Lab 1213 Pointers/Structures/linked lists/Dynamic Memory Allocation – ComSc 265

Dynamic memory allocation allows you to create and destroy objects in memory during the execution of the program. Dynamic declarations are different form Static declarations. Static declarations are written when the programmer write the source code.

When you do a normal declaration, you know the Variable Name(alias) and can get the address.

With dynamic memory declaration/allocation, you only get the address.

Syntax: type \* ptrObject = New type;

Syntax: delete ptrObject;

Points(40)

1. Create a file of charge card purchases, called ‘CustomerPurchases’.

Use 5 different charge card numbers, each with 6 different purchases.

Use the following for the record layout:

Account Number, First Name, Last name, Item Name, Cost, Item Description, and Quantity

Example: 1001, Joe, Smith, TV, 1500.00, 51’’ Sony Plasma HD, 1

1. Read in the file into the following:

Create a structure for purchases, as defined above, called ‘purchase’

Create an ‘array’ of pointers, of the purchases structure type, called ‘Customers’.

Note: ‘Customers’ is an array of pointers, not a pointer variable to an array.

1. Read in the purchases file into the customer array, each member in the array is a different customer, each customer member has a sprouting linked List of purchases.

NOTE: Each element of the array can have its own linked list hanging from it.

Dynamically create a purchase structure for each purchase record read from the file.

Sequentially read each element of the ‘array’.

Check to see if that charge card is already in the ‘array’

If yes, then add the purchase to the ‘linked list’ hanging off that array element

If not, then add the purchase record to the next unused ‘array’ element

Close file after read last record.

1. Create individual functionality to:
   1. Add a purchase to the ‘linked list’ for an existing person, found at an array index
   2. Add a purchase to the array for an new person
   3. Find if customer name is in list
   4. Find if charge number is in list
   5. List Names of Purchasers
   6. List of Charge Card Numbers
   7. List all purchases for a selected Name
   8. Total amount purchased from one customer
   9. Total amount purchased from all customers
   10. Other function as required.

Note: functionalize your program.

1. Save updates to new file called ‘purchaseUpdates’