Asztali alkalmazások fejlesztése - Java - 11. óra

November 10

2021

Generikus típusok

Feladatlap

Tartalom

Generikus típus használatának előnyei

A generikusok használatával **típus paramétereket** adhatunk meg osztályokhoz, interface-ekhez, illetve metódusokhoz. Az előny amelyet behoznak az, hogy erős típus ellenőrzéseket tud végezni a fordító, így már ő tudja jelezni, ha a típusokkal valami nincs rendben. Ezen felül rengeteg castolástól szabadít meg minket, mely igencsak rontja a kód olvashatóságát. A generikusok használatával próbálunk minél több futási idejű hibát fordítás közben detektálhatóvá, így könnyeben javíthatóvá tenni a kódunkat.

List osztály	**Generikus List osztály**

`List list = new ArrayList();`
`list.add("szoveg");`
`String s = (String) list.get(0);`
	`List<String> list = new ArrayList<>();`
	`list.add("szoveg");`
	`String s = list.get(0);`

	Mivel a List-nek megmondtuk a '<>' jelek között, hogy String-eket akarunk tárolni, ezért tudja, hogy amikor a 0. elemet kérem el akkor az egy String típusú objektum lesz. Továbbá a hozzáadáskor nem is engedi, hogy String helyett valami más elemet rakjak bele.

A generikusok használatával általános érvényű algoritmusokat is implementálhatunk, melyek különböző típusú elemekre egyaránt működnek, nem kell az algoritmust mindre megírni. Például: Collections.sort().

Generikus típus

Egy generikus típus olyan osztály vagy interface, mely generikus típusparaméterrel rendelkezik.

Egyszerű doboz osztály generikusok nélkül	Doboz osztály generikusokkal

`public class Box {`
`  private Object object;`
 	
`  public void set(Object object)`
`  { this.object = object; }`
 	
`  public Object get()`
`  { return this.object; }`
`}`
	`public class Box<T> {`
	`  private T t;`
	
	`  public void set(T object)`
	`  { this.t = t; }`
	
	`  public T get()`
	`  { return this.t; }`
	`}`

A doboz bármilyen objektumot képes eltárolni. Ez menet közben hibákhoz vezethet, ha valamilyen feltételezésekkel élünk arról, hogy milyen típusú elem van éppen a dobozban.	*Itt jelenik meg először a T típus paraméter. Ezen a ponton mondjuk azt, hogy a Box osztály vár egy típus paramétert is. A T típus paraméter ezek után bárhol használható az osztályon belül.*
	A fenti generikus osztály használata
	`Box<Integer> integerBox = new Box<>();`

A típus paramétereket általában egy nagy betűvel szoktuk jelölni.

Generikus Típus betűjel	Jelentés
T	Type – Típus
K	Key – Kulcs
E	Element – Elem
V	Value – Érték
N	Number - Szám

Diamond operátor

A Java 7-es verziójában megjelent a diamond operátor <>, melynek köszönhetően nem kell kiírni a típus paramétert a konstruktor hívásba, ha a típus paraméter a környezet alapján kikövetkeztethető.

Ennek köszönhető, hogy a generikus típusok példányosítása egyszerűsödött:

```
Box<Integer> integerBox =
new Box<Integer>();
```

helyett

```
Box<Integer> integerBox =
new Box<>();
```

Típus paraméter megszorítások (Bound Type Parameters)

Van amikor korlátozni akarjuk a típus paraméterek lehetséges értékeit. Például lehet, hogy egy függvény csak számokon dolgozik, ezért le akarjuk korlátozni, hogy csak Number-ből származó oszályt adhassunk át. Ez egy felső korlátot határoz meg. Ennek megadásához az extends kulcsszót kell használnunk! Típus paraméter megszorításnál az extends általánosan használandó class-ra és interface-re is (interface-re nem az implements-et kell használni).

```
public <T extends Number> void print(List<T> list){
    list.forEach(item -> System.out.println( item ) );
}

public <T extends Number> void copy(List<T> src, List<T> dest){
    ...
}
```

Feladatok

1. Feladat - Halmaz

A halmazok nagymértékben megkönnyítik minden olyan feladat megoldását, amelyben az ismétlődő elemeket csak egyszer vehetjük figyelembe. Írjunk generikus osztályt egy Set (Halmaz) reprezentálására!

A megvalósítandó műveletek:

- konstruktor
- add - hozzáad egy elemet a halmazhoz.
- remove - eltávolítja az elemet.
- getSize - visszaadja a halmaz méretet
- clear – kiüríti a halmazt
- isElement - megadja, hogy egy elem benne van-e a halmazban
- isEmpty – üres-e a halmaz
- isEquals - összehasonlít két halmazt
- isPartOf – része-e az egyik halmaz a másiknak
- unio - az uniót számolja ki
- intersection - kiszámolja a halmazok metszetét
- different - kiszámolja a különbséget, Az A és B halmaz (ebben a sorrendben tekintett) különbségének nevezzük azoknak az elemeknek a halmazát, amelyek elemei az A halmaznak és nem elemei a B halmaznak.
- toString – kiírja a halmaz tartalmát

2. Feladat – Halmaz alkalmazása – Szobakerékpár

A családi szobabiciklit a férj és a feleség beosztott időrend szerint használják. A feleség a hónap 1., 3., 4., 5., 7., 8., 9., 10., 14., 17., 18., 19., 20., 24., és 27. napján használja. A férj a hónap 7., 13., 14., 15., 20., 21., 23., 24., 25., 29., és 30. napjain szoba biciklizik. Egy nem szökőév átlagos hónaphosszát figyelembevéve a foglalt napok számát tekintve hány százalékos a szoba kerékpár kihasználtsága?

3. Feladat – Halmaz alkalmazása – Busz átszálás

Adott a budapesti 105 és 102 autóbusz megállói. Határozd meg, hogy hány olyan megálló van, ahol az egyikről átszállhatunk a másikra? Melyek ezek a megállók?

105-ös busz megállói	**102-es busz megállói**
Apor Vilmos tér Kiss János altábornagy utca Nagy Jenő utca Márvány utca Királyhágó utca Győri út Ág utca Krisztina tér Clark Ádám tér Széchenyi István tér	Széll Kálmán tér Maros utca Maros utcai rendelőintézet Déli pályaudvar Kék Golyó utca Királyhágó tér Kiss János altábornagy utca Apor Vilmos tér Németvölgyi út.

József nádor tér Deák Ferenc tér Bajcsy-Zsilinszky út Opera Oktogon Vörösmarty utca Kodály körönd Bajza utca Hősök tere Vágány utca Lehel utca Hun utca Lehel utca – Róbert Károly krt. Béke tér Frangepán utca Fiastyúk utca Násznagy utca József Attila tér Cziffra György park Gyöngyösi utca.	

További budapesti busz menetrendeket találhat a BKK futár oldalán. Ha érdekel, akkor határozd meg, hogy hány közös megállója van az 5-ös illetve a 7-es buszoknak!?

4. Feladat – Halmaz alkalmazása – Lottó - 2005 májusi érettségi

Magyarországon 1957 óta lehet ötös lottót játszani. A játék lényege a következő: a lottószelvényeken 90 szám közül 5 számot kell a fogadónak megjelölnie. Ha ezek közül 2 vagy annál több megegyezik a kisorsolt számokkal, akkor nyer. Az évek során egyre többen hódoltak ennek a szerencsejátéknak és a nyeremények is egyre nőttek.

Adottak a lottosz.dat szöveges állományban a 2003. év 51 hetének ötös lottó számai. Az első sorában az első héten húzott számok vannak, szóközzel elválasztva, a második sorban a második hét lottószámai vannak stb.

Bemenet lotto.dat
37 42 44 61 62 18 42 54 83 89 ... 9 20 21 59 68

A lottosz.dat állományból beolvasott adatok alapján döntse el, hogy volt-e olyan szám, amit egyszer sem húztak ki az 51 hét alatt! A döntés eredményét (Van/Nincs) írja ki a képernyőre!

5. Feladat – Halmaz alkalmazása - Órarend

Egy iskola tanárairól tudjuk, hogy mikor milyen órát tartanak. A tanárokat, a tantárgyakat, a hét napjait, a napokon belüli órákat sorszámukkal azonosítjuk. Készíts programot, amely megadja:
- **A** részfeladat: minden napra az aznap órát tartó tanárok számát;
- **B** részfeladat: azt a tantárgyat, amit a legtöbb tanár tanít;
- **C** részfeladat: az adott T tanárt egész héten helyettesíteni tudó tanárt.

A bemenet első sorába olvassad be a tanárok számát (1 ≤ N ≤ 100), a tantárgyak számát (1 ≤ M ≤ 100) és egy tanár sorszámát (1 ≤ T ≤ N), egy-egy szóközzel elválasztva. A következő sorok mindegyikében 4 egész szám van, egy-egy szóközzel elválasztva: tanár sorszám, tanított tantárgy sorszáma, nap (1 és 5 közötti egész szám), óra (0 és 8 közötti egész szám). Például 3 7 2 0 azt jelenti, hogy a harmadik tanár a hetedik tantárgyat a hét második napján a nulladik órában tanítja.

Az kimenetre négy sort kell írni! Az első sorba az A, a másodikba a B, a harmadikba C részfeladat eredményét. Ha több megoldás van, akkor az elsőt kell kiírni. Ha nincs megoldás (C részfeladatban), akkor -1-et kell kiírni! Az első sorban 5 szám szerepeljen, egy-egy szóközzel elválasztva!

Bemenet	**Kimenet**
3 4 1 **1 1 1 6** 1 1 2 2 1 2 1 3 2 1 2 2 2 2 3 1 **3 4 1 2** 3 2 1 4 3 3 2 1	2 3 1 0 0 2 3

A példában szereplő 3 tanár órarendje

1. tanár	1. nap	2. nap	3. nap	**2. tanár**	1. nap	2. nap	3. nap	**3. tanár**	1. nap	2. nap	3. nap
0. óra				0. óra			T2	0. óra			
1. óra				1. óra				1. óra		T3	
2. óra		T1		2. óra		T1		2. óra	**T4**		
3. óra	T2			3. óra				3. óra			
4. óra				4. óra				4. óra	T2		
5. óra				5. óra				5. óra			
6. óra	**T1**			6. óra				6. óra			

Ötlet a megvalósításhoz:

- Állítsuk elő minden napra az aznap órát tartó tanárok halmazát (NAP) – a megoldás e halmazok elemszáma!
- Állítsuk elő minden tárgyra az azt tanító tanárok halmazát (TÁRGY)– a megoldás a legnagyobb elemszámú halmaz elemszáma!
- Állítsuk elő minden tanárra az órái halmazát (ÓRÁK) – a megoldás egy olyan halmaz sor-száma, aminek a T tanár halmazával nincs közös eleme – azaz soha nincs egyszerre órájuk!

6. Feladat – Verem

A verem adatszerkezet egy speciális szekvenciális tároló, amelyből mindig a legutoljára betett elemet vehetjük ki legelőször. Emiatt szokás a vermet LIFO (Last In – First Out) szerkezetnek nevezni.

Írjunk generikus osztályt egy verem (Stack) reprezentálására! Belül használhatunk akár List-et is.

A verem megengedett műveletei:

- inicializálás (konstruktor): alapállapotba helyezés, ürítés.
- verembe (push): egy elem betétele a verembe (a verem „tetejére”).
- veremből (pop): a verem „tetején” található elem kivétele a veremből. Csak akkor tudunk elemet kivenni, ha a verem nem üres.
- felső (top): a verem „tetején” található elem lekérdezése.
- üresE (isEmpty): igaz, ha a veremben egyetlen elemet sem tárolunk.
- ürít (empty): minden elemet eltávolít a veremből

Néhány példa a verem adatszerkezet alkalmazására:

- **Programozási nyelvekben az egymásba ágyazott eljárások, illetve függvények, valamint a rekurzió megvalósítása.** Az eljárások, függvények lokális változói, érték szerint átvett paraméterei a rendszer veremben kerülnek létrehozásra. Az eljárás végrehajtása után kikerülnek a veremből, azaz megszűnnek. Beágyazott eljárás, függvény hívásakor a veremre kerül a visszatérési cím is, vagyis az, hogy melyik sornál kell majd folytatni a programot, ha a beágyazott eljárás végrehajtásra került.
- **Zárójelezett kifejezés** / (), [], {} / helyességének az ellenőrzése. Bal zárójel a verembe kerül, jobb zárójel esetén kivesszük a verem tetején lévő bal zárójelet, és összehasonlítjuk az aktuális jobb zárójellel. Ha párt alkotnak, nem romlott el a zárójelezés, tovább vizsgáljuk. Zárójelezési hibák: nem megfelelő pár; túl sok bal zárójel (a vizsgálat végén marad elem a veremben); túl sok jobb zárójel (üres veremből akarunk kivenni). Hasonlóan ellenőrizhető Pascal programban a begin/end; repeat/until; case/end párok helyessége.
- **Lengyel forma**. Lukasewich lengyel matematikus az 50-es években a matematikai formulák olyanfajta átalakítását dolgozta ki, amelynek segítségével a fordítóprogram könnyen ki tudja számítani a kifejezés értékét. Erre azért volt szükség, mert az ember által megszokott „infix” és zárójeles írásmód nem látszott alkalmas struktúrának a kiértékelés céljára. A bevezetett új ábrázolási formát a szerző tiszteletére lengyelformának is nevezik. Másik elnevezés a posztfix forma.
- **Undo (mégsem) funkció** megvalósítása az alkalmazói programokban. Kissé módosított verem, ugyanis a túlságosan régen végrehajtott műveleteket nem tároljuk. A végrehajtott műveletek kódja (esetleg néhány jellemzője) a verembe kerül, a mégsem végrehajtásakor a legutolsó művelet kódját kivesszük, és visszavonjuk a műveletet.

7. Feladat – Verem alkalmazása – Tükörszó

Írja meg a verem típushoz rendelt műveletek segítségével az alábbi feladatot megoldó programot! Olvasson be egy S szöveget, majd mondja meg, hogy **tartalmaz-e a szöveg tükörszót**? (a tükörszó egy legalább 2 hosszúságú, oda és visszafelé olvasva ugyanaz a szöveg). Amennyiben igen, add meg a tükörszót és azt, hogy az S hányadik karakterénél kezdődik!

Bemenet	**Kimenet**
S=”ALMA”	hamis
S=”ALLAH”	igaz, a szó: ALLA, az 1. pozíciónál kezdődik
S=”ATTI”	igaz, a szó: TT, a 2. pozíciónál kezdődik
S=”ALATT”	igaz, a szó: ALA, az 1. pozíciónál kezdődik

S=”HARCRA”	igaz, a szó: ARCRA, a 2. pozíciónál kezdődik

8. Feladat – Verem alkalmazása – Számrendszer átváltó

Írjon programot, amely a **10-es számrendszerből más számrendszerbe** vált át egy pozitív egész számot! Ehhez a számot **maradékosan osztjuk az új alapszámmal** mindaddig, **amíg a hányados 0 nem lesz**. Ekkor a maradékok fordított sorrendben adják meg az új számrendszerben felírt értéket. A fordított sorrendet egy verem segítségével állítsa vissza a szokásos, balról jobbra csökkenő helyi érték szerinti elrendezésbe.

Gondoskodjon arról, hogy a 10-nél nagyobb alap esetén az ábécé betűi jelenjenek meg a számjegyek helyett (A=10, B=11 stb.)!

$89_{10} = 324_5$	$89_{10} = 1011001_2$
89:5 =17 maradt a 4 17:5=3 maradt a 2 3:5=0 maradt a 3	89:2=44 maradt az 1 44:2=22 maradt a 0 22:2 = 11 maradt a 0 11:2 = 5 maradt az 1 5:2 = 2 maradt 1 2:2 = 1 maradt 0 1:2 = 0 maradt 1

9. Feladat – Verem alkalmazása – Helyesen zárójelezett-e

Ellenőrizze egy **összetett kifejezés zárójelezésének helyességét**, ha **kerek, szögletes, kapcsos** zárójelek mindegyike előfordulhat! **(),[],{}**

A zárójel-helyesség ellenőrzésének elvei:

- Csukó zárójel nem lehet megelőző nyitózárójel nélkül.
- A kifejezés vizsgálatának végén nem maradhat bezáratlan nyitózárójel.
- Nem lehet keresztzárójelezés, azaz egy fajta nyitózárójel után nem következhet másfajta csukó zárójel.

Ennek megfelelően célszerű a kifejezés elemzése során a **nyitózárójeleket verembe tenni**, **csukó zárójel esetén az utolsó nyitózárójelet kivenni**. Hibás a kifejezés, ha bármely csukó zárójel esetén a **verem tetején nem neki megfelelő nyitózárójel** van, vagy üres a verem, ill. a kifejezés levizsgálása után nem üres a verem.

például: S=”[(3+(5]-7)*6)” - *helytelen*

10. Feladat – Verem alkalmazása – Kifejezés kiértékelése

Adott egy kifejezés, teljes zárójelezéssel. pl: ((3 + (4 - 12)) / (5 - (2 * 1)))

A teljes zárójelezés azt jelenti, hogy minden művelthez tartozik egy nyitó és egy csukó zárójel: (a - b). Az egyszerűség kedvéért a további megszorításokat is feltételezzük:

- minden token (zárójel, műveleti jel, szám) szóközökkel van egymástól elválasztva
- csak az egészeken értelmezett négy alapművelet szerepelhet, az osztás "egész osztás" (DIV)
- a kifejezés helyes

Írjunk programot, ami kiszámolja a kifejezés értékét.

bemenet.txt tartalma	kimenet.txt tartalma
(1 + 0) (2 + ((3 + 4) * (5 * 6))) ((2 * 2) / (2 + 3)) ((3 + (4 - 12)) / (5 - (2 * 1))) (3 / (4 / 5))	1 212 0 -1 HIBA

Házi feladat

1. Feladat - Multihalmaz

Írjunk generikus osztályt egy Multihalmaz (MultiSet, Bag) reprezentálására! A multihalmazok és a halmazok között annyi a különbség, hogy a multihalmaz egy elemet többször is tartalmazhat. Egy elem multiplicitása azt adja meg, hogy az elem hányszor van benne a halmazban.

A Multi halmaz műveletei azonosak a halmazéval, azonban figyeljen a megvalósítás során az alábbi műveletekre:

- remove – csökkenti az adott elem multiplicitását, ha az nullára csökkenne, eltávolítja az elemet.
- unio - az unióban az egyes elemek multiplicitásai az elem eredeti halmazokban lévő multiplicitásának összege
- intersection - kiszámolja a multihalmazok metszetét: itt az azonos, de többször szereplő elemeket különbözőnek kell tekinteni a halmazhoz képest
- isEqual - összehasonlít két halmazt: itt is az azonos, de többször szereplő elemeket különbözőnek kell tekinteni az összehasonlításkor
- different - kiszámolja a különbséget (itt a multiplicitások is kivonódnak, ha negatív, akkor az elem nem szerepel a különbséghalmazban)

2. Feladat – Multihalmaz alkalmazása

Egy almatermelő N (1≤N≤100) fajta almát termel, ismerjük, hogy melyik fajtából mennyit. Egy kereskedő M (1≤N≤100) fajta almát szeretne venni tőle, azt is tudjuk, hogy melyik fajtából mennyit.

Készíts programot, amely megadja, hogy

- (A) a termelőnek melyik fajtából mennyi marad!
- (B) a kereskedő melyik fajtából mennyit tud vásárolni!

Bemenet	
Termelő fajtái száma: 3 1. fajta neve: jonagold 1. fajta mennyisége: 100 2. fajta neve: golden 2. fajta mennyisége: 30 3. fajta neve: idared 3. fajta mennyisége: 500	Kereskedő fajtái száma: 2 1. fajta neve: golden 1. fajta mennyisége: 50 2. fajta neve: starking 2. fajta mennyisége: 100

Kimenet	
Termelőnél marad: jonagold 100 idared 500	Kereskedő vesz: golden 30

Az A feladat megoldása a termelő és a kereskedő multihalmazának különbsége, a B feladat megoldása pedig a két multihalmaz metszete.

3. Feladat – Multihalmaz alkalmazása

Egy programozási versenyen minden versenyző választhat egy programozási nyelvet, amin dolgozni fog. Készíts programot a következő feladat megoldására.

A programod olvassa be a választható nyelvek számát (1≤M≤10) és a versenyen induló tanulók számát (1≤N≤100), majd a választható nyelveket, s legvégül az egyes tanulók által választott nyelveket!

Ezután adja meg, hogy mely tanulók választottak illegális nyelvet (olyat, ami nem szerepelt a felsoroltak között), mely nyelveket nem választotta senki, s melyik választott nyelvet hányan választották!

Bemenet	Kimenet
Nyelvek száma: 3 Versenyzők száma: 5 Választható nyelvek: Pascal Logo C++ Választott nyelvek: Pascal Pascal Delphi C++ Pascal	Illegális nyelv: 3. versenyző Nem választott nyelv: Logo Választott nyelvek: Pascal: 3 versenyző C++: 1 versenyző

A harmadik részfeladat egy multihalmaz előállítása, majd kiírása.