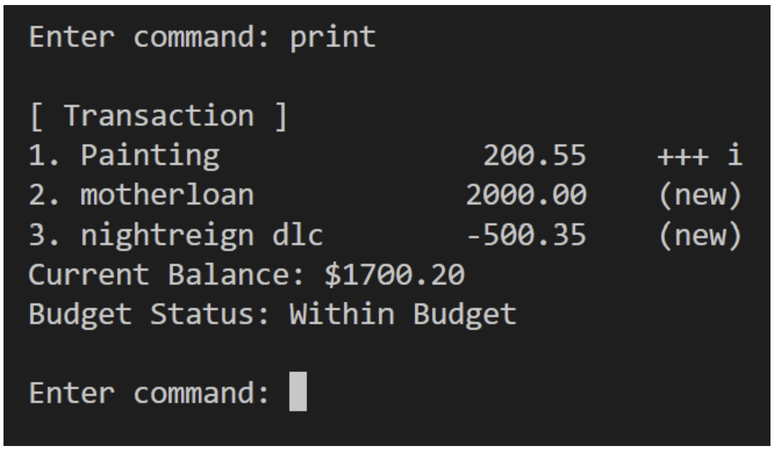
```
Enter command: add income
Enter income description: motherloan
Enter amount: 2000
Do you want to insert at a certain position? (y/n): n
Income added.

Current Balance: $2000.00
Budget Status: Within Budget

Enter command: 
```

Let’s add more transaction and this time let’s use insertion



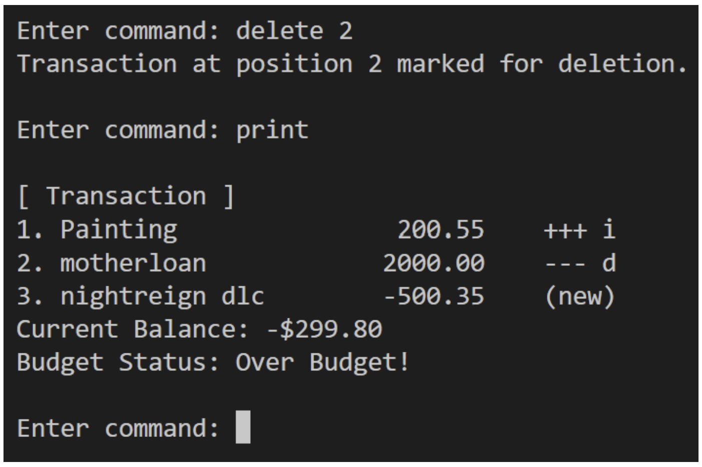
```
Enter command: add expense
Enter expense description: nightreign dlc
Enter amount: 500.35
Do you want to insert at a certain position? (y/n): n
Expense added.

Current Balance: $1499.65
Budget Status: Within Budget

Enter command: add income
Enter income description: Painting
Enter amount: 200.55
Do you want to insert at a certain position? (y/n): y
Insert at position: 1
Income added at position 1.

Current Balance: $1700.20
Budget Status: Within Budget
```

Let’s “print” and see our transaction



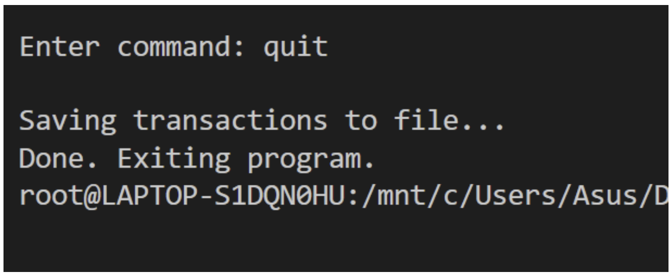
```
Enter command: print

[ Transaction ]
1. Painting              200.55    +++ i
2. motherloan           2000.00    (new)
3. nightreign dlc       -500.35    (new)
Current Balance: $1700.20
Budget Status: Within Budget

Enter command: 
```

Painting was inserted at the first node labelled as +++i. Now let’s try deleting a node at some position.

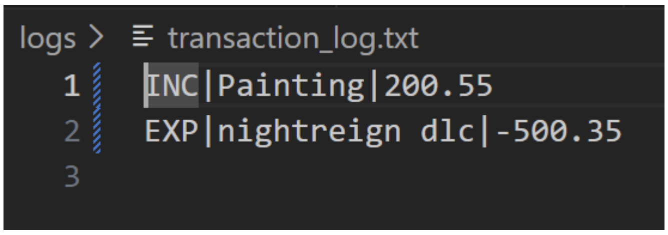


```
Enter command: delete 2
Transaction at position 2 marked for deletion.

Enter command: print

[ Transaction ]
1. Painting              200.55    +++ i
2. motherloan           2000.00    --- d
3. nightreign dlc       -500.35    (new)
Current Balance: -$299.80
Budget Status: Over Budget!

Enter command: 
```

Now we are over budget and node 2 motherloan has been tagged as - - - d which is DELETED but it is not deleted yet until we say "quit". Let's quit for now and check our txt file

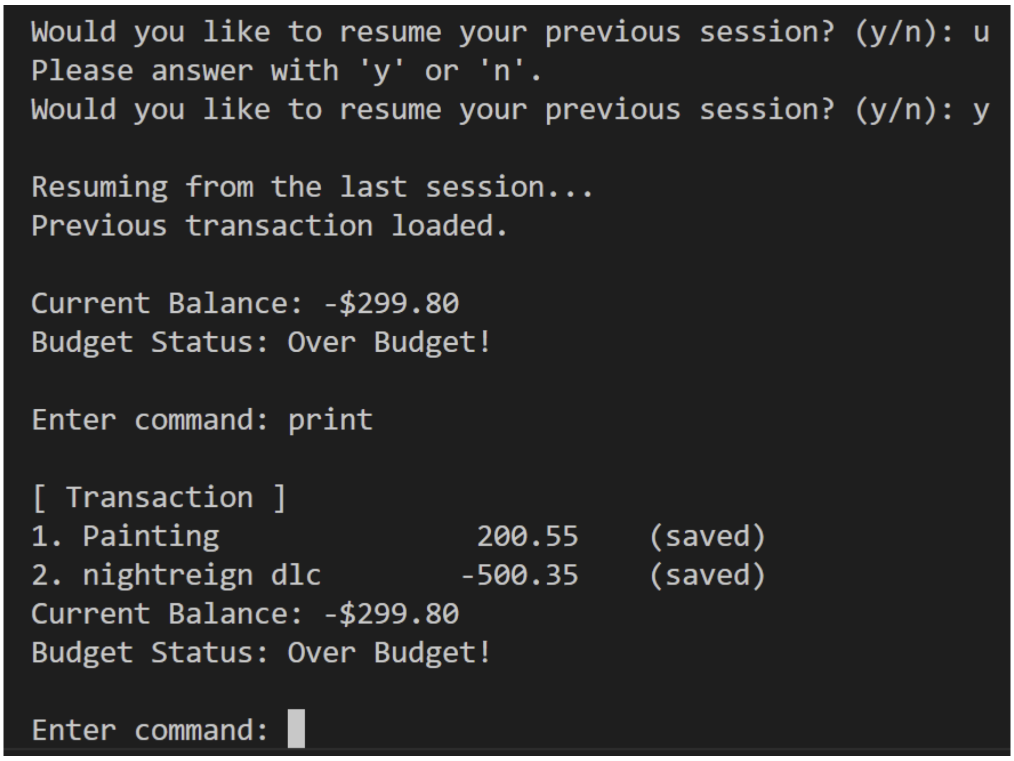


```
Enter command: quit

Saving transactions to file...
Done. Exiting program.
root@LAPTOP-S1DQN0HU:/mnt/c/Users/Asus/D
```



```
logs > transaction_log.txt
INC|Painting|200.55
EXP|nightreign dlc|-500.35

```

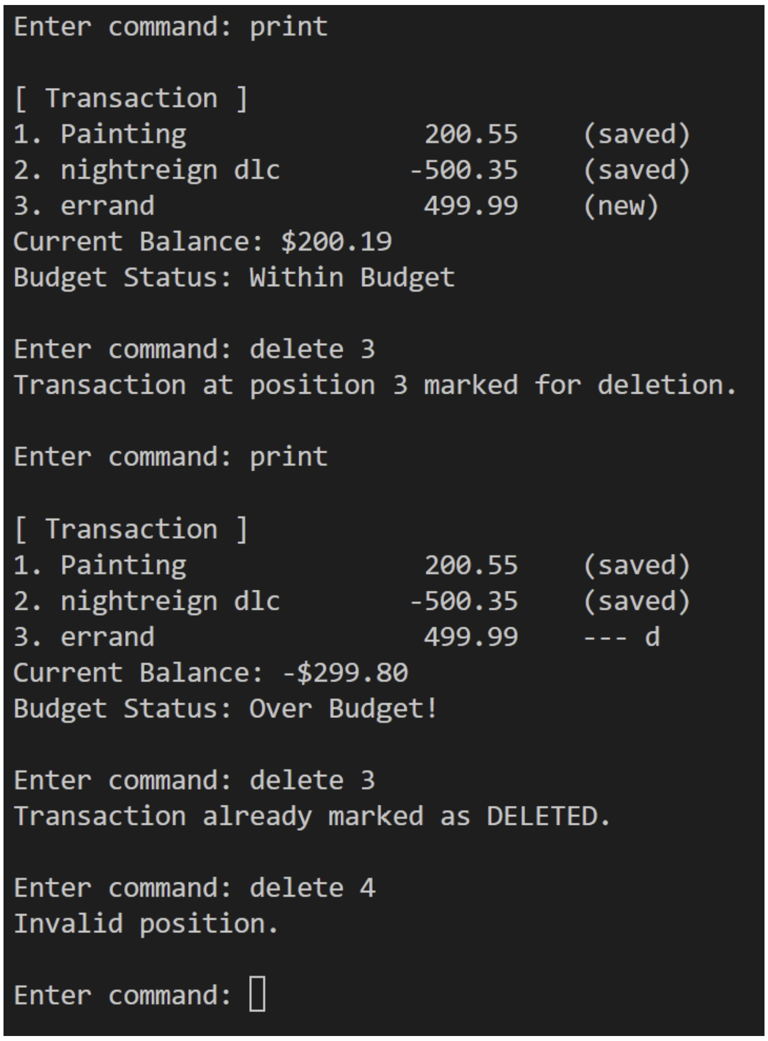
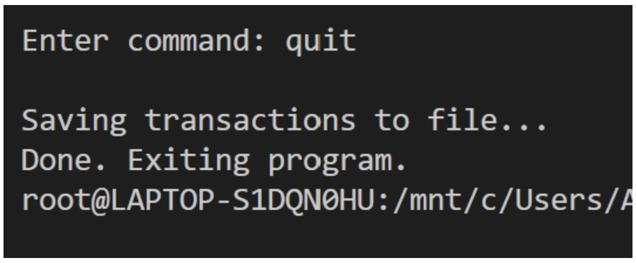
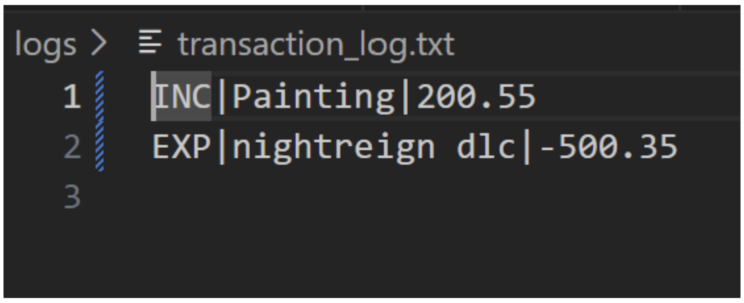
Our log is working as expected. Let's open the file again using ./release/fintrack and this time we will resume from the previous session by typing 'y'.



```
Would you like to resume your previous session? (y/n): u
Please answer with 'y' or 'n'.
Would you like to resume your previous session? (y/n): y

Resuming from the last session...
Previous transaction loaded.

Current Balance: -$299.80
Budget Status: Over Budget!

Enter command: print

[ Transaction ]
1. Painting                200.55    (saved)
2. nightreign dlc         -500.35    (saved)
Current Balance: -$299.80
Budget Status: Over Budget!

Enter command: 
```

We read the file gracefully even if we wrote 'u' we can answer the program again. Our transaction is tagged as (saved) because we loaded from the file.

Let's try deleting the same node multiple times


```
Enter command: print

[ Transaction ]
1. Painting                 200.55    (saved)
2. nightreign dlc          -500.35    (saved)
3. errand                   499.99    (new)
Current Balance: $200.19
Budget Status: Within Budget

Enter command: delete 3
Transaction at position 3 marked for deletion.

Enter command: print

[ Transaction ]
1. Painting                 200.55    (saved)
2. nightreign dlc          -500.35    (saved)
3. errand                   499.99    --- d
Current Balance: -$299.80
Budget Status: Over Budget!

Enter command: delete 3
Transaction already marked as DELETED.

Enter command: delete 4
Invalid position.

Enter command: 
```

```
Enter command: quit

Saving transactions to file...
Done. Exiting program.
root@LAPTOP-S1DQN0HU:/mnt/c/Users/A
```

```
logs > transaction_log.txt
INC|Painting|200.55
EXP|nightreign dlc|-500.35

```