Momentum Software Engineering

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

CMPT 275 Group T

Momentum Software Engineering's

Flight the Freights Manager
Detailed Design Document
Revision 1

July 19, 2002

Approved by: (By alphabetical order)

Vincent Chu _____
Steven Essen _____
Sung Hwang _____
David Jeong _____

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

Revision History Page

Revision 1	-Written jointly by David Jeong, Sung Hwang, Steven Essen and Vincent Chu. Compiled together into one document.

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

Table of Contents

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

Section 1.0 – Discussion of Detailed Design

Much of the discussion of the implementation of the modules is contained in the comments of the modules themselves. This section deals with issues that primarily affect more than one module. For discussion on an individual module, please refer to the comments in the appropriate source code at the back of this binder.

1.1 Overview

The implementation of the modules was based on the guidelines and scenarios set out in the User Manual and the Architectural Design. Helper functions such as reading user input from the keyboard, padding variables for output to printer or text file, asking yes/no questions, displaying the content of a record, doing binary file I/O, and so on were included in each module as to simplify the coding. Each module has been individually tested but not yet linked all together (except Cargo, Customer, and Flight which have been linked to their respective file I/O module). Note that init() must be called on all non-file I/O modules in order to open the files correctly. Stub and driver testing were performed, two of which are presented in Appendix A of this document.

1.2 Weaknesses

Some weaknesses in the implementation are:

- no sorted records by key/ID in the files
- no rounding of real numbers to a certain decimal place when storing in file or doing output (without padding)
- inefficient linear searches for records
- some modules might not check for valid ranges of input or data formatting

1.3 Speed and Memory Inefficiencies

The implementation has some speed and memory inefficiencies. One issue is that file records are not sorted and linear searches are performed so finding a particular record could take long. However, insert new records are easily put at the end of the file without the need for going through the entire file to place the new records in the right spot. Thus, accessing a particular record is relatively slow access – B+ trees some other efficient data structure could have been used for file access.

Also, when deleting a record from file, it requires moving the last record of the file to the position of the deleted record and truncating file. This won't work well for sorted file records, as the correct order of the records cannot be maintained. A better way has not been found in Java

Despite memory (RAM) being way faster than disk, a good thing is that all the instances of one object are not stored in memory but on disk. However, some functions require retrieving a dynamic array of all the cargo items for a flight (which means some cargo items must temporarily be put into memory and thus, slow down the system).

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

Derived attributes require extra space for the flight record but they help eliminate the complex calculations required when finding the weight and footprints remaining on the flight. This reduces the amount of time spent on computer processing when inquiring about a flight.

1.4 When Porting to Another Operating System or Different Computer

When running Flight the Freights Manager on another operating system or different computer, the location of the Cargo, Customer, and Flight data file may have to be changed in the respective file I/O files as well as the location of flight load report (which is changed in the Flight module).

The same file I/O functions should not pose a problem on different operating systems as Java is capable of working on different operating systems.

1.5 Alternatives

In this subsection, particular implementation details are discussed in relation with alternatives.

Padding (with spaces or zeros as well as chopping but not rounding) is done on all attributes when outputting a list to the screen or to the printer. If this wasn't done, the columns won't be aligned properly and the list would look messy. Since Momentum is striving for user friendliness, padding was necessary to see clearly the option numbers and the ID string or number.

Un-ordered fixed-length records were used. As discussed above, despite allowing quick inserts, accessing a particular record specified by an ID (for inquiring and doing ot her operations) is quite slow and inefficient. Non-linear searches (using sorted records) s uch as binary search or data structures such as B+ trees could have been used to speed up looking for a record (although inserting and deleting could be slow and inefficient). How ever, since very crude file organization is only required for the data files and due to the li mited amount of time required for the implementation phase, un-ordered fixed-length rec ords are used.

Sub-class implementation of Cargo was done mainly using a roll-down strategy w here all the attributes of the subclasses are stored in the subclasses' file; thus, there are tw o files, one for containers and the other for pallets. The disadvantage of this is one does n ot know which file has the right cargo in question. The advantage of this over roll-up is t hat files have records of the same length and since there are a lot more containers than pal lets, each container (which are short in length) are stored only in the container file. A spli t strategy was also considered where attributes airID, weight, flightID, and isPallet are in one file and airID, length, and width in another file. isPallet asks as the type discriminato

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

r which makes it easy to determine the type of cargo item; thus, if the cargo item is a pall et, one can automatically look in the pallet file. The drawback is that airID has to be stor ed in the pallet file (acts as a foreign key). The decision made was to use modified roll-d own strategy which uses the type discriminator feature of the split strategy; therefore, dis criminating between the two types of cargo item is made easier.

The function getCargoOnFlight() returns a Vector (dynamic array) consisting of all cargo items (objects) on a particular flight. The Vector class is then, used by operations that need those cargo items. Vectors (dynamic arrays) are used because they are easy to manipulate when inserting and deleting items and they automatically truncate the array size when items are deleted. However, the items are not sorted. Static arrays are the opposite: they can be unsorted or sorted but the array size is fixed and deletion of an item is not efficient as gaps are left between remaining items. Thus, since unsorted items are only required, vectors are used.

Appendix A – Unit Testing

A.1 – Using a Driver on a Particular File I/O Module

Test Cases

Class: Unit Test
Type: Using Driver to test fCargo (Structual)
Description:

- The binary file I/O class for cargo (fCargo.class) deals with creating/modifying and deleting records for pallet and container records. Pallet records and container records are stored in two different files. This test case, using a driver, would uncover any bugs that occur in manipulating records. When the records are being manipulated, the function written in this class calls the operating system to handle the files. We want to test to see if those functions calls are appropriate and handle the files (and hence the records correctly)
- A driver is needed for fCargo.class because it's on the lowest level of the module hierarchy (that is, there is no classes below fCargo).
- The test would attempt to create some pre-defined records, modifying a few of them, and then print out a list of items that are already stored in the record.
- 90% of all the branches are covered
- This is a black box test, and the entire test case is written before the File I/O class is implemented by a ~~procrastinating~~ programmer

Preconditions:

- The module fCargo.java must have been compiled to fCargo.class

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

- Cargo.class is needed for fCargo.class. A stub for Cargo.class is needed. The stub Cargo.java must also have been compiled to Cargo.class
- The hard disk c:\ (where the test record files are stored) must not be full (i.e. must have at least 1MB free space). The program should have read and write access to the directory.
- c:\pallet.dat and c:\container.dat does not already exist.
- We do not run the driver test from the main module. We need to type java fCargo.class to start running the driver.

Exact Test Input:

1. Start the driver test by typing java fCargo.class.

The test would run automatically, and print out the output.

According to the implementation in the driver, this is what we have performed:

1. Write 4 container records into container.dat – the record file for containers
2. Write 4 pallet records into pallet.dat – the record file for pallets
3. Replace the third record in container.dat
4. Replace the first record in pallet.dat
5. Delete the fifth pallet
6. Delete all the pallets
7. Delete the second container
8. Delete all the containers

Example Result: (Results are generated by a simple looping program that would println() the expected results – i.e. with the appropriate weight, width, length, name and flight id, etc – the following results are NOT generated by running the test case. The exact results that is generated by the test case are in the next sub-section)

When 5 pallets are added:

```
Now in record file: (pallet)
Name: Vincent Con1
Flight ID: 0001:FEB01
Airway ID: 6
Weight: 106.0
Width: 201.0
Length: 301.0
Pallet

Name: Steven Con1
Flight ID: 0002:FEB02
Airway ID: 7
Weight: 107.0
Width: 202.0
Length: 302.0
Pallet

Name: David Con1
Flight ID: 0003:FEB03
Airway ID: 8
Weight: 108.0
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

```
Width: 203.0
Length: 303.0
Pallet

Name: Sung Con1
Flight ID: 0004:FEB04
Airway ID: 9
Weight: 109.0
Width: 204.0
Length: 304.0
Pallet

Name: sans nom Con5 super over long na
Flight ID: 0005:FEB05
Airway ID: 10
Weight: 110.0
Width: 205.0
Length: 305.0
Pallet
```

When 5 containers are added:

```
Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: Steven Con2
Flight ID: 0002:JAN02
Airway ID: 2
Weight: 102.0
Width: 0.0
Length: 0.0
Container

Name: David Con3
Flight ID: 0003:JAN03
Airway ID: 3
Weight: 103.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

When the first pallet is replaced:

```
Now in record file: (pallet)
Name: Mr. ReplacementPallet
Flight ID: Generic Fl
Airway ID: 12
Weight: 112.0
Width: 206.0
Length: 306.0
Pallet

Name: Steven Con1
Flight ID: 0002:FEB02
Airway ID: 7
Weight: 107.0
Width: 202.0
Length: 302.0
Pallet

Name: David Con1
Flight ID: 0003:FEB03
Airway ID: 8
Weight: 108.0
Width: 203.0
Length: 303.0
Pallet

Name: Sung Con1
Flight ID: 0004:FEB04
Airway ID: 9
Weight: 109.0
Width: 204.0
Length: 304.0
Pallet

Name: sans nom Con5 super over long na
Flight ID: 0005:FEB05
Airway ID: 10
Weight: 110.0
Width: 205.0
Length: 305.0
Pallet
```

When the third container is replaced:

```
Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: Steven Con2
Flight ID: 0002:JAN02
Airway ID: 2
Weight: 102.0
Width: 0.0
Length: 0.0
Container
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

```
Name: Mrs. ThirdContainer
Flight ID: SOSHORT
Airway ID: 11
Weight: 111.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container
```

When the last pallet in record class is deleted:

```
Now in record file: (pallet)
Name: Mr. ReplacementPallet
Flight ID: Generic Fl
Airway ID: 12
Weight: 112.0
Width: 206.0
Length: 306.0
Pallet

Name: Steven Con1
Flight ID: 0002:FEB02
Airway ID: 7
Weight: 107.0
Width: 202.0
Length: 302.0
Pallet

Name: David Con1
Flight ID: 0003:FEB03
Airway ID: 8
Weight: 108.0
Width: 203.0
Length: 303.0
Pallet

Name: Sung Con1
Flight ID: 0004:FEB04
Airway ID: 9
Weight: 109.0
Width: 204.0
Length: 304.0
Pallet
```

When all the pallets are deleted:

```
Now in record file: (pallet)
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

When the second record in containers are deleted:

```
Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container

Name: Mrs. ThirdContainer
Flight ID: SOSHORT
Airway ID: 11
Weight: 111.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container
```

When all the records in container are deleted:

```
Now in record file: (container)
```

A display of the both files are called again:

```
Now in record file: (pallet)
```

Exact Results: (Generated by the test driver:)

```
Now in record file: (pallet)
Name: Vincent Con1
Flight ID: 0001:FEB01
Airway ID: 6
Weight: 106.0
Width: 201.0
Length: 301.0
Pallet

Name: Steven Con1
Flight ID: 0002:FEB02
Airway ID: 7
Weight: 107.0
Width: 202.0
Length: 302.0
Pallet
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

```
Name: David Con1
Flight ID: 0003:FEB03
Airway ID: 8
Weight: 108.0
Width: 203.0
Length: 303.0
Pallet

Name: Sung Con1
Flight ID: 0004:FEB04
Airway ID: 9
Weight: 109.0
Width: 204.0
Length: 304.0
Pallet

Name: sans nom Con5 super over long na
Flight ID: 0005:FEB05
Airway ID: 10
Weight: 110.0
Width: 205.0
Length: 305.0
Pallet


Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: Steven Con2
Flight ID: 0002:JAN02
Airway ID: 2
Weight: 102.0
Width: 0.0
Length: 0.0
Container

Name: David Con3
Flight ID: 0003:JAN03
Airway ID: 3
Weight: 103.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

```
Now in record file: (pallet)
Name: Mr. ReplacementPallet
Flight ID: Generic Fl
Airway ID: 12
Weight: 112.0
Width: 206.0
Length: 306.0
Pallet

Name: Steven Con1
Flight ID: 0002:FEB02
Airway ID: 7
Weight: 107.0
Width: 202.0
Length: 302.0
Pallet

Name: David Con1
Flight ID: 0003:FEB03
Airway ID: 8
Weight: 108.0
Width: 203.0
Length: 303.0
Pallet

Name: Sung Con1
Flight ID: 0004:FEB04
Airway ID: 9
Weight: 109.0
Width: 204.0
Length: 304.0
Pallet

Name: sans nom Con5 super over long na
Flight ID: 0005:FEB05
Airway ID: 10
Weight: 110.0
Width: 205.0
Length: 305.0
Pallet


Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: Steven Con2
Flight ID: 0002:JAN02
Airway ID: 2
Weight: 102.0
Width: 0.0
Length: 0.0
Container
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

```
Name: Mrs. ThirdContainer
Flight ID: SOSHORT
Airway ID: 11
Weight: 111.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container


Now in record file: (pallet)
Name: Mr. ReplacementPallet
Flight ID: Generic Fl
Airway ID: 12
Weight: 112.0
Width: 206.0
Length: 306.0
Pallet

Name: Steven Con1
Flight ID: 0002:FEB02
Airway ID: 7
Weight: 107.0
Width: 202.0
Length: 302.0
Pallet

Name: David Con1
Flight ID: 0003:FEB03
Airway ID: 8
Weight: 108.0
Width: 203.0
Length: 303.0
Pallet

Name: Sung Con1
Flight ID: 0004:FEB04
Airway ID: 9
Weight: 109.0
Width: 204.0
Length: 304.0
Pallet
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

```
Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: Steven Con2
Flight ID: 0002:JAN02
Airway ID: 2
Weight: 102.0
Width: 0.0
Length: 0.0
Container

Name: Mrs. ThirdContainer
Flight ID: SOSHORT
Airway ID: 11
Weight: 111.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container


Now in record file: (pallet)

Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: Steven Con2
Flight ID: 0002:JAN02
Airway ID: 2
Weight: 102.0
Width: 0.0
Length: 0.0
Container
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

```
Name: Mrs. ThirdContainer
Flight ID: SOSHORT
Airway ID: 11
Weight: 111.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container


Now in record file: (pallet)

Now in record file: (container)
Name: Vincent Con1
Flight ID: 0001:JAN01
Airway ID: 1
Weight: 101.0
Width: 0.0
Length: 0.0
Container

Name: sans nom Con5 super over long na
Flight ID: 0005:JAN05
Airway ID: 5
Weight: 105.0
Width: 0.0
Length: 0.0
Container

Name: Mrs. ThirdContainer
Flight ID: SOSHORT
Airway ID: 11
Weight: 111.0
Width: 0.0
Length: 0.0
Container

Name: Sung Con4
Flight ID: 0004:JAN04
Airway ID: 4
Weight: 104.0
Width: 0.0
Length: 0.0
Container


Now in record file: (pallet)

Now in record file: (container)
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

Pass/Fail Decision:

The exact result shows that the file I/O work as expected. The exact layout of the exact result is slightly different from the expected result is an effect of mere presentation difference. Because Momentum is testing the functionality of the File I/O class's ability to create/modify/delete records, these mere display differences have no significance. The important result is that the ordering of records, the number of records in each file in different stages of the program, and the expected content of each record matches what are being dictated in the expected results. It can be said that the File I/O class, rather unexpectedly, pass the test case.

A.2 – Using Stubs on the User Interface

Class: Unit Test
Type: Using Stubs to test the UI (Structural)
Description:

- According to the overall OCD (as shown in the architectural design), the modules communicating with User Interface (UI) module (the unit under test) are:
  - Cargo module
  - Customer module
  - Flight module

  These modules are on lower level than the UI so that is why these three modules need to be stubbed.
- Tests whether menus are being called correctly and whether the right functions are being called for the menu items – the unit test will focus on calling the functions related to the cargo module (one function (reassign) will call the flight module).
- Also tests whether a non-existent menu item is entered by the user and an error message shows up and menu is reshown – the unit test will focus on the add cargo menu (as same technique is used in the another menus).
- 89% of the branch exits/paths have been tested.

Preconditions:

- The modules listed above must have been compiled into .class files
- The program is currently active at the main menu.
- File I/O modules do not have to be opened as they are not required.
- Actual code (found in the ui.class) for running the test case:

```
private static void main (String args[]) throws Exception{
        init();
}
```

Exact Test Input:

1. From the Main Menu, type **2** for Cargo Maintenance and press Enter.
2. In the Cargo Maintenance Menu, type **1** for Add a new cargo item and press Enter.
3. In the Add a new cargo item Menu, type **1** for adding a container and press Enter.
4. Type **3** and press Enter.
5. Type **0** and press Enter to go back to previous menu.
6. Type **2** and press Enter.
7. Type **3** and press Enter.
8. Type **4** and press Enter.

2. 9. Type **5** and press Enter.
10. Type **0** and press Enter to go back to previous menu.

Example Result:

After Step 1:

```
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 2:

```
=======Add a new cargo========
1) Standard Container
2) Pallet
0) Cancel
Choose an option [0-2] and press ENTER:
```

After Step 3:

```
add a container

=======Add a new cargo========
1) Standard Container
2) Pallet
0) Cancel
Choose an option [0-2] and press ENTER:
```

After Step 4:

```
Please enter the number within the range.

=======Add a new cargo========
1) Standard Container
2) Pallet
0) Cancel
Choose an option [0-2] and press ENTER:
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

After Step 5:

```
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 6:

```
assign cargo

======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 7:

```
deassign cargo

======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 8:

```
delete cargo item

======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

After Step 9:

```
reassign Cargo

======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 10:

```
=====Main Menu======
1) Customer Maintenance
2) Cargo Maintenance
3) Flight Maintenance
4) Reports and Inquiries
0) Exit
Enter the number you want(0-4):
```

Exact Results:

After Step 1:

```
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 2:

```
=======Add a new cargo========
1) Standard Container
2) Pallet
0) Cancel
Choose an option [0-2] and press ENTER:
```

After Step 3:

```
add a container
=======Add a new cargo========
1) Standard Container
2) Pallet
0) Cancel
Choose an option [0-2] and press ENTER:
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

After Step 4:

```
Please enter the number within the range.
=======Add a new cargo========
1) Standard Container
2) Pallet
0) Cancel
Choose an option [0-2] and press ENTER:
```

After Step 5:

```
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 6:

```
assign cargo
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 7:

```
deassign cargo
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 8:

```
delete cargo item
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, B.C.
Canada. V5A 1S6

After Step 9:

```
reassign Cargo
======Cargo Maintenance======
1) Add a new cargo item
2) Assign a cargo item to a flight
3) De-assign a cargo item from a flight
4) Delete a cargo item
5) Reassign all cargo items on a flight to a different flight.
0) Go back to Main Menu
Choose an option [0-5] and press ENTER:
```

After Step 10:

```
=====Main Menu======
1) Customer Maintenance
2) Cargo Maintenance
3) Flight Maintenance
4) Reports and Inquiries
0) Exit
Enter the number you want(0-4):
```

Pass/Fail Decision:

The exact result shows that cargo maintenance is being shown and behaving properly when the user enters something outside the range of menu item numbers. It also shows that the right functions are being called in the Cargo class. The effect of mere presentation differences may give slightly different exact results from the expected result; this effect is not really significant. Thus, the user interface (ui) class passes the test case.