



ΟΝΤΟΚΕΝΤΡΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

2016-2017

JAVA



ΔΙΔΑΣΚΩΝΤΕΣ : Ι. ΧΑΤΖΗΛΥΓΕΡΟΥΔΗΣ, Χ. ΜΑΚΡΗΣ, Μ. ΡΗΓΚΟΥ

ΕΤΟΣ : 1ο

ΕΞΑΜΗΝΟ : 2ο

ΠΑΡΑΔΟΣΗ : 30 ΑΠΡΙΛΙΟΥ 2017

ΕΡΓΑΣΙΑ ΟΝΤΟΚΕΝΤΡΙΚΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ 2016-2017

JAVA



Τα μέλη της ομάδας :

ΟΜΟΜΑΤΕΠΩΝΥΜΟ	ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ	EMAIL
Βασδάρης Όμηρος	1054429	vasdaris@ceid.upatras.gr
Καλαματιανού Δήμητρα	1054406	kalamatianou@ceid.upatras.gr
Κωστορρίζος Δημήτρης	1054419	kostorrizos@ceid.upatras.gr



Μια σύντομη περιγραφή του προγράμματος :

- ☞ Η κλάση **Booking** έχει μεθόδους οι οποίες ορίζουν το όνομα πελάτη (set/getClientName), τον κωδικό κράτησης (set/getBookingID), την ημέρα της άφιξης (set/getArrivalDate), τις μέρες διαμονής (set/getAccommodationDays), το πλήθος των ατόμων (set/getPeopleNum) και το δωμάτιο (set/get Room). Συγκεκριμένα η μέθοδος setRoom αναθέτει το αντικείμενο το οποίο είναι τύπου **Room** στην μεταβλητή room της κλάσης **Booking**. Η μέθοδος printBooking εκτυπώνει το όνομα πελάτη, τον κωδικό κράτησης, την ημέρα άφιξης και τις μέρες διαμονής. Την έχουμε δημιουργήσει για να βλέπουμε ποια είναι τα στοιχεία της τυχαίας κράτησης.
- ☞ Η κλάση **Room** έχει μεθόδους οι οποίες ορίζουν τον κωδικό δωματίου (set/getRoomID), την μέγιστη χωρητικότητα (set/getMaxPeople) και την τιμή ανά άτομο (set/getPricePerPerson). Περιέχει ακόμα και τις μεθόδους που ζητήθηκαν (προσθήκης κράτησης : addBooking, Τιμολόγησης : getPriceOfRoom, Ακύρωσης : cancelBooking, Πληρότητας : checkFullness). Αυτή η κλάση έχει δύο μεταβλητές στις οποίες αξίζει να αναφερθούμε. Η

roomInstanceCounter, που είναι static και μετράει το πόσα αντικείμενα δημιουργήθηκαν. Δεν μπορούμε όμως να την χρησιμοποιήσουμε ως μεταβλητή για το ID γιατί κάθε φορά που αλλάζει τιμή θα αλλάζει για όλα τα αντικείμενα. Έτσι δημιουργούμε και άλλη μία μεταβλητή την roomId, που δεν είναι static και κρατά το ID για το δωμάτιο.

- 🖱️ Η κλάση **RoomTypeA** είναι υποκλάση της **Room** και υπερκαλύπτει την μέθοδο `getPriceOfRoom` ώστε η τιμή του δωματίου να υπολογίζεται ανά μέρα διαμονής και όχι με βάση πόσα είναι τα άτομα.
- 🖱️ Η κλάση **RoomTypeB** είναι υποκλάση της **RoomTypeA** και υπερκαλύπτει την μέθοδο `getPriceOfRoom` ώστε να λειτουργεί όπως στην κλάση **RoomTypeA** αλλά για κάθε επιπλέον μέρα διαμονής μιας κράτησης η τιμή της μέρας μειώνεται διαδοχικά με το ποσό `discountPerDay` (έκπτωση ανά μέρα). Η τιμή της μέρας μετά την έκπτωση δεν πρέπει να πέσει κάτω από το 50% της αρχικής τιμής ανά μέρα. Επίσης υπερκαλύπτει την μέθοδο `cancelBooking` ώστε να μην μπορεί να γίνει η ακύρωση του δωματίου επιστρέφοντας μήνυμα "Can't cancel this kind of room".
- 🖱️ Η κλάση **RoomTypeC** είναι υποκλάση της **Room** και υπερκαλύπτει την μέθοδο `addBooking` ώστε να συγκρίνει τα άτομα της κράτησης με τον

ελάχιστο αριθμό ατόμων τις μέρες διαμονής με τις ελάχιστες μέρες διαμονής.


🖱 Η κλάση `RoomTypeD` και η κλάση `RoomTypeE` είναι δύο ακόμα δικές μας κλάσεις που κληρονομούν άμεσα ή έμμεσα την `Room` και υλοποιούν διαφορετικά κάποιες μεθόδους της. Συγκεκριμένα η `RoomTypeD` είναι υποκλάση της `Room` και υπερκαλύπτει την μέθοδο `getPriceOfRoom` και αφορά κρατήσεις μια μόνο μέρας διαμονής. Επιπρόσθετα γίνεται υπερκάλυψη της μεθόδου `cancelBooking` ώστε να μην μπορεί να γίνει η ακύρωση του δωματίου επιστρέφοντας μήνυμα "Can't cancel this kind of room". Από την άλλη η κλάση `RoomTypeE` είναι υποκλάση της `RoomTypeC` και υπερκαλύπτει την μέθοδο `getPriceOfRoom` με την τιμή να μειώνεται αφού η τιμή ανά άτομο είναι πλέον η μίσση αν οι μέρες διαμονής είναι περισσότερες από τρεις.

🖱 Η κλάση `Hotel` έχει μέθοδο η οποία ορίζει το όνομα του ξενοδοχείου (`getHotelName`) καθώς και της μεθόδους που ζητήθηκαν (προσθήκη δωματίου : `addHotelRoom`, ανάκτηση δωματίου από κωδικό : `recoverRoomOutOfID`, ανάκτηση κράτησης από κωδικό : `recoverBookingOutOfID`, προσθήκη κράτησης σε δωμάτιο :

`addBookingToRoom`, προσθήκη κράτησης : `addBookingNoID` , ακύρωση κράτησης : `deleteBooking`, υπολογισμός εσόδων : `income`, πλάνο κρατήσεων : `bookingPlan`). Στην κλάση `hotel` έχουν δημιουργηθεί δύο `arraylist` ως μια δομή δεδομένων για να αποθηκευτούν τα δωμάτια και οι κρατήσεις. Με τον δημιουργό της κλάσης `Hotel` δημιουργούμε ένα αντικείμενο της κλάσης με όνομα που δίνεται ως όρισμα. Πιο συγκεκριμένα η μέθοδος `AddHotelRoom` προσθέτει τα αντικείμενα τύπου δωμάτιο και της υποκλάσης τους. Η μέθοδος `recoverRoomOutOfID` υπάρχει για την ανάκτηση δωματίου από το `array list` δίνοντας ως όρισμα το ID ενός δωματίου. Η `recoverBookingOutOfID` μέθοδος ομοίως κάνει ανάκτηση δωματίου από το `array list` δίνοντας ως όρισμα το ID μιας κράτησης. Με την μέθοδο `addBookingToRoom` ουσιαστικά δίνοντας ως όρισμα το ID από ένα δωμάτιο και μια κράτηση το πρόγραμμα ελέγχει άμα μπορεί να κάνει αυτήν την κράτηση. Είναι σημαντικό να τονίσουμε ότι αν ο χρήστης επιλέξει να μείνει πάνω από μια μέρα στο ξενοδοχείο και κάνει κράτηση δίνοντας το ID δωματίου που θέλει να μείνει τότε η κράτηση θα δημιουργηθεί άσχετος του αν υπάρχει δωμάτιο τύπου D, το οποίο και είναι δωμάτιο για μια μέρα. Ομοίως αν ο χρήστης κάνει κράτηση επιλέγοντας να μείνει περισσότερες

από τις ελάχιστες μέρες καθώς και τα άτομα είναι περισσότερα από το ελάχιστο άτομα που υπάρχουν ως παράμετροι στα δωμάτια τύπου C και διαλέξει δωμάτιο διαφορετικού του τύπου από το C η κράτηση θα δημιουργηθεί ανεξαρτήτως του εάν υπάρχουν διαθέσιμα δωμάτια τύπου C. Η `addBookingNoID` μέθοδος αντιστοιχίζει μια κράτηση σε ένα δωμάτιο αν ο χρήστης δεν επιλέξει συγκεκριμένο δωμάτιο. Αξίζει να σημειωθεί ότι το σύστημα αυτόματης κράτησης (δηλαδή η μέθοδος προσθήκης κράτησης σε δωμάτιο χωρίς ID) θέτει κάποιους περιορισμούς. Αν ο χρήστης επιλέξει να μείνει περισσότερες από τις ελάχιστες μέρες και τα άτομα είναι περισσότερα από τα ελάχιστα άτομα που υπάρχουν ως παράμετροι στο δωμάτιο τύπου C και δεν υπάρχει δωμάτιο τύπου C διαθέσιμο τότε η κράτηση δεν θα πραγματοποιηθεί. Αλλιώς αν υπάρχει δωμάτιο τύπου C τότε η κράτηση θα γίνει αναγκαστικά σε αυτό το δωμάτιο. Ομοίως εάν ο χρήστης διαλέξει να μείνει στο ξενοδοχείο για μια μέρα και δεν υπάρχει δωμάτιο τύπου D (που είναι δωμάτιο για κράτησης μιας μέρας) διαθέσιμο εκείνες τις ημερομηνίες τότε η κράτηση δεν θα γίνεται. Διαφορετικά αν υπάρχει δωμάτιο του D τότε η κράτηση θα γίνει αναγκαστικά σε δωμάτιο τύπου D. Η μέθοδος `deleteBooking` διαγραφεί μια κράτηση από την λίστα


κρατήσεων του ξενοδοχείου. Η `income(int ID)` μέθοδος εκτυπώνει τα έσοδα που έχει το ξενοδοχείο από το συγκεκριμένο δωμάτιο του οποίου το ID δόθηκε. Η `income` εκτυπώνει τα έσοδα που έχει το ξενοδοχείο από όλα τα δωμάτια συνολικά. Τέλος η μέθοδος `bookingPlan` εκτυπώνει τον πίνακα διαθεσιμότητας.

 Η κλάση `RunTime` έχει την `main` συνάρτηση καθώς και την μέθοδο `setCancellationChance` η οποία θέτει την πιθανότητα ακύρωσης (πιθανότηταΑκυρωσης = $(1/\text{cancelationChan}) * 100\%$). Έπειτα θέτουμε τιμές στις `static` μεταβλητές για εύκολη παραμετροποίηση του κώδικα. Ο πίνακας `names` περιέχει `Strings` με διάφορα ονόματα που θα χρησιμοποιηθούν για τυχαία ακύρωση. Δημιουργούμε τα αντικείμενα `dice` για τυχαία επιλογή ενός αριθμού, `choose` και `input` για είσοδο από το πληκτρολόγιο. Έπειτα δημιουργούμε και προσθέτουμε τα `room` στα δωμάτια του ξενοδοχείου. Μέσα στην `do-while` έχουμε την προσομοίωση. Κάθε `loop` της `do-while` είναι και μια προσομοίωση κρατήσεων. Στην συνέχεια με την `switch case` ο χρήστης επιλέγει τι θέλει να κάνει από τις παρακάτω επιλογές:

- Επόμενη Επανάληψη (το πρόγραμμα συνεχίζει την λειτουργία του)


- Προσθήκη Κράτησης (ο χρήστης δίνει: Όνομα, Άφιξη, μέρες, άτομα και προαιρετικά συγκεκριμένο κωδικό δωματίου)
- Ακύρωση Κράτησης (ο χρήστης δίνει τον κωδικό της κράτησης)
- Προβολή Κρατήσεων (εκτυπώνεται πίνακας με όλες τις κρατήσεις του Ξενοδοχείου): Κωδικός Κράτησης | Όνομα Πελάτη | Κωδικός Δωματίου
- Προβολή Δωματίων (εκτυπώνεται πίνακας με τα δωμάτια του Ξενοδοχείου) Κωδικός Δωματίου | Πληρότητα | Έσοδα - Προβολή Πλάνου Κρατήσεων (καλείται η αντίστοιχη μέθοδος του Ξενοδοχείου)
- Προβολή Εσόδων (ο χρήστης δίνει προαιρετικά συγκεκριμένο κωδικό δωματίου)
- Τερματισμός (το πρόγραμμα τερματίζει την λειτουργία του).

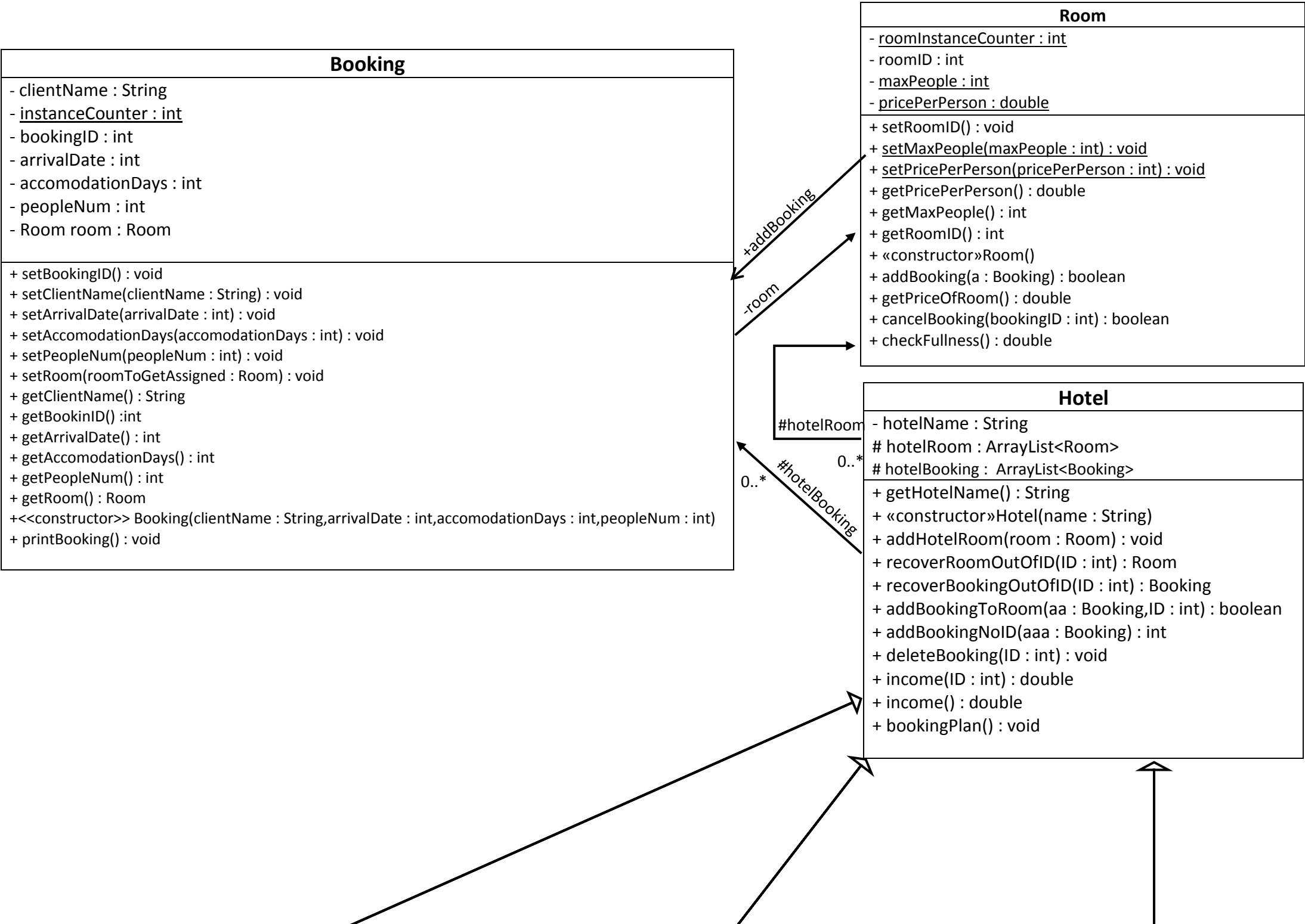
Στην αρχή του loop δημιουργούμε μια τυχαία κράτηση δίνοντας τυχαίες τιμές στις μεταβλητές του booking. Αμέσως μετά υπάρχει κώδικας για τυχαία ακύρωση κράτησης. Τέλος μετά την δημιουργία της τυχαίας κράτησης και της ακύρωσης τυχαίας κράτησης δίνεται η επιλογή στον χρήστη για ακύρωση κράτησης.

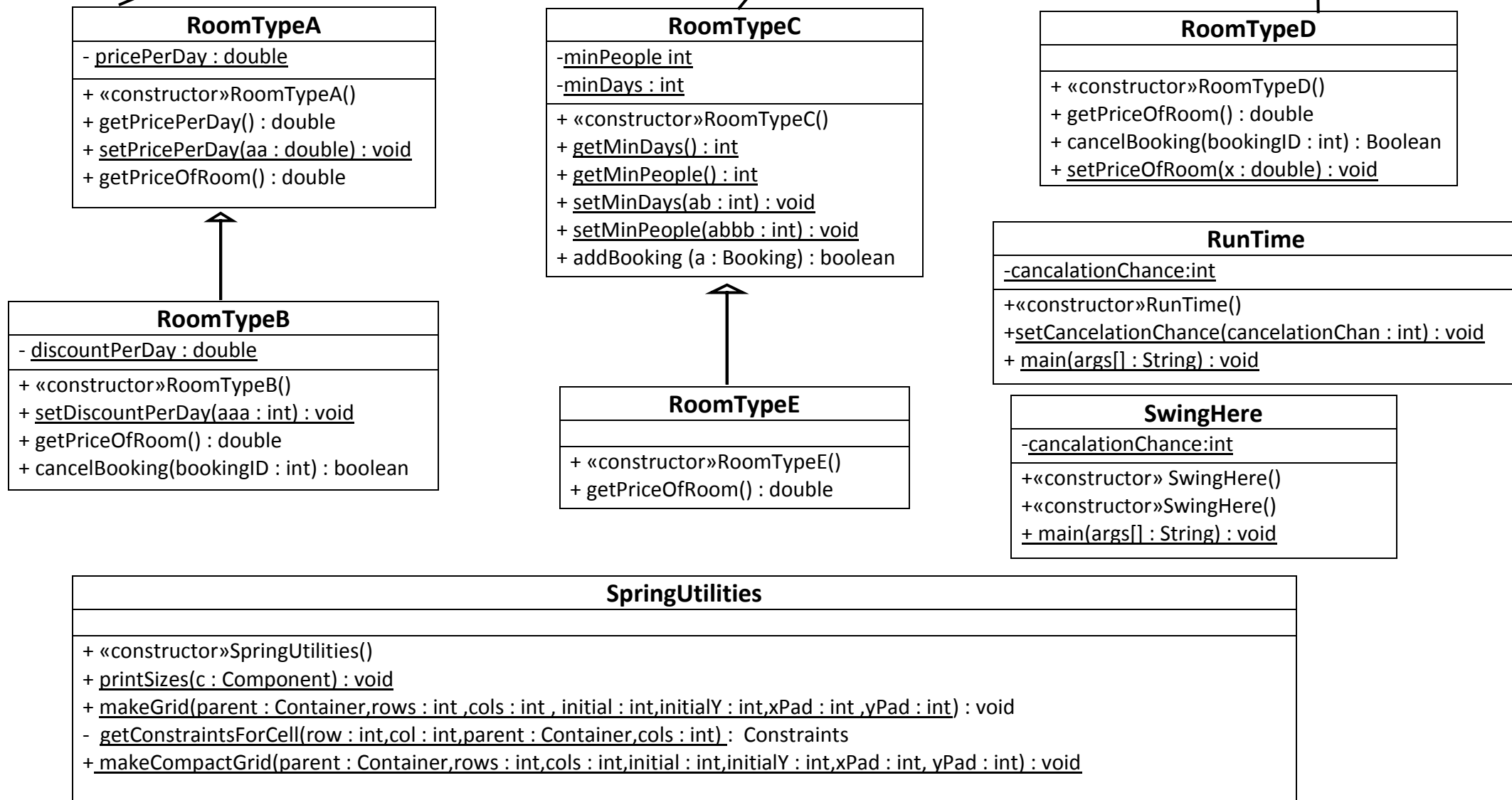
 Η κλάση [SpringUtilities](#) υπάρχει για να υποστηρίξει τη διάταξη σε κατασκευαστές GUI(Graphical User Interface) και συγκεκριμένα στο SpringLayout.

 Η κλάση [SwingHere](#) υπάρχει για παραθυρική εφαρμογή.

 Το πρόγραμμα έχει υλοποιηθεί και με την χρήση της βιβλιοθήκης Swing.

 Παρακάτω παρατίθεται και το UML διάγραμμα κλάσεων. Κάθε πίνακας περιέχει το όνομα της κλάσης, τις μεταβλητές και τις μεθόδους της. Κάθε βέλος δείχνει την σχέση μεταξύ των κλάσεων και των υποκλάσεων τους. Επίσης παρατίθεται ένα υπόμνημα που εξηγεί βασικά σύμβολα του διαγράμματος :





Υπόμνημα

+ : public
 - : private
 # : protected
 «constructor» : κατασκευαστής
Υπογραμμισμένες μεταβλητές : static