HF TAPASZTALATOK + ZH TUDNIVALÓK

A HF-EK APROPÓJÁN

Néhány javaslat:

1. Mielőtt nekikezd kódolni, **OLVASSA EL** rendesen a feladat szövegét – akár többször is, alá is lehet húzni.
2. Ne írjon kilométeres sorokat.
3. Lehetőleg ne használjon ilyen változóneveket:
   o, O, l, I
   (kis O, nagy O, kis L, nagy I)

Olvassa el rendesen a szöveget, és csak az kerüljön be a kódba, ami a szövegben is szerepel.

Szövegértés!

A HF-EK APROPÓJÁN

Végig kell gondolni, hogy mi változó, mi metódus, mit hol kell kiszámolni (meghatározni). A vezérlő osztály feladata lehetőleg csak a vezérlés.

Alap-osztályokban NINCS input/output $\Rightarrow$ ha valahogy tudatni kell a hívó osztállyal, hogy eredményes-e a hívás vagy sem, akkor void helyett pl. boolean metódust írunk.

Hibakeresés:

Olvassa el a hibaüzenetet!!

Tesztelje a futást!

ex.printStackTrace()

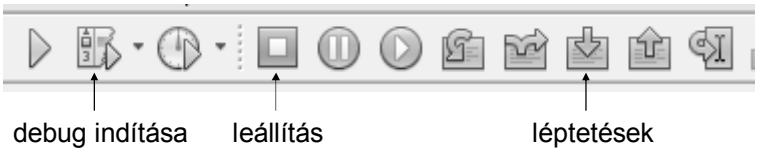
A HF-EK APROPÓJÁN

Tesztelés:

1. Lehetnek teszt célú **kiíratások**. de inkább debug, pl:

```
27      public void rendel(Ital ital){
            int index = italok.indexOf(ital);
```

Berakhatunk break pointo(ka)t, ezeket figyelhetjük.



debug indítása leállítás léptetések

http://www.cs.uga.edu/~shoulami/sp2009/cs1301/tutorial/NetBeansDebuggerTutorial/NetBeansDebuggerTutorial.htm

A HF-EK APROPÓJÁN

Osztálynév kiíratása:
vagy this.getClass().getSimpleName().toLowerCase(),
vagy kell egy String getOsztalyNev() metódus.

Lehet-e két toString()?

Nem, de

a/ lehet mellette más String eredmenyString() metódus;

b/ lehet egy feltételtől függő elágazás a toString()-en belül.

HF TAPASZTALATOK: EGYEDI SORSZÁM

Pl. Ember példányok egyedi sorszámozása

Sokan így oldották meg: Az Ember osztályban vagy a konstruktorban adták meg, vagy set() metódussal.

Probléma: nem egyedi, hiszen a sorszám értéke a példányosítótól vagy a példány használójától függ.

(Ez persze nem jelenti azt, hogy ne adhatnánk kívülről generált, egymástól eltérő sorszámokat, csak ilyenkor lehetne egyformákat is adni.)

EGYEDI SORSZÁM

Megoldás: Az Ember osztályban bevezetünk egy minden emberre jellemző statikus változót – ez lesz az utolsó ember sorszáma. Az új ember a rákövetkező sorszámot kapja:

```
private int sorszam;
private static int utolsoSorszam;

public Ember( ...  ) {
        ...
    utolsoSorszam ++;
    this.sorszam = utolsoSorszam;
}
```

HF TAPASZTALATOK

– mi kerüljön a konstruktor paraméterlistájába?

– mihez kell set metódus, mihez nem, mikor „tilos” a set...

– statikus / nem statikus változók, metódusok;
hol adjuk meg a statikus változó értékét?

A vezérlő osztályban!

– input/output helye: SOHA nem az alaposztály

– Az egyik alaposztályban:
if (Global.RENDEZES_ELOTT == true) mi a baj?

- nem szabad az alaposztályból hivatkozni a globalra.

HF TAPASZTALATOK

Alaposztályban kiíratás helyett:

a/ toString() más String metódus is lehet

b/ bármilyen visszatérési értékkel rendelkező metódus értéke kiíratható a vezérlő osztályban

c/ hibaüzenet? pl. boolean típusú metódus

vagy void helyett int – és az egyes értékek jelzik a hibát, stb.

HF TAPASZTALATOK

Amennyire lehet, kerüljük a kódismétlést!

De hogyan?

– annak végiggondolásával, hogy mi kerüljön az ősosztályba, ill. általában az osztályba

– annak végiggondolásával, hogy mit lehet többször felhasználható metódusba írni

Ha az utód osztály metódusa ugyanazt csinálja, mint az ős, akkor TILOS megírni.

HF TAPASZTALATOK

Metódusok felülírása:
Előbb mindig gondoljuk végig, hogy a super használható-e, és milyen mértékben. NE számolja újra a képleteket, hanem hivatkozzon az ősre. (pl. lakóingatlan adó)

A metódusok lehetőleg legyenek függetlenek egymástól, vagyis hacsak nem valami részfeladat megoldására vonatkozik, akkor kerüljük azt, hogy az egyik metódus hívja a másikat.
Pl. ne a beolvasásból hívjuk a kiíratást.

Egy metódus mindig csak egyetlen dolgot csináljon

HF TAPASZTALATOK

Néhány észrevétel a rendezésről, max/min keresésről

Jó-e?

```
Collections.sort(sportolok,new KorSzerint());
legfiatalabb: sportolok.get(0).getNev()
```

Ha nem akar saját max/min metódust, akkor használjon beépített max/min-t, de ne a rendezés segítségével keresse!

Figyeljen a holtversenyre!

HF TAPASZTALATOK

Még néhány szó a keresésről

Adott: List<Diak> diakok = new ArrayList<>();

Hogyan vizsgáljuk meg, hogy szerepel-e már a diakok között egy adott kódú ember?

Attól függ…

a/ létrehozhatunk egy Diak diak = new Diak(kod) példányt és contains() vagy indexOf()

b/ Saját ciklusban végigmegyünk a diakok listán és if(diak.getKod.equals(kod)) $\Rightarrow$ talált + break.

Hacsak lehet, jobb a beépített (minden metódus esetén).

HF TAPASZTALATOK

Általában hol generáljuk a véletlen értéket?

NE az alaposztályban!

Alaposztályban SZINTE SOHA nincs random generálás. Ez csaknem mindig a beolvasást helyettesíti, márpedig annak a vezérlésben van helye.

ÖMLESZTETT ÉSZREVÉTELEK

Például: Véletlen példány megkeresése, vagy olyan jellegű feladat, amely azt kéri, hogy az elemek X%-ával történjen valami.

1. véletlen példány: véletlen index 0 és lista.size() között, majd lista.get(index)….
2. Ciklus a listaelemekre, és if(Math.random() < százalék) akkor…

ÖMLESZTETT ÉSZREVÉTELEK

```
[private] List<ŐsOsztaly> lista = new ArrayList<>();
for(ŐsOsztaly elem: lista){
        if(elem instanceof GyerekOsztaly){
                ((GyerekOsztaly) elem).metodus(…);
        }
        else elem.metodus(…);
}
```

a „nem példánya”: !(elem instanceof GyerekOsztaly)

TERMÉSZETESEN nem ez az egyetlen lehetőség!!!

ÉS nem biztos, hogy mindig „ömlesztett” lista kell.

GRAFIKUS FELÜLET

TILOS a konstruktorból indítani a beolvasást, és általában bármilyen végrehajtó metódust!

Javaslat: A panelt akkor húzza rá a frame-re, amikor még nincs rajta saját kód, de vannak már komponensek.

Vagyis érdemes a felület létrehozásával kezdeni a projekt felépítését.

A frame mindig BorderLayout.

GRAFIKUS FELÜLET

Mikor használjunk modellt, és mikor List példányt?

Modellt csak akkor, ha JList felületen akarjuk megjelentetni az objektumokat.

Hol rendeljük egymáshoz a JList-et és a modellt?

Ha már létezik a modell példány, akkor célszerű már a konstruktorban, vagy amikor beolvastuk, de mindenképpen CSAK EGYSZER rendeljük egymáshoz őket.

Alaposztályban csak List lehet, hiszen nem tudja, hogy milyen egyéb projektekben használják. Mit kezd egy modellel egy konzolos alkalmazás?

GRAFIKUS FELÜLET

JList elemének kiválasztása

valueChanged vs mouseClicked ?

Legyen értelmes hibaüzenet.

(Esetleg JOptionPane.showMessageDialog(…))

Ne legyenek fölösleges importok, üres esemény-metódusok

Véletlenül generált esemény leszedése:

komponens – properties – events fül

GRAFIKUS FELÜLET

Ez mitől lehet?

Note: Some input files use unchecked or unsafe operations.
Note: Recompile with -Xlint:unchecked for details.

ÖMLESZTETT ÉSZREVÉTELEK

Érdemes ellenőrizni az adatfájlban lévő UTF-8-as pöttyöt. Beolvasáskor meg kell adni a kódolási típust! (char_set), és „src”-ből olvasunk!!

• Beolvasáskor kötelező a try-catch, a try fejében nyissuk meg, amit meg kell nyitni. Ekkor NEM kell még + close.

• Adattag private! Metódus lehet protected.

• Lista gettere másolatot adjon vissza

• Átlagszámításkor figyeljünk a nullával való osztásra.

ÖMLESZTETT ÉSZREVÉTELEK

• Listába vagy modellbe olvassunk? Mikor olvashatunk modellbe?

• Több olyan feladat is volt, hogy rendezés után megváltozik a listaszövegek külalakja.
Ekkor nem új modell kell, hanem a régihez kell "másik" toString(). Vagyis kellene egy (statikus) logikai változó, és ennek értékétől függően vagy ilyen, vagy olyan a toString(). (vagy ha kettőnél többféle variáció lehet, akkor nem logikai változó, hanem egy int vagy enum.)

Feladattól függ, hogy ez a változó statikus-e vagy sem.

ÖMLESZTETT ÉSZREVÉTELEK

• Az redundancia, ha az adatokat listában is és modellben is tároljuk.

• Olvasás külön osztállyal, vagy nem? Ha igen, hogyan?

• Felületfrissítés: updateUI vagy repaint?

• FONTOS: lista valuechanged metódus – kiválasztott elem: mindig ellenőrizni kell, hogy ez null érték-e vagy sem.

• A panelméretre számolja rá a kereteket.

• Funkcionális kifejezések!

ÖMLESZTETT ÉSZREVÉTELEK

• Elterjedt ☹ ez a „megoldás”: két fájl beolvasásakor csak egy adatfájlt adnak meg példányosításkor. Ez miért rossz?

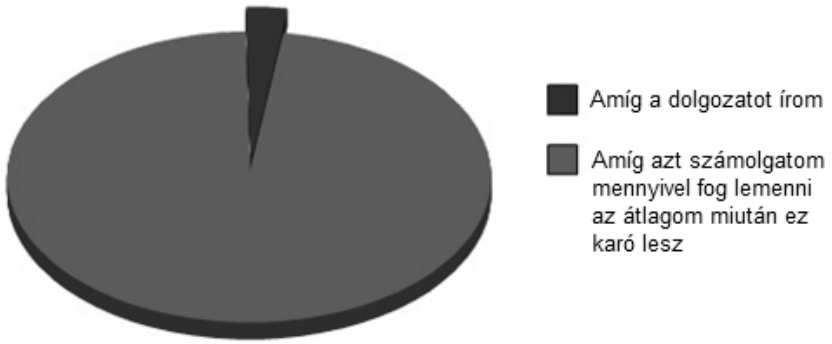
• Ne hagyja csíkként vagy pöttyként a labelt, lehet, hogy nem fog látszódni. És a listafelületen se maradjanak „item”-ek.

• Figyelje az adatfájlt, de ne szolgaian másolja az alapján a konstruktort.

• **alt shift f !!!!**

NEHOGY ÍGY JÁRJON ☺

DOLGOZATÍRÁS KÖZBEN



HANEM HAJRÁ!!!! ☺