CIS2520

Lab#2: Avoid pointing

Week 1: Sept 18th, 2017

# Overview

In this lab session, students will develop a C program that utilizes void pointers and function pointers to generate a reusable interface. This lab must be submitted on CourseLink to receive marks, please see the due date information on CourseLink.

# Learning Outcomes

Upon successful completion of this lab students should be able to:

* Implement an interface with void pointers
* Apply knowledge of function pointers to dynamically alter program functionality
* Practice pointer usage and proper memory management

# Review Material

* Pointers
* Void pointers
* Function Pointers

# Introduction

In this lab, students will begin building a small store that sells books and board games. The store will not actually do that much but it will have inventory that should be displayed.

# Getting Started

Every submission for this course must use the proper project directory structure. Start by creating a project root directory and including all the require subfolders and mandatory files (See Lab 1).

# Building a Project

## Book Header File

Take the Book structure and related function code from the course GitHub repository: <https://goo.gl/sZbbyV>

(Alternatively, the path from Github main is lectures/cReviewLecture/Book/bookMain.c )

Convert the book file into two files (header and source) and put them in the appropriate project folders.

## Main Program

When creating code, we always want to test our code as early as possible to identify bugs before moving to the next task. We should test that book, main, and our makefile all work together. Create a main for your project (called main.c) add a single Book in the main. Modify your makefile so that it compiles without error.

## Board Game Header File

Declare a header file called ‘boardgame.h’ in your include folders. Create a struct called BoardGame that has the following members: name, price, and brand name.

Declare accessors and mutators for each member in the header file and implement them in your source file. Also, declare and implement a print function that displays the information with a tab between each piece of information. Your print function should follow the declaration of the Book print function.

Ex>

<name> <price> <brandName>

Make sure to comment your code using Doxygen style formatting and include BoardGame in your main.

## Store Header File

Now that we have items to sell let's add a basic store, in a new file called store.h create a struct called Store. Note, this store will only hold one item at a time for this lab. The store will contain two members a void pointer to an ‘item’ and a function pointer that returns void and has one argument passed in.

**Question:** What type should that one argument be if this function will print either books or board games?

Declare, comment, and implement the store member accessors and mutators functions.

The idea of a function pointer as a struct member is probably novel, recall from class that function pointers mostly act the same as other pointers. This means you can store or pass them as you would with other pointers.

Declare and implement the following function:

void displayStore( Store\* s );

This will be the only function used in the main to display information.

## Main Program

1) Create pointers to a Store, two books, and two board games, you may select all the information.

2) Set the store to display the two books one per line. (Do not call directly call your printBook)

3) Set the store to display the board games one per line. (Do not directly call your printBoardGame)

**Question:** Change only the store item to be a book, what happens when you try to display the store?

Change that line back so that the two books and two board games are displayed.

# Submission

You must submit all files associated with this lab except files generated by running your makefile. Your makefile does not need a separate rule for testing in this lab.

Your program must run and output the requested information from the Main Program Section. Any book or board game information may only be displayed using the displayStore function. To achieve full marks, you must use the pointers as described.

# Submission Guidelines

Your source code must be organized and styled according to the course coding guidelines. Submit a single tar.gz file to the Dropbox. **Do not** use zip, rar, or any other compression program. The tar file should unpack to a folder that is your login name. Inside that folder should be your src/ bin/ include/ assets/ docs/ and lib/ folders. Do not submit any binary files.

**Do** submit all source files, your testing files, a makefile, and a README file.

Follow all other submission guidelines provided for this course.

# Grading

This lab is worth 2% of your final mark and will be graded out of 10 marks. You can lose marks for not following the submission guidelines, but will not be given marks for that step. Marks are given for successfully demonstrating mastery of the learning outcomes for this assignment.