VG101 — Introduction to Computer and Programming

Project 2

Manuel — UM-JI (Summer 2018)

- Include simple comments in the code
- If applicable, split the code over several functions
- Extensively test your code and impove it
- Write a single README file per assignment
- Zip all the files and upload the archive on Canvas

1 Project setup

The goal of this project is the design of a program to manage a hotel. The program should be written in C with a special emphasis on structures and pointers.

Note that many questions are left to the appreciation of the programmers, who in turn should argue on their choices based on their knowledge, research and understanding of hotel management. All the decisions should be discussed in the README file.

The minimum set of parameters to define and use is as follows.

- Hotel: more than one floor, each being of different sizes, with flexible bedrooms sizes.
- Visitors: single, couple, family. Visitors can decide to sleep in a dorm, that is sharing their room with others at a cheaper cost.
- Price: depends on the type of room, if breakfast is provided, etc...
- When checking-in a customer receives a card showing
  - The date of arrival
  - The room number
  - The number of keys provided
  - The number of visitors
  - The name of each visitor
  - The ID number of each visitor
- When a customer checks-out he receives a receipt showing
  - The number of nights spent in the hotel
  - The room number
  - The number of breakfasts served
  - The total price

2 Project tasks

Ex. 1 — *Hotel utils*

Using a link list data structure program several functions allowing to read and write information from a text file in the format described in box 1.

When parsing the file, anything following the character "#" should be considered a comment and be ignored. A section represents all the details of a room; each line is composed of a field, some optional spaces, and a value.

Various fields such as the price and number of nights can be updated in the database file; write the corresponding functions.

The functions related to the implementation of the linked list should be written in a file called `hotel-ll.c`. The functions specific to the hotel database format should be written in a file called `hotel-db.c`. This

file should also include functions that can alter the database, e.g. add/remove breakfasts, increase the number of nights, adjust the price, or set the current date.

hotel.db – truncated hotel database file

```
#
# hotel management database file
#

[room 101]
type = dorm # room type (dorm, family, double, single)
vacant=3 # available beds (0/1->occupied/vacant for other room types)
visitors=3 # 3 visitors

name1= John Lovis # name of visitor 1
id1= 123456AF # ID number of visitor 1
arrival1=11/06/2018 # arrival date of visitor 1
keys1=1 # 1 keys given
breakfast1=0 # 0 breakfast ordered, can be increased later on
nights1=1 # number of nights booked, can be increased later on
price1=40 # total price, can be adjusted with respect to breakfast and nights

name2 = Maria Suza
id2 = TV98733K
arrival2=12/06/2018 # arrival date of visitor 1
key2= 2
breakfast2=1 # 0 breakfast ordered, can be increased later on
nights2=2 # number of nights booked, can be increased later on
price2=90 # total price, can be adjusted with respect to breakfast and nights

name3= Eva Turi
id3=00000A
arrival3=12/06/2018 # arrival date of visitor 1
key3=1
breakfast3=0 # 0 breakfast ordered, can be increased later on
nights3=1 # number of nights booked, can be increased later on
price3=40 # total price, can be adjusted with respect to breakfast and nights

[room 514]
type = family
vacant=1

[room 102]
type = double
vacant = 0
name1=Malcolm Man
id1=012add2
name2=Malcolm Woman
id2=111sde3
arrival=15/05/2018
keys=2
breakfast=4
nights=2
price= 210

[room 103] # file truncated here
```

Box 1: Sample file format

Ex. 2 — *Hotel management*

1. Write a function which initialises a hotel and its corresponding database file.

2. Write a function taking care of customers checking-in. It should at least update the hotel database file, and return a card. The card should be generated in a different function.
3. Write a function taking care of customers checking-out. It should at least update the hotel database file and return a receipt. The receipt should be generated in a different function.
4. Write some functions that allow the number of nights to be increased and the number of breakfast to be increased or decreased.
5. Write a function which given a type of room returns the list of available rooms and their number.
6. Write a function which returns the list of all the guests, sorted in alphabetical order, their ID, and how many they are in the hotel. Implement a recursive algorithm.
7. Write a function which returns how much money has been received at the end of a day. Assume customers pay on a daily basis.

Write all the previous functions in a file called `hotel-mt.c`.

Ex. 3 — *Demonstration*

In the previous two exercises all the functions assuring the good management of a hotel were defined. Now write a set of functions to simulate a hotel, i.e. generate a random hotel with random visitors that check-in and out over time, etc.. Demonstrate the well functioning of the setup program,

Allow the user to select (i) the current date, (ii) the time span of the demonstration (in days), (iii) the number of rooms per floor. Create a directory called `receipts` and dump all the receipts in it. If the room is not a dorm them each receipt should be named after the `id1` field. Otherwise a receipt should be issues for each visitor.

All those functions should composed a file called `hotel-main.c`.

Ex. 4 — *Optional tasks*

1. Compile exercises 1 and 2 into a library.
2. Write a Command Line Interface (CLI) to manage a hotel.
3. Write a Graphical User Interface (GUI) to manage a hotel. Use a cross-platform toolkit such as GTK+, or QT. Using a toolkit that is incompatible with other platforms will not bring any bonus.

Note: feel free to discuss any other idea for a bonus with the teaching team.

3 Project submission

Before submitting the project on Canvas ensure the project compiles on JOJ.

- A project that has not been submitted to JOJ will not be graded;
- JOJ will compile the code using the flags `-Werror -pedantic -Wall -std=c11`;
- A project that is not compiling on JOJ will not be graded;
- No test case is offered for the project, the goal of JOJ is only to ensure the written code compiles and complies with the C standard;
- If the Canvas submission includes a GUI, please contact the teaching team when uploading the code since it will not compile on JOJ;