W05 Task Submission (Pointers and Memory Allocation)

You should watch the videos under Instructor Videos for this module before completing these tasks. You may need to perform additional research to solve these tasks. You are encouraged to use the Support Forum to get help with these tasks. After completing your tasks, please submit your original C source files. You do not need to submit files you did not write. In each of your programs, use comments before each section of code to explain what the code does.

1. Write a function that reads two numbers from the user using scanf and stores the values entered into pointer arguments. This function will return the minimum of the two numbers. The function header should look like this:

```
int find_minimum(int *a, int *b);
```

The main function will declare two local variables, pass their addresses into the function, and display the minimum of the two numbers.

2. Write a program that represents account information for a bank. First, create a type that has the following information: account number (numeric), name (string), account type (character), balance (numeric), and date established. The date should itself be a structure consisting of a day, a month, and a year. The account structure should also contain a pointer to another account structure.

Create two functions: the first function will allow the user to input information about the account. The second function will display the information about the account. Both functions should take a pointer to the structure as a parameter.

Your main function will dynamically allocate memory to be used to store one bank account record. Use your functions to have the user input the information into the record and display the information of this record. Then free up the memory used by the record.

3. Extend the previous project to enable a menu system interface to your account system. This menu should give the user two choices: Add a new record, and display all the information in all of the records. You should create new specialized functions to provide this functionality.

4. Extend the previous project to add a new menu item: Find account. If selected, ask the user for an account number, and search your list of records for that account number. If found, display the record contents. If not found, display an appropriate error message.

5. Extend the previous project to add a new menu item: Update an account. If selected, ask the user for an account number, search for that account record, and if found, give the user the opportunity to re-enter the information stored in that record.

6. (Advanced, Optional) Extend the previous project to add a new menu item: Delete an account. If selected, ask the user for an account number, search for that account record, and if found, delete that record while retaining all of the other records.

7. (Advanced, Optional) Modify the previous project by storing the bank records in order by account number. Whenever a bank record is added, insert it into the linked list in the right place so that when the records are displayed they are in order by the account number. When a record

is modified, remove it from the linked list and re-insert it into the list so that the account order is maintained.