Team 8’s Laptop Shop

By
Chi Huynh, Nikita Klimov, Ethan Ly, Angelo Rosario, Le Duy Vu

Preview

- Overview of Laptop
- Data Set Fields
- UML Diagram
- Results

Overview

Our store sells a variety of laptops. A laptop is a portable personal computer that is suitable for mobile use. Its name comes from "lap", as it was deemed to be placed for use on a person's lap. Laptops combines the components, inputs, outputs, and capabilities of a desktop computer and come in different forms with many combinations of specifications.

Data Set

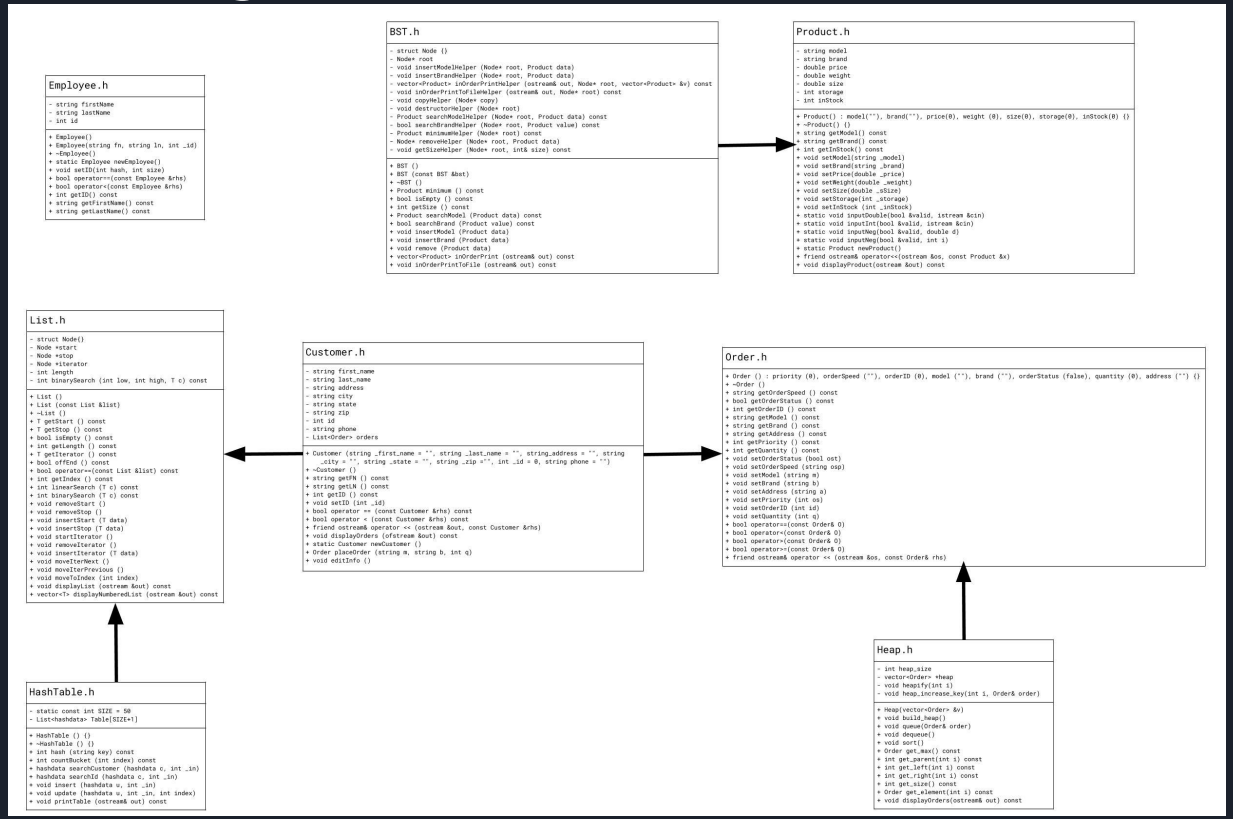
All laptop data sets were found on Amazon.com

The data set fields are:

Model, Brand (string)
Price, Screen Size , Weight (double)
Storage , # In Stock (int)

These fields were used to distinguish each laptop from another.

UML Diagram



https://goo.gl/uDYcvr

BST.h

```
- struct Node {}
- Node* root
- void insertModelHelper (Node* root, Product data)
- void insertBrandHelper (Node* root, Product data)
- vector<Product> inOrderPrintHelper (ostream& out, Node* root, vector<Product> &v) const
- void inOrderPrintToFileHelper (ostream& out, Node* root) const
- void copyHelper (Node* copy)
- void destructorHelper (Node* root)
- Product searchModelHelper (Node* root, Product data) const
- bool searchBrandHelper (Node* root, Product value) const
- Product minimumHelper (Node* root) const
- Node* removeHelper (Node* root, Product data)
- void getSizeHelper (Node* root, int& size) const
```

```
+ BST ()
+ BST (const BST &bst)
+ ~BST ()
+ Product minimum () const
+ bool isEmpty () const
+ int getSize () const
+ Product searchModel (Product data) const
+ bool searchBrand (Product value) const
+ void insertModel (Product data)
+ void insertBrand (Product data)
+ void remove (Product data)
+ vector<Product> inOrderPrint (ostream& out) const
+ void inOrderPrintToFile (ostream& out) const
```

Product.h

```
- string model
- string brand
- double price
- double weight
- double size
- int storage
- int inStock
```

```
+ Product() : model(""), brand(""), price(0), weight (0), size(0),
      storage(0), inStock(0) {}
+ ~Product() {}
+ string getModel() const
+ string getBrand() const
+ int getInStock() const
+ void setModel(string _model)
+ void setBrand(string _brand)
+ void setPrice(double _price)
+ void setWeight(double _weight)
+ void setSize(double _sSize)
+ void setStorage(int _storage)
+ void setInStock (int _inStock)
+ static void inputDouble(bool &valid, istream &cin)
+ static void inputInt(bool &valid, istream &cin)
+ static void inputNeg(bool &valid, double d)
+ static void inputNeg(bool &valid, int i)
+ static Product newProduct()
+ friend ostream& operator<<(ostream &os, const Product &x)
+ void displayProduct(ostream &out) const
```

Order.h

```
- int orderID
- string model
- string brand
- string orderSpeed
- string address
- bool orderStatus
- int priority
- int quantity
```

```
+ Order () : priority (0), orderSpeed (""), orderID (0), model (""), brand (""),
      orderStatus (false), quantity (0), address ("") {}
+ ~Order ()
+ string getOrderSpeed () const
+ bool getOrderStatus () const
+ int getOrderID () const
+ string getModel () const
+ string getBrand () const
+ string getAddress () const
+ int getPriority () const
+ int getQuantity () const
+ void setOrderStatus (bool ost)
+ void setOrderSpeed (string osp)
+ void setModel (string m)
+ void setBrand (string b)
+ void setAddress (string a)
+ void setPriority (int os)
+ void setOrderID (int id)
+ void setQuantity (int q)
+ bool operator==(const Order& O)
+ bool operator<(const Order& O)
+ bool operator>(const Order& O)
+ bool operator>=(const Order& O)
+ friend ostream& operator << (ostream &os, const Order& rhs)
```

Heap.h

```
- int heap_size
- vector<Order> *heap
- void heapify(int i)
- void heap_increase_key(int i, Order& order)
```

```
+ Heap(vector<Order> &v)
+ void build_heap()
+ void queue(Order& order)
+ void dequeue()
+ void sort()
+ Order get_max() const
+ int get_parent(int i) const
+ int get_left(int i) const
+ int get_right(int i) const
+ int get_size() const
+ Order get_element(int i) const
+ void displayOrders(ostream& out) const
```

Priority = 0 - (real time in seconds + days to ship * 86400)

Customer.h

```
- string first_name
- string last_name
- string address
- string city
- string state
- string zip
- int id
- string phone
- List<Order> orders
```

```
+ Customer (string _first_name = "", string _last_name = "", string_address
      = "", string _city = "", string _state = "", string _zip ="", int _id
      = 0, string phone = "")
+ ~Customer ()
+ string getFN () const
+ string getLN () const
+ int getID () const
+ void setID (int _id)
+ bool operator == (const Customer &rhs) const
+ bool operator < (const Customer &rhs) const
+ friend ostream& operator << (ostream &out, const Customer &rhs)
+ void displayOrders (ofstream &out) const
+ static Customer newCustomer ()
+ Order placeOrder (string m, string b, int q)
+ void editInfo ()
```

HashTable.h

```
- static const int SIZE = 50
- List<hashdata> Table[SIZE+1]
```

```
+ HashTable () {}
+ ~HashTable () {}
+ int hash (string key) const
+ int countBucket (int index) const
+ hashdata searchCustomer (hashdata c, int _in)
+ hashdata searchId (hashdata c, int _in)
+ void insert (hashdata u, int _in)
+ void update (hashdata u, int _in, int index)
+ void printTable (ostream& out) const
```

List.h

```
- struct Node{}
- Node *start
- Node *stop
- Node *iterator
- int length
- int binarySearch (int low, int high, T c) const
```

```
+ List ()
+ List (const List &list)
+ ~List ()
+ T getStart () const
+ T getStop () const
+ bool isEmpty () const
+ int getLength () const
+ T getIterator () const
+ bool offEnd () const
+ bool operator==(const List &list) const
+ int getIndex () const
+ int linearSearch (T c) const
+ int binarySearch (T c) const
+ void removeStart ()
+ void removeStop ()
+ void insertStart (T data)
+ void insertStop (T data)
+ void startIterator ()
+ void removeIterator ()
+ void insertIterator (T data)
+ void moveIterNext ()
+ void moveIterPrevious ()
+ void moveToIndex (int index)
+ void displayList (ostream &out) const
+ vector<T> displayNumberedList (ostream &out) const
```

Employee.h

```
- string firstName
- string lastName
- int id
```

```
+ Employee()
+ Employee(string fn, string ln, int _id)
+ ~Employee()
+ static Employee newEmployee()
+ void setID(int hash, int size)
+ bool operator==(const Employee &rhs)
+ bool operator<(const Employee &rhs)
+ int getID() const
+ string getFirstName() const
+ string getLastName() const
```

Walk-Through

Functionality tested	Feedback
Product search, remove, add	Anticipate for bad user input ● Only numbers should be accepted as price and stock values when adding a product ● *Incorporated into our program*
Allow for user to change his/her personal information	This did not exist, but it is implemented ● User can change his name or shipping address ● If name is changed, user gets new ID

Walk-Through

Functionality tested	Feedback
Search product and place order	*Searching for model shouldn't be case-sensitive* *-Place order has the same issue*
View Purchases	Provides price, shipping, and tax