**2. modul**

Programming Basics

Java Basics

oop basics, programming basics, java basics, java core, junit, sql basics

# Előszó

A feladat megoldáshoz bármilyen segédanyagot használhatsz. A részfeladatokat rész pontozzuk, így nem probléma ha esetleg valamelyik részfeladatot kihagyod mert nem tudod, nyugodtan menj tovább.

A pajkaszegi wellness hotel megnyitotta kapuit. Többfajta szoba típusból lehet választani foglalás során (standard, superior és king), a szobák egységesen kétágyasak, de utólag pótágy elhelyezésérére is van lehetőség. Az árazás is időszakosan változik a szobáknak, melyet az alábbi táblázat tartalmaz *(az árak szoba árak, nem számít hány fő száll meg bennük)*:

| **Tavasz (mar - maj )** | **Nyár (jun-aug)** | **Ősz (szep-nov)** | **Tél (dec-febr)** |
| --- | --- | --- | --- |
| 11.000 Ft | 21.000 Ft | 15.000 Ft | 11.000 Ft |

A pótágy díja egységesen: 3.000 Ft/éj

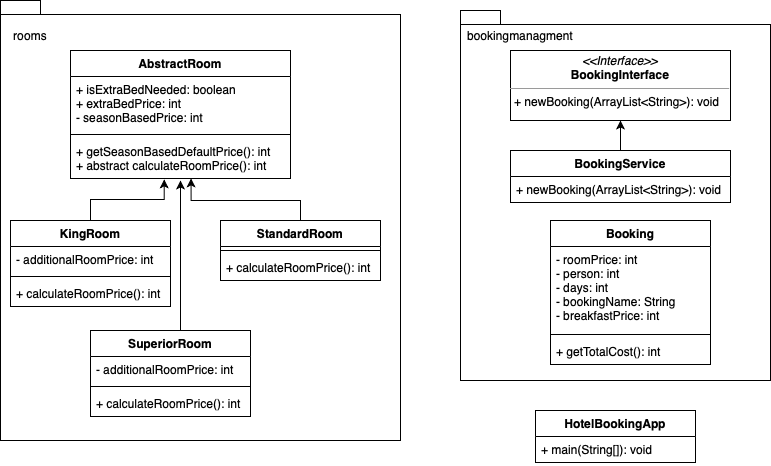
A superior szoba felára: 3.500 Ft

A king szoba felára: 11.000 Ft

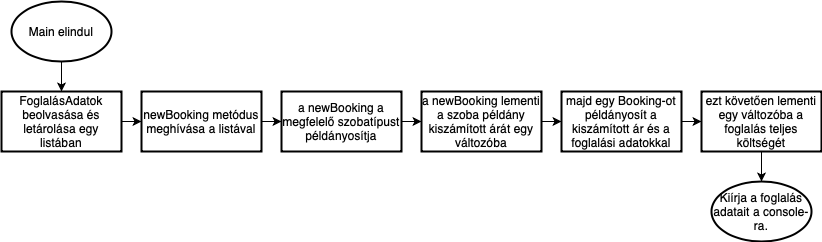
Reggeli opcionálisan kérhető: 4.500 Ft/fő *(de csak foglalásonként kérhető, nem lehet olyan hogy 2 fő foglal és csak egy valaki kér reggelit a szobában)*

A célja a gyakorlatnak, hogy egy szállásfoglaló konzolos alkalmazást készítsünk, melyben le tudjuk foglalni a pajkaszegi álomnyaralás wellness szállását. A foglaláshoz szükséges adatokat konzolról kérje el tőlünk a rendszer, majd írjon ki egy megerősítő üzenetet a foglalásról, melynek része a várható összköltség is. Előfoglalás nincs, csak a helyszínen lehetséges bejelentkezni és aktuális naptól lehet foglalni.

Az alábbi ábra mutatja a 2 package-et, amiben a szükséges osztályok vannak, valamint a main metódust tartalmazó osztályt:



Az alábbi ábra pedig a flow diagramot mutatja, hogy hogyan fut le a program:



# Feladatok

1. Létrehozni a console-os projektet, a megadott nevű main metódust tartalmazó osztállyal. Majd az ugyanilyen nevű private repository-ba való feltöltése, a vezetéknév, keresztnév hozzáfűzésével. Pl.: HotelBookingApp-ZsoltSzurovecz **(2 pont)**
2. Hozz létre 2 package-et az ábrának megfelelően (**1 pont**)
3. Készítsd el az ***AbstractRoom***osztályt
   1. az ábra szerinti attribútumokkal és abstract metódussal (**1 pont**)
   2. legyen egy ***getSeasonBasedDefaultPrice*** metódus, amely az aktuális hónapnak megfelelően automatikusan meghatározza a szoba alapárát és beállítja a ***seasonBasedPrice*** privát osztályváltozót. (**3 pont**)
4. Származtasd le a szükséges szobatípusokat, amelyikhez szükséges add hozzá a felárat tartalmazó ***additionalRoomPrice*** privát attribútumot. (**1 pont**)
   1. dolgozd ki a kötelezően kidolgozásra kerülő ***calculateRoomPrice*** metódust. Minden szükséges értéket a leszármazott és ősosztályban megtalálsz ami kellhet. (**1 pont**)
5. A main metódusban kérd el a felhasználótól a foglaláshoz szükséges adatokat és tárold le azokat egy String típusú listában:
   1. Hány főre foglal? Hány napot marad(nak)? Reggelit kér(nek)-e? Milyen néven rögzíthető a foglalás? Milyen szobát szeretne? Pótágyra szükség van? (**1 pont**)

Tehát egy példa lista tartalma: “3”,”2”,”Y”,”Szifon”,”King”,”Y”

1. Készítsd el a ***Booking*** osztályt
   1. az ábrán látható privát attribútumok (ahol “-” előtte szerepel) és publikus metódusával. Hozd létre a kontruktorát is, hogy be tudd állítani az attribútumok értékeit és a metódus is tudjon számolni velük. Minden attribútum példányosításkor kötelezően megadandó érték. (**2 pont**)
   2. Értelemszerűen készítsd el a metódus törzsét, hogy visszaadja az osztály változókkal történő számítás eredményét, a foglalásnak megfelelően (**1 pont**)
2. Hozd létre a ***BookingInterface*** nevű interfészt, melyben csak a fenti ábrán látható metódus szerepel. Készítsd el a ***BookingService*** osztály, amely implementálja a *BookingInterface*-t. (**1 pont**)
3. A main metódusban példányosítsd a *BookingService* osztályt és hívd meg a *newBooking* metódust. Paraméterként add át, a felhasználótól beolvasott adatokat tartalmazó listát.(**1 pont**)
4. A ***newBooking***metódus hivatott arra, hogy a paraméterként kapott felhasználótól beolvasott adatokat tartalmazó listát feldolgozza, és példányosítva egy *Booking*példányt, létrehozza a foglalást.
   1. Készíts egy a felhasználói adatnak megfelelő szobatípus példányt. (**2 pont**)
   2. Tárold le a létrehozott példánytól elkért kiszámított szoba árát egy változóba (**1 pont**)
   3. Példányosíts egyet a *Booking*-ból a megfelelő foglalási adatokkal (**2 pont**)
   4. Mentsd le egy változóba a *Booking* *getTotalCost* metódusától visszakapott foglalás értékét egy változóba (**1 pont**)
   5. Írd a foglalás minden adatát és a foglalás árát a konzolra (**2 pont**)
5. Töltsd fel munkádat és ellenőrzid le, hogy a legfrissebb változat van a remote repositoryban (**1 pont**)
6. Clean code - szubjektív pontok (a kód jól strukturált, jó elnevezések és megvalósítások) (**2 pont**)

# Extra feladatok

Az alábbiak közül csak akkor válassz, ha már a fentiekkel elkészültél, leokéztuk és van még időd, kedved kódolni:

* A fogalás adatait lementi egy fájlba is, melynek neve az foglalási adatok - aktuális nap.txt. Pl.: foglalási adatok-2022.01.14.txt)
* Külön osztályban tároljuk a felhasználótól beolvasásra kerülő adatokat, és azokat a beolvasás sorrendjében, az osztály setter-eivel állítjuk be, mentjük le. Pl.: UserInput osztály esetén, userinput.setExtraBedNeeded(true). Az adatok a logikus adattípusban tárolódnak, int, boolean, String és a szoba típusa a neki készített Enum. Ezt a létrehozott objektumot adjuk tovább a newBooking metódusnak.
* A program futása ne fejeződjön be egy foglalásnál, mindaddig lehessen foglalást rögzíteni, amíg új foglalást kér a felhasználó.
* Véletlenszerűen változzon, hogy melyik szobából mennyi van szabad és ne lehessen olyan szobát foglalni amiből nincs elég elérhető.
* Annyi szobát foglaljunk, amennyi alapágyas szoba szükséges hogy megadott vendégeket elszállásoljuk. Pl.: 5 főre foglalunk Superior szobát, akkor az 3 Superior szoba. Vagy ha a felhasználó pótágyat szeretne, akkor 2 Superior szoba és az egyik pótágyas.