UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING

20232024/1

SECJ2013 – DATA STRUCTURES & ALGORITHMS

SECTION 02

GROUP ASSIGNMENT 2 : LINKED LIST

AIRLINE RESERVATION SYSTEM

LECTURER NAME : MRS LIZAWATI MI YUSUF

GROUP NAME : GUSION

NAME	MATRIC ID
CHE MARHUMI BIN CHE AB RAHIM	A22EC0147
MUHAMMAD ARIFF DANISH BIN HASHNAN	A22EC0204
MUHAMMAD IMAN FIRDAUS BIN BAHARUDDIN	A22EC0216

TABLE OF CONTENT

1. Objective

- Able to apply linked list in system (Add and Delete) list of customer based on position (beginning, middle (user selected position) , end)
- Able to sort customer booking list based on name, date of departure, destination and airlines (in ascending order)
- Able to search for information in the list based on the term provided
- Can display related information based on user search term
- Can display a newly sorted and updated list after sorting, and add and delete algorithms.
- To apply data structure operations of sorting and searching, and also linked list application in the system

2. Synopsis

The system's goal is to make managing and modifying a linked list easier by introducing a unique class structure that consists of Cust , Node and LinkedList. The user can work with the system to manipulate the linked list in several ways. Above all, they provide for list growth adaptability by allowing for the addition of additional nodes (customer) at the start, middle, or end of the list. To provide flexibility in various situations, the user can also easily remove any node; the first, middle, or last node. The system makes it easy for users to locate a particular node by providing a search key linked to the Cust class's properties. The system has an added function that lets users sort the list (in ascending order) to improve organisation. Lastly, the viewer has the option to inspect the linked list's whole contents, giving them insight into the nodes' composition and organisation.

3. Class Design

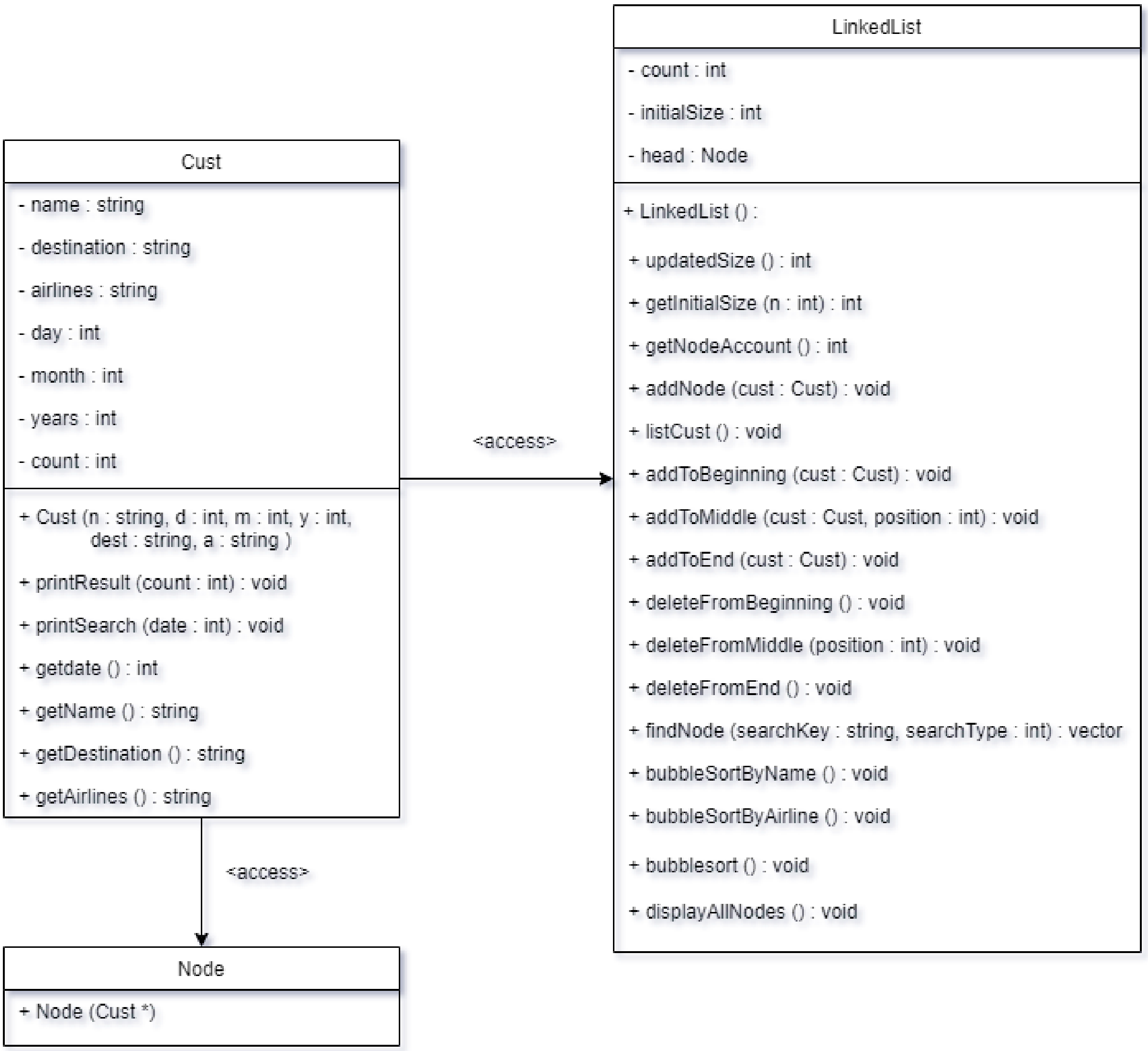
LinkedList
- count : int
- initialSize : int
- head : Node
+ LinkedList () :
+ updatedSize () : int
+ getInitialSize (n : int) : int
+ getNodeAccount () : int
+ addNode (cust : Cust) : void
+ listCust () : void
+ addToBeginning (cust : Cust) : void
+ addToMiddle (cust : Cust, position : int) : void
+ addToEnd (cust : Cust) : void
+ deleteFromBeginning () : void
+ deleteFromMiddle (position : int) : void
+ deleteFromEnd () : void
+ findNode (searchKey : string, searchType : int) : vector
+ bubbleSortByName () : void
+ bubbleSortByAirline () : void
+ bubblesort () : void
+ displayAllNodes () : void
Cust
- name : string
- destination : string
- airlines : string
- day : int
- month : int
- years : int
- count : int
+ Cust (n : string, d : int, m : int, y : int,
dest : string, a : string)
+ printResult (count : int) : void
+ printSearch (date : int) : void
+ getdate () : int
+ getName () : string
+ getDestination () : string
+ getAirlines () : string
<access>
<access>
Node
+ Node (Cust *)

Figure 1: class diagram

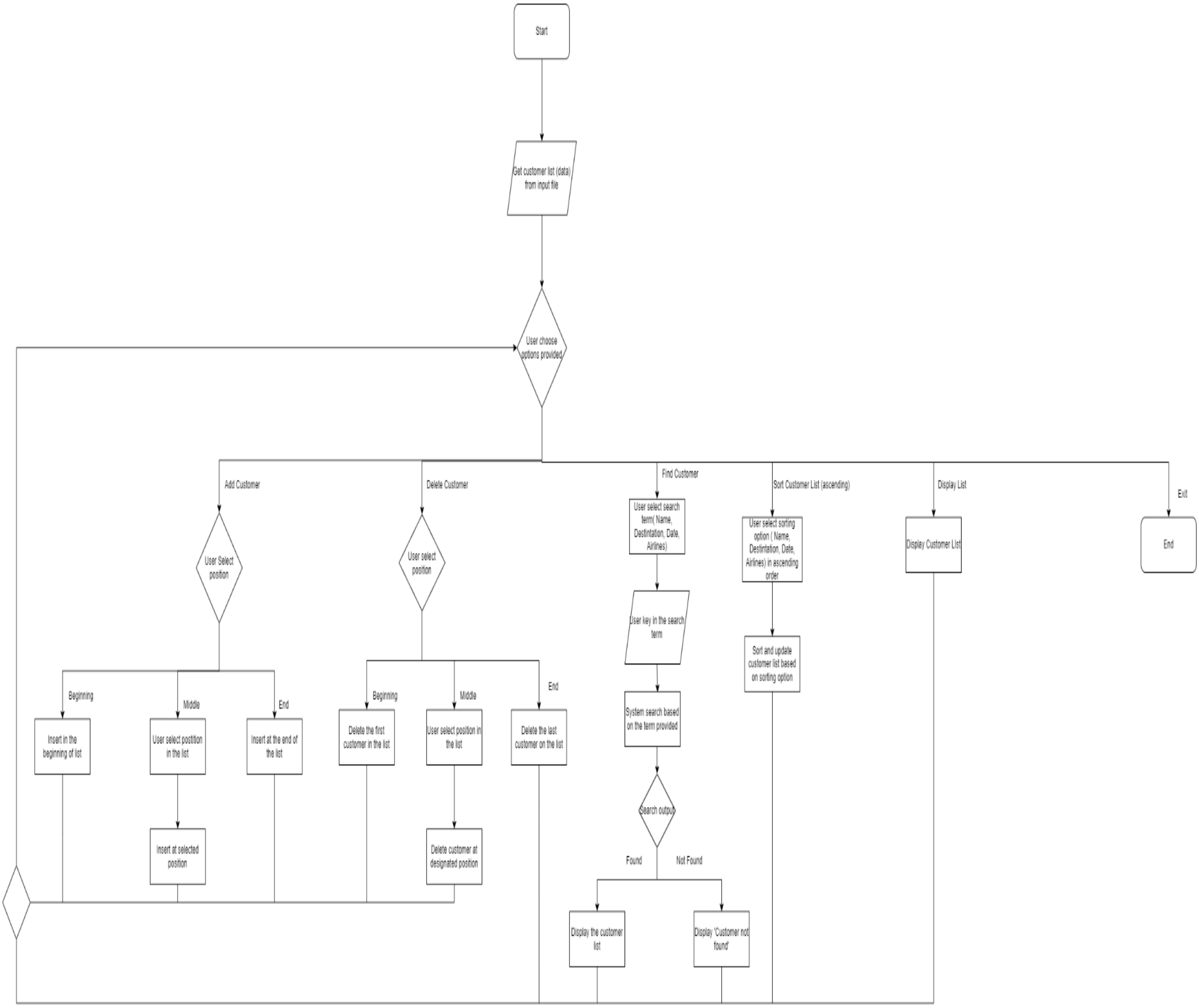
Start
Get customer list (data) from input file
User choose options provided
Add Customer
Delete Customer
Find Customer
Sort Customer List (ascending)
Display List
Exit
User Select position
User select position
User select search term(Name, Destintation, Date, Airlines)
User select sorting option (Name, Destination, Date, Airlines) in ascending order
Display Customer List
End
User key in the search term
Sort and update customer list based on sorting option
Beginning
Middle
End
Beginning
Middle
End
Insert in the beginning of list
User select postition in the list
Insert at the end of the list
Delete the first customer in the list
User select position in the list
Delete the last customer on the list
System search based on the term provided
Search output
Insert at selected position
Delete customer at designated position
Found
Not Found
Display the customer list
Display 'Customer not found'

Figure 2: flowchart of the program

4. Implementation of sorting

The application sorts the linked list of Cust objects according to name and airline criteria, respectively, using independent functions (bubbleSortByName and bubbleSortByAirlines). The Bubble Sort algorithm is used in the programme. Through each iteration of the array, the sorting functions compare neighbouring elements and, if needed, swap them. Users can view sorted lists of client reservations by using these sorting routines when they select ‘4 : Sort the list’ from the menu and display the list when select option 5 in the main menu in the main() function.

5. Implementation of searching

For the searching function in the system, we implemented a sequential search method for the user to search for specific information such as to find customers with the name 'Iman'. Users are able to input their choice to find details based on their need and the system will display the output correctly entered by the user when the user selects ‘3 : Find a customer’ from the menu and if the information does not exist the system will display an error message. We implemented 4 functions to use for searching purposes which are searchByName(), searchByDestination(), searchByDate(), and searchByAirline().

6. Implementation of linked list

We have used a linked list technique in our program code to control the data collection procedure. Our code consists of three primary classes: Cust (which stands for customer), Node (which is a linked list node), and LinkedList (which comprises the overall structure of linked lists). The Cust class has methods for accessing private member variables and displaying client details. The next node in the list and a Cust object are the two pointers that make up the Node class. The main functionalities of Linked List class is adding , deleting , searching and sorting the customer details.