SI2001-5 - Algorithms and Data Structures

Practice I

School of Applied Sciences and Engineering - EAFIT University

Lecturer Alexander Narváez Berrío.

August 2025

PRACTICE I: APPLICATION OF LINKED LISTS IN C++

Objetive: Put linked lists and OOP into practice in solving a specific case using the C++ programming language.

Instrucctions:

Create a text file named 'hotel.txt'. The text file will have the following structure:

name
numberRoom
name
numberRoom
name
numberRoom

Build a doubly linked list that allows you to load each guest's information into a node in the list as follows:

| previous | data1 | data2 | next|

Donde:

- **previous:** will be the pointer to the previous node
- **data1:** will be the name of the guest
- **data2:** will be the room number

SI2001-5 - Algorithms and Data Structures

Practice I

School of Applied Sciences and Engineering - EAFIT University

Lecturer Alexander Narváez Berrío.

August 2025

- **next will be the pointer to the next node** When the list is created, the system must load the information from the text file into the list and offer a menu (non-graphical text interface) that allows the following actions:

- Enter a new guest (save it in the list and in the file with the extension '.txt')

- Search for an existing user by requesting either the name (in which case the room number where they are located must be returned and displayed in a message) or the room number (in which case the name of the guest staying there must be returned and displayed). If the guest is not found, display the message "that guest was not found."

Show list option 1: the system should display the list of existing guests in alphabetical order.

- Show list option 2: the system should show the list of existing guests in order of arrival figure 1.1.

If the room chosen by the customer is occupied, the program should offer them the room before the one they chose, if it is available, or the one after it. If both (the previous and the next) are occupied, ask them to choose a different room number.

Example: the customer requests room 7, but it is already occupied, so the system should recommend room 6 if it is available, or room 8 if room 6 is also occupied. If the neighboring rooms (room numbers, not nodes) are occupied, ask the customer to choose an available number, or the system can generate a room number that is not occupied.

- Consult neighbors: given the room number, display the names of the neighbors (previous and next room numbers) in a message.

SI2001-5 - Algorithms and Data Structures

Practice I

School of Applied Sciences and Engineering - EAFIT University

Lecturer Alexander Narváez Berrío.

August 2025

Hotel

Figure 1.1. Check-in upon arrival at the hotel

Scenario

The manager of a hotel wants to record the name of each guest upon arrival at the hotel, along with the room number they are occupying ->the old guest book.

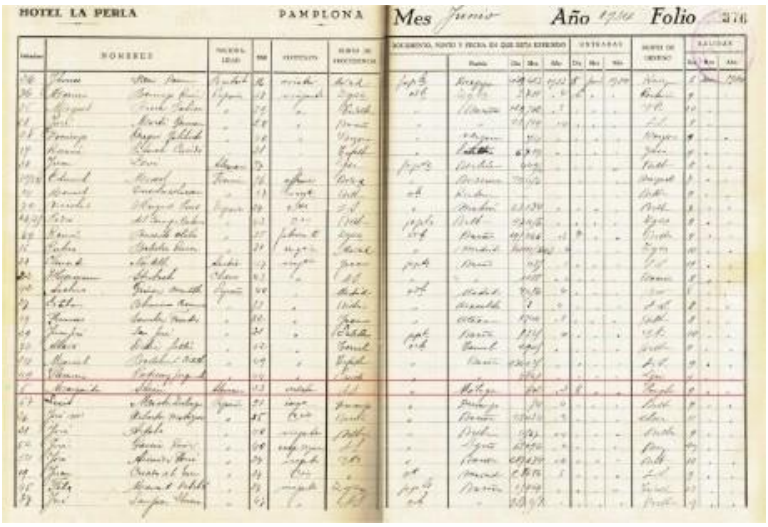

Figure 1.2. old guest book or guest register

You also want to have an alphabetical list of your customers available at any time.

SI2001-5 - Algorithms and Data Structures

Practice I

School of Applied Sciences and Engineering - EAFIT University

Lecturer Alexander Narváez Berrío.

August 2025

Since it is not possible to record customers alphabetically and chronologically in the same list, either separate alphabetical lists are needed or pointers must be added to the existing list, so that only a single list is used.
The manual method in the book required a lot of crossing out and rewriting; however, a computer using a suitable algorithm will do this easily.
For each node in the list, the information or data field has two parts: customer name and room number.

If **x* is a pointer to one of these nodes, L[x]→name and L[x]→numberRoom will represent the two parts of the information field.
The alphabetical listing is achieved by following the order of the pointers in the list (pointer field). A HEADER(S) variable is used to point→ to the first guest.

```
HEADER ← 3
```

Thus, HEADER(S) is 3, since the first customer, Antolín, occupies position 3. In turn, the pointer associated with the node occupied by Antolín contains the value 10, which is the second name of the customers in alphabetical order, and this has the value 7 as its pointer field, and so on. The pointer field of the last guest, Tomás, contains the null pointer indicated by a 0 or a Z. See Figure 1.3.

SI2001-5 - Algorithms and Data Structures

Practice I

School of Applied Sciences and Engineering - EAFIT University

Lecturer Alexander Narváez Berrío.

August 2025

S(=3)

Registro	*Nombre*	*Habitación*	*Puntero*
1	Tomás	324	z (final)
2	Cazorla	28	8
3	Antolín	95	10
4	Pérez	462	6
5	López	260	12
6	Sánchez	220	1
7	Bautista	115	2
8	García	105	9
9	Jiménez	173	5
10	Apolinar	341	7
11	Martín	205	4
12	Luzárraga	420	11
⋮	⋮	⋮	⋮

Figure 1.3. Linked list of hotel guests.

Assessment: this assignment accounts for 15% of the final grade for the course and is weighted as follows:

- Total coding of the project in C++ 10%
- Loading the list from a text file and vice versa 10%
- Use of files (*.h and *.cpp) in classes 10%
- Presentation 70%

Please share the project through the GitHub repository, including a ‘Readme.md’ file (markdown format) with the names of the members, a description of the project, indicating the version of the compiler used, and the URL of the video with the presentation, in which each member can be seen explaining in their own words the main functionalities of the code.

SI2001-5 - Algorithms and Data Structures

Practice I

School of Applied Sciences and Engineering - EAFIT University

Lecturer Alexander Narváez Berrío.

August 2025

Note: this assignment can be completed by one or two students and the deadline for submission is three weeks (see calendar in Interactiva).