

**Computer Science Department
San Francisco State University
CSC 340
Spring 2016**

Assignment 2 - Enumerations and Structs

Due Date

Wednesday, March 2, at midnight.

Overview

The purpose of this project is to begin to introduce some simple domain objects, using **structs** into our budget application: **Budget**, **Account**, and **Envelope**. In addition, we will use **Enumerations** to help to clean up our Menu code.

Submission

See the submission guidelines posted on iLearn.

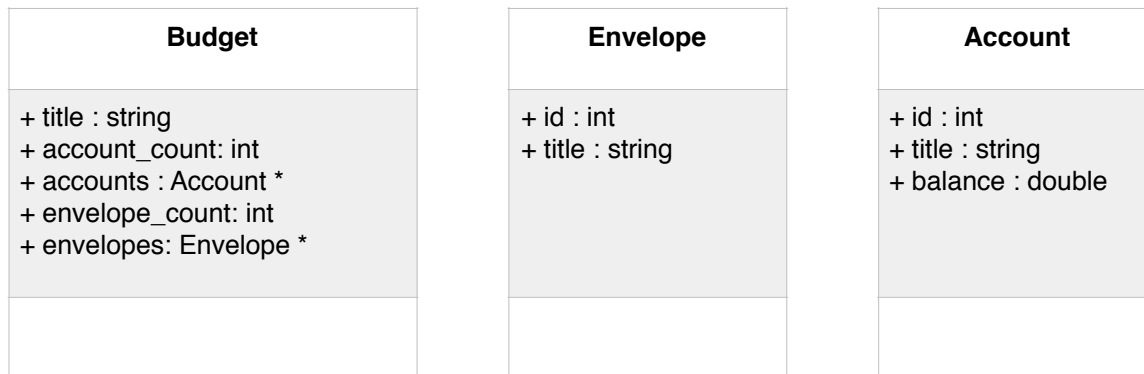
Requirements

1. Implement the menu system using **enums** for the options **EXIT** and **SHOW_BUDGET**.
 - 1.1. At each iteration, the menu should be displayed, a selection obtained from the user and then processed
 - 1.2. The menu should loop until the **EXIT** option is selected
2. Implement the structs discussed in class, as defined in the UML diagrams below
 - 2.1. **Budget**, **Account**, and **Envelope**
 - 2.2. All file input and output must be handled using these structs!
3. Implement the following functions:
 - 3.1. **void showBudget(Budget &)**: Displays the budget (see the sample output below)
 - 3.2. **Budget loadBudget()**: Loads a budget by reading a file in the current directory named **BUDGET**, and returns that budget. Note that this function should only be called once in this application!
 - 3.3. **void storeBudget(Budget &)**: Stores the budget into the **BUDGET** file. Contents of the **BUDGET** file will be replaced by the contents of the budget provided as a parameter.
 - 3.4. **void destroyBudget(Budget &)**: Destroys the budget. This should only be called once! Note that I added this after our conversation in class; since we have to dynamically allocate memory, we need to be good programmers and clean it up, too.

Resources

The makefile for this assignment, as well as a sample **BUDGET** file, can be found at <https://gist.github.com/jrob8577/127f9f00c72866f3b046>.

Appendix A: UML Diagrams



Appendix B: Budget File Format

File Format	Sample Data
TITLE ENVELOPE_COUNT ENVELOPE_ID ENVELOPE_TITLE // Repeated up to ENVELOPE_COUNT ACCOUNT_COUNT ACCOUNT_ID ACCOUNT_TITLE ACCOUNT_BALANCE // Repeated up to ACCOUNT_COUNT	My Budget 2 1 Rent 2 Groceries 2 345 Checking 10.00 876 Savings 100.00

Note the output format of the decimals (precision is 2, decimal point shown).

Sample Output

User input is *italicized, in bold*.

→ **make**

```
g++ -c main.cpp
```

```
g++ main.o -o driver
```

→ **cat BUDGET**

```
NoSpacesInTitleBudget
```

```
2
```

```
123 WoW
```

```
456 Groceries
```

```
2
```

```
789 Checking 100
```

```
120 Savings 55.55
```

→ **./driver**

```
1. Show Budget
```

```
0. EXIT
```

```
Enter your selection: 1
```

```
----- NoSpacesInTitleBudget -----
```

2 Accounts:

789: Checking - 100

120: Savings - 55.55

2 Envelopes:

123: WoW

456: Groceries

1. Show Budget

0. EXIT

Enter your selection: **0**

→ **cat BUDGET**

NoSpacesInTitleBudget

2

123 WoW

456 Groceries

2

789 Checking 100

120 Savings 55.55

→