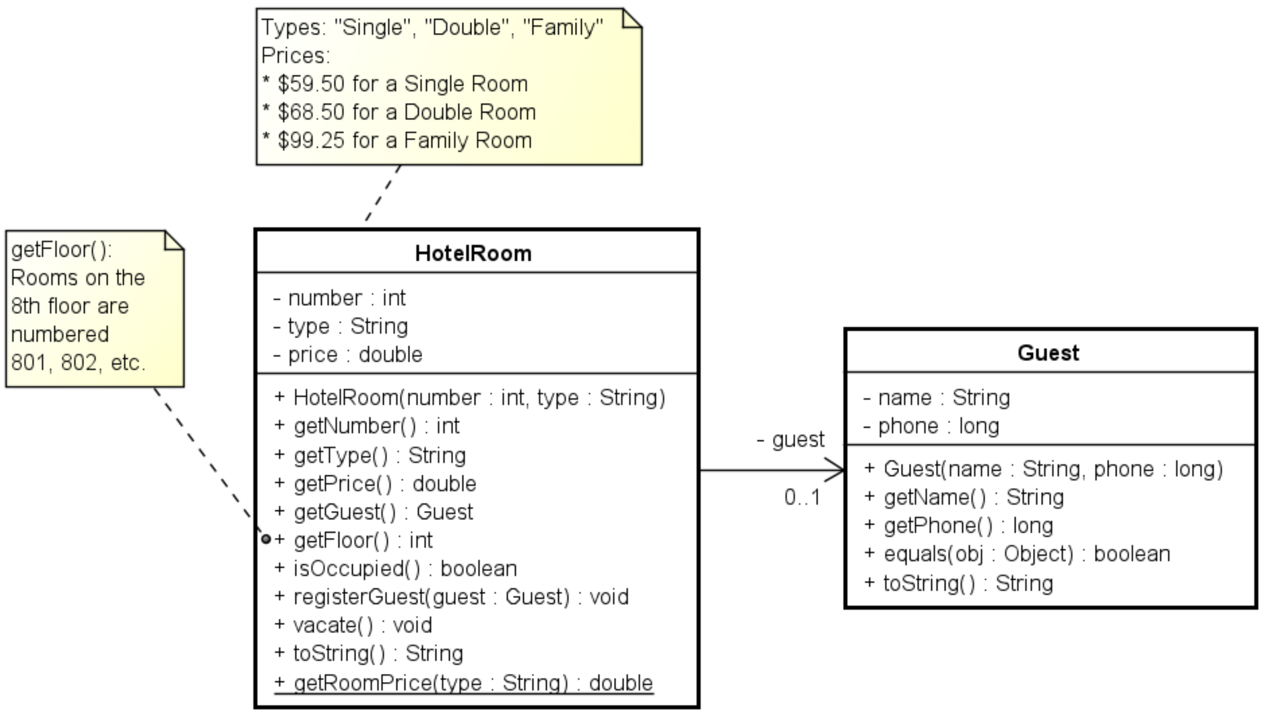
Exercise: A Hotel Room (version 1)

Implement the classes `Guest` and `HotelRoom` shown in the class diagram



Types: "Single", "Double", "Family"
Prices:
* $59.50 for a Single Room
* $68.50 for a Double Room
* $99.25 for a Family Room
getFloor(): Rooms on the 8th floor are numbered 801, 802, etc.
HotelRoom
- number : int
- type : String
- price : double
+ HotelRoom(number : int, type : String)
+ getNumber() : int
+ getType() : String
+ getPrice() : double
+ getGuest() : Guest
+ getFloor() : int
+ isOccupied() : boolean
+ registerGuest(guest : Guest) : void
+ vacate() : void
+ toString() : String
+ getRoomPrice(type : String) : double
- guest
0..1
Guest
- name : String
- phone : long
+ Guest(name : String, phone : long)
+ getName() : String
+ getPhone() : long
+ equals(obj : Object) : boolean
+ toString() : String

Start with class `Guest` (why this class before `HotelRoom`?)

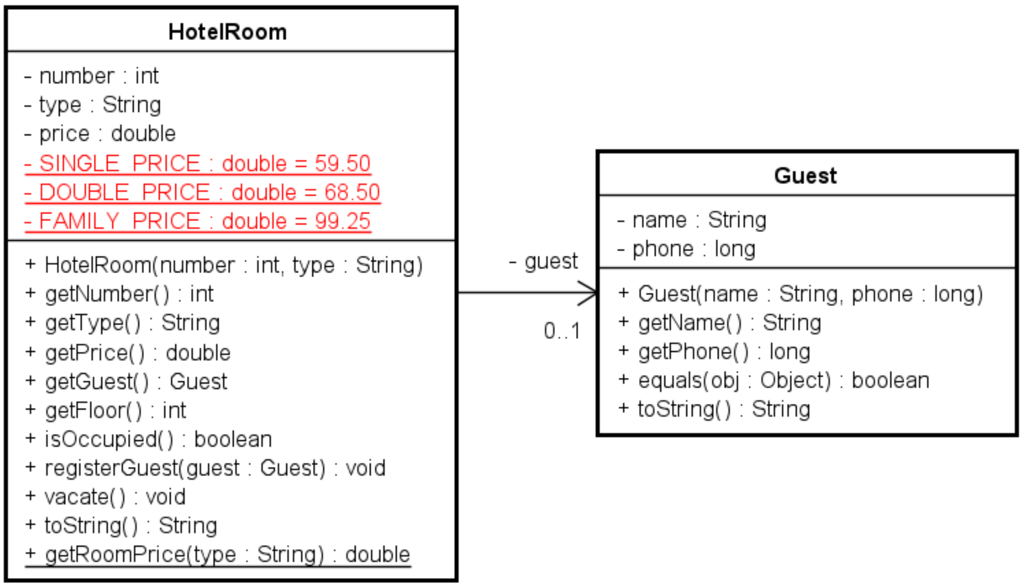
Remarks to class `HotelRoom`:

- Contains 4 instance variables (why?)
- Start with the static method `getRoomPrice` returning the price for a given type of room, use e.g. a switch
- Purpose of the constructor is to initialize all instance variables, set `guest` to `null` (because of the multiplicity 0..1) and use the static method for the price
- Getters for each of the 4 instance variables
- Method `getFloor`, see left note
- Method `isOccupied` checks if guest has been set (is not `null`)
- Method `registerGuest` is a set method in disguise
- Method `vacate` sets the guest to `null`
- Method `toString` return a string with all information. If occupied then the string contain the guest info otherwise the string "available"

Implement a test class with a `main` method and

- Create at least one room and at least one guest.
- Print out the room before and after the guest has been registered to the room.
- Print out the result form calling method `getFloor`
- Test also the static method `getRoomPrice` for all 3 types

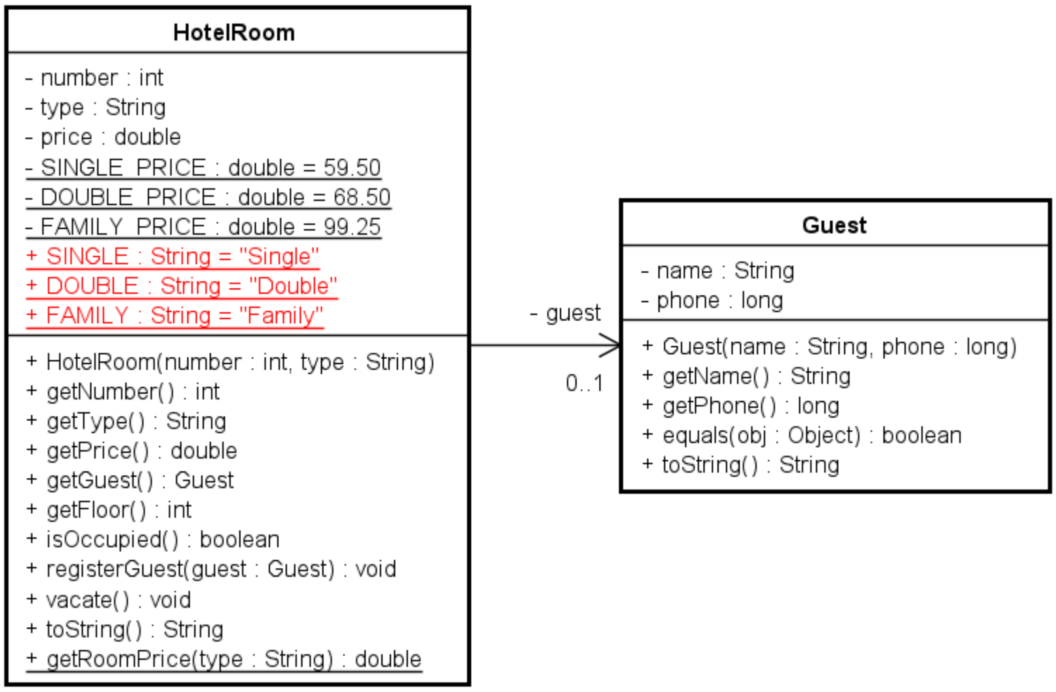
Exercise: A Hotel Room (version 2)



Modify the previous exercise

- Define 3 constants (final) as private and static with the prices as shown in the class diagram above
- Use these constants in the code where you use one of the three prices - which is probably only in the static method `getRoomPrice` (why is this better?)

Exercise: A Hotel Room (version 3)


HotelRoom
- number : int
- type : String
- price : double
- SINGLE_PRICE : double = 59.50
- DOUBLE_PRICE : double = 68.50
- FAMILY_PRICE : double = 99.25
+ SINGLE : String = "Single"
+ DOUBLE : String = "Double"
+ FAMILY : String = "Family"
+ HotelRoom(number : int, type : String)
+ getNumber() : int
+ getType() : String
+ getPrice() : double
+ getGuest() : Guest
+ getFloor() : int
+ isOccupied() : boolean
+ registerGuest(guest : Guest) : void
+ vacate() : void
+ toString() : String
+ getRoomPrice(type : String) : double
- guest
0..1
Guest
- name : String
- phone : long
+ Guest(name : String, phone : long)
+ getName() : String
+ getPhone() : long
+ equals(obj : Object) : boolean
+ toString() : String

Modify the previous exercise

- Define 3 more static constants (static final). This time as public fields with the string values as shown in the class diagram above (why public?)
- Use these constants in your test class (in the `main` method) instead of defining a string as you did before (why is this better?)