Podatkovni tipi in konstante

Programiranje 2, Tomaž Dobravec

Java – strogo tipiziran jezik

- Pred uporabo moramo spremenljivko deklarirati

```
int i;
```

Z deklaracijo spremenljivki določimo tip.

- Hkrati z deklaracijo lahko spremenljivko tudi inicializiramo:

```
int i = 42;
```

Primitivni podatkovni tipi

Tip	Pomen	Velikost	Privzeta vrednost
`char`	znak	2	'\u0000'
`boolean`	logični vrednosti `true`/`false`	1	`false`
`byte`	celo število	1	0
`short`	celo število	2	0
`int`	celo število	4	0
`long`	celo število	8	0L
`float`	realno število (enojna natančnost)	4	0.0f
`double`	realno število (dvojna natančnost)	8	0.0d

Znakovni tip `char`

- Tip `char` uporabljamo za shranjevanje enega znaka

```
char znak = 'A';
```

- Znak je v *javi* predstavljen z dvema bajtoma, zato lahko hranimo poljuben Unicode znak

```
char pi = '\u03C0'; //  Unicode znaka  pi
```

- Prvih 127 znakom Unicode ustreza prvih 127 znakov ASCII

ASCII tabela

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Računanje z znaki

- Znak in številka sta močno povezana

- Vsakemu znaku pripada številka (pretvorba poteka po ASCII oziroma po Unicode tabeli)

```
char a = 65;  // a = 'A' (ASCII tabela)

char b = 'A', c = 'D';
int i = c - b;  // i = 3 ('D' - 'A' = 3)

char p = ' ';
System.out.printf("ASCII koda presledka je: %d", (int) p);
```

Znakovne konstante

- Ubežne sekvence:

Znak	Pomen	Znak	Pomen
\n	prehod v novo vrstico	\t	tabulator
\b	pomik nazaj	\r	pomik na zacetek vrstice
\f	nova stran	\a	zvočni signal
\\	znak \	\?	znak ?
\'	znak '	\"	znak "

```
char a = 'A',          // znak 'A'
     b = '\n',         // nova vrsta
     c = '\a',         // zvoncek
     pi = '\u03C0';    // Unicode znak pi
```

Celoštevilski podatkovni tipi

- Java pozna 4 celoštevilske tipe:

Tip	Bajti	Obseg		
byte	1	-128	127	
short	2	-32.768	+32.767	(32kb)
int	4	-2.147.483.648	+2.147.483.647	(2ib)
long	8	–9,223,372,036,854,775,808	+9,223,372,036,854,775,807	(9Eib)

Vsi *javanksi* celoštevilski tipi so predznačeni.

- Konstante:

```
int a= 45;              // a = 45
int b=-1;               // b = -1
int c=012;              // c = 10   (osmisko)
int d=0xFF;             // d = 255  (sestnajstisko)
```

Realni podatkovni tipi

- Java pozna dva realne podatkovna tipa:

`float` in `double`

- Konstante

```
double d1 = 123.4;
double d2 = 1.234e2;
float  f1 = 123.4f;
```

Nizi – razred `String`

- Deklaracija in inicializacija:

```
String niz1 ="To je nek niz";

String niz2;
niz2 = "Danes je lep dan";

String niz3 = new String("Tudi tako gre");
```

- Dolžina niza (metoda `length()`):

```
String a="POMLAD";
int dolzina = a.length() ;        //   dolzina =  6
```

- i-ti znak (metoda `charAt()`):

```
System.out.println(a.charAt(3));      // izpise L
```

Nizi – primerjanje in spreminjanje

- Nizov **NE** primerjamo z operatorjem ==
- Za primerjavo uporabimo metodo `equals()`

```
String a = preberiIme(); // prebere ime in ga shrani v a

// NAROBE!                    // PRAVILNO!!!
if (a == "Lojze") ...         if (a.equals("Lojze"))  ...
```

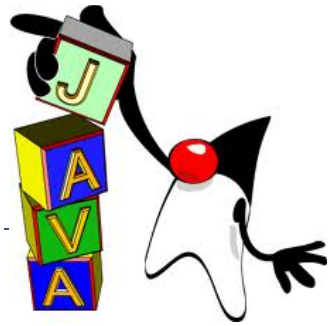
- Niza **ne moremo** spreminjati (niz je konstanta)!
  - Metoda `setCharAt()` ne obstaja!

Metode razreda `String`

- `compareToIgnoreCase()`
  primerja dva niza in pri tem zanemari velikost črk
- `startsWith()`
  ali se niz začne z danim podnizom
- `endsWith()`
  ali se niz konča z danim podnizom
- `indexOf()`
  poišče mesto prve pojavitve podniza
- `substring()`
  vrne podniz
- `replace()`
  zamenja prvo pojavitev podniza
- `replaceAll()`
  zamenja vse pojavitve podniza
- `split()`
  razbije na podnize
- ...

Naloga

Palindrom



`razno/Palindrom.java`

Napiši metodo `jePalindrom()`, ki za podani niz preveri, ali je palindrom (t.j., ali se dani niz enako bere v obe smeri).

Primer:

```
jePalindrom("pomlad")   -> false
jePalindrom("cepec")    -> true
```

Razbitje nizov – metoda `split()`

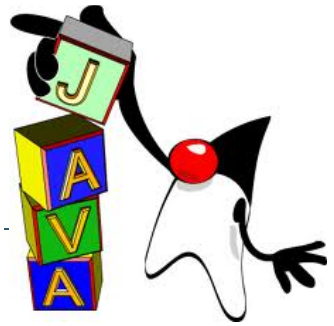
Primer:

```
String vrstica = "leo:x:131:100:Leo Novak:/home/leo:/bin/bash";
String polja[] = vrstica.split(":");
System.out.println(polja[4]);  // izpis: Leo Novak
```

- Rezultat je tabela nizov.
- Ločilo je regularni izraz.

Naloga

Barva las



razno/RjaviLasje.java

V datoteki `studenti.txt` so zapisani podatki o študentih. Vsakemu študentu pripada vrstica oblike

```
ID:Ime:Priimek:OcenaP2:Visina:BarvaLas
```

Primer datoteke s podatki o študentih:

```
63000001:Micka:Kovaceva:9:183.5:RJAVA
63000002:Janez:Novak:8:176:RDECA
63000003:Miha:Dolzan:10:180.5:CRNA
63000004:Metka:Maze:10:178.5:RJAVA
63000005:Tadej:Hocevar:9:165.2:CRNA
63000006:Stefka:Drobtina:6:178:BLOND
```

Preberi datoteko in izpiši imena vseh študentov, ki imajo rjave lase.

Razred `StringBuffer`

- Javanski niz (`String`) se ne more spreminjati.

- Kaj izpiše spodnji program?

```
String dan = "ponedeljek";
dan = "torek";
System.out.println("Danes je " + dan);
```

Zakaj?

Razred `StringBuffer`

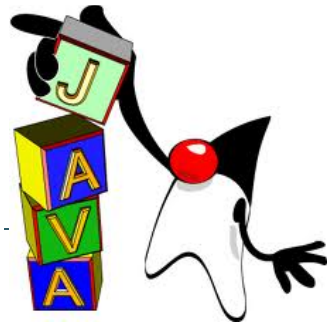
- Za "spreminjajoče" nize uporabimo razred StringBuffer

```
StringBuffer ime = new StringBuffer("miha");
ime.setCharAt(0, 'M');
System.out.println(ime);
```

- Metode razreda StringBuffer:
  - `setCharAt(), chatAt()`
  - `append(), insert(), delete(),`
  - `indexOf, substring(),`
  - `reverse(), replace, getChars`
  - …

Naloga

Uporaba razreda `StringBuffer`



`razno/DodajKImenu.java`

Napiši metodo

```
String dodajKImenu(String ime, String dodatek)
```

ki k danemu imenu datoteke doda predpisani niz.

Primer:

```
dodajKImenu("podatki.txt", "CRC")  -> "podatkiCRC.txt".
```

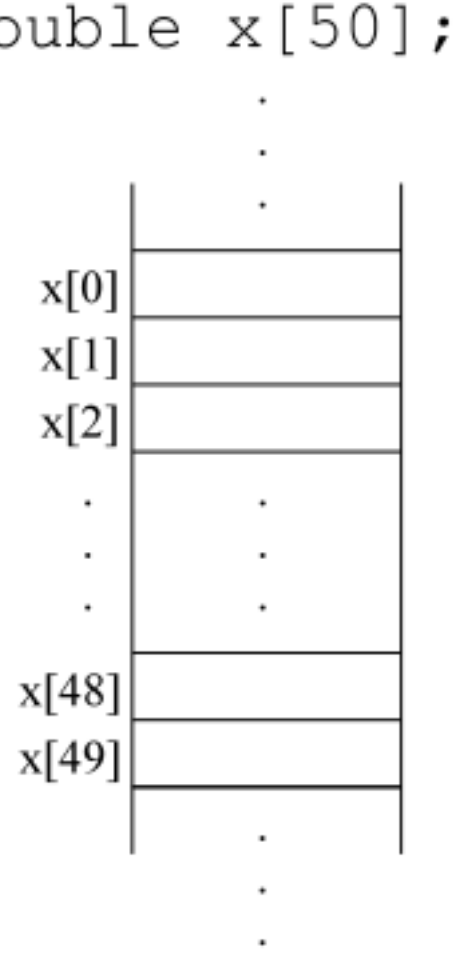
Tabele

- Tabela v Javi ima nespremenljivo dolžino
- Deklaracija tabele: `int tabela[];`
- Inicializacija tabele:
- Rezerviram prostor in nastavim vrednosti

```
tabela = new int[3];
tabela[0] = 1;
tabela[0] = 6;
tabela[0] = 3;
```

- Lahko tudi vse hkrati takole:

```
int tabela[] = {1,6,3};
```



Tabele

- Še en primer:

```
int tabela[] = new int[3]; // v tabeli hranimo 3 stevila
tabela[0] = 15; tabela[1] = 7; tabela[2] = 105;

String imena[] = {"Micka", "Miha", "Janez", "Lojzka"};
System.out.println(imena[2]);    // Janez
```

- Dolžina tabele: atribut `length`

```
short meseci[] =
   {31,28,31,30,31,30,31,31,30,31,30,31};
```

```
int  dolzina = meseci.length;
System.out.println(dolzina); // izpise se 12
```

Tabele

- Indeksiranje: od `0` do `tabela.length-1`

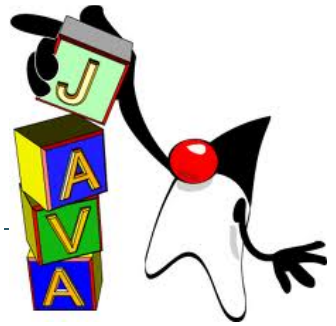
```
tabela[0], tabela[1], …, tabela[tabela.length-1]
```

- Uporaba indeksa izven tega obsega vrže izjemo

```
ArrayIndexOutOfBoundsException
```

Naloga

Mešanje črk niza



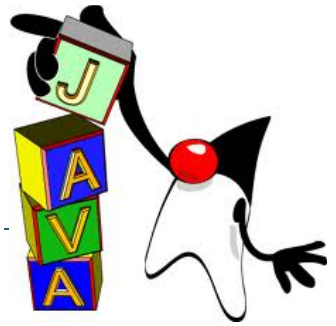
`razno/Anagram.java`

Napiši program, ki prebere niz, premeša njegove črke in ga izpiše.

Namig: prebrani niz najprej pretvori v tabelo znakov, nato premešaj črke v tabeli in na koncu izpiši vsebino tako premešane tabele.

Naloga

Kodiranje s pomočjo slovarja



razno/Skrivno.java

V datoteki `besede.txt` so zapisane besede, v datoteki `skrivno.txt` pa je navodilo, kako iz besed sestaviti sporočilo (datoteka `skrivno.txt` vsebuje zaporedje parov (`stevilka_besede, stevilka_crke`)).

Primer:

```
besede.txt:
        To je vsebina datoteke z besedami. V njej so razlicne besede ki
        imajo lahko pomen ali pa tudi ne.
skrivno.txt:
        4 3 19 2 3 2 4 3
```

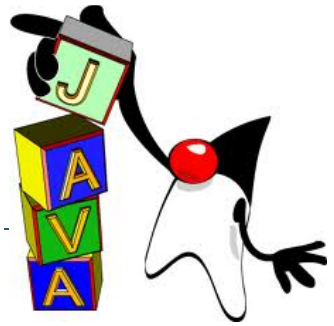
Pomen skrivnega besedila: 4 3 (4. beseda, 3. crka: t), 19 2 (19. beseda, 2. crka: e), 3 2 (3. beseda, 2. crka: s) in 4 3 (4. beseda, 3. crka: t) – test.

Napiši program, ki s pomočjo obeh datotek sestavi sporočilo in ga izpiše na zaslon.

Opomba: koda 0 0 pomeni presledek!.

Naloga

Popravljen program za Loto



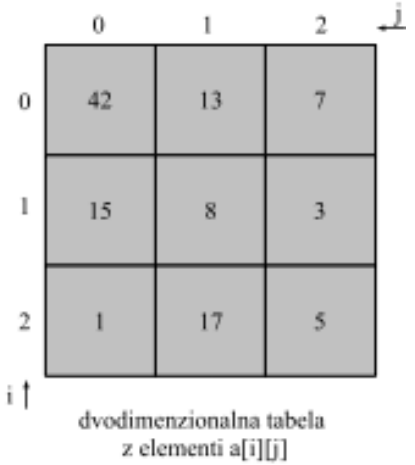
`tabele/Loto2.java`

Popravi program Loto.java tako, da se izpisana števila ne bodo ponavljala.

Namig: preden novo naključno število vstaviš v tabelo, preveri, če v tabeli tako število že obstaja.

Večdimenzionalne tabele

- Pri večdimenzionalnih tabelah do elementov dostopamo z več indeksi:
  - 2. dimenzionalno: `a[1][4] = 4;`
  - 3. dimenzionalno: `a[4][0][7] = 1;`
  - ...



dvodimenzionalna tabela
z elementi a[i][j]

- Deklaracija in inicializacija s privzetimi vrednostmi:

```
int a[][] = new int [3][3];
```

Večdimenzionalne tabele

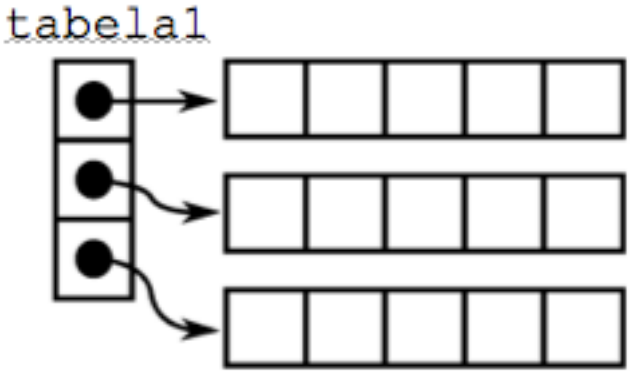
- Deklaracija in inicializacija s podanimi vrednostmi

```
int y[3][3] = {
    {7, 2, 1},    /* y[0][0], y[0][1], y[0][2] */
    {3, 9, 4},    /* y[1][0], y[1][1], y[1][2] */
    {5, 8, 6}     /* y[2][0], y[2][1], y[2][2] */
};
```

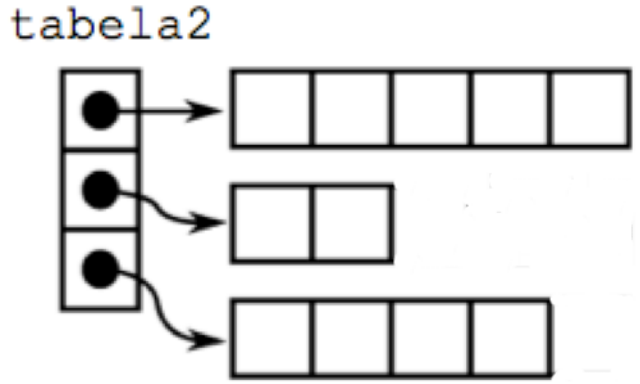
Večdimenzionalne tabele

Večdimenzionalna tabela v javi je **tabela tabel**

```
int tabela1[][] = new int[3][5];
```

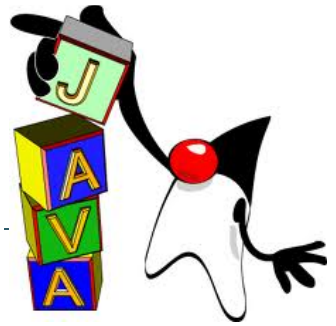



```
int tabela2[][] = new int[3][];

tabela2[0] = new int[5];
tabela2[1] = new int[2];
tabela2[2] = new int[4];

tabela2[0][2] = 5; // OK
tabela2[1][2] = 4; // napaka
```



Naloga

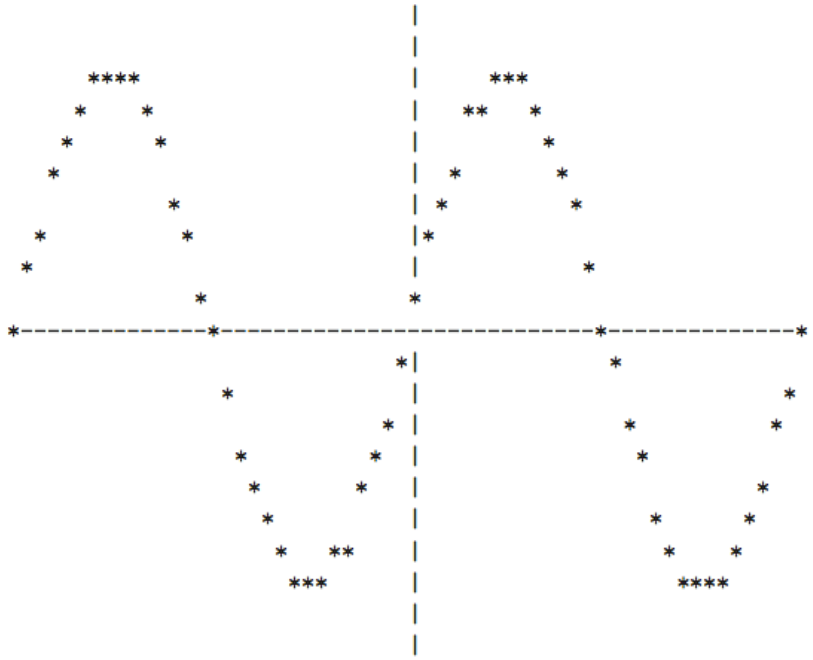
Risanje matematičnih funkcij



`tabele/Graf.java`

Napiši program za risanje grafov matematičnih funkcij na tekstoven zaslon.

Primer: sin(x)



$$x = \frac{x2 - x1}{79} * i + x1, \quad j = 24 * \frac{y - y1}{y2 - y1}$$

Primitivni in ovojni podatkovni tipi

- Java pozna 8 osnovnih podatkovnih tipov: `byte`, `short`, `int`, `long`, `float`, `double`, `char` in `boolean`.

- Za vsak osnovni podatkovni tip obstaja tudi ovojni (wrapper) podatkovni tip

Tip	Število bitov	Ovojni tip
byte	8	Byte
short	16	Short
int	32	Integer
long	64	Long
float	32	Float
double	64	Double
char	16	Character
boolean		Boolean

Primitivni in ovojni podatkovni tipi

```
int a = 5;                          // primitivni tip z vrednostjo 5
Integer aObj = new Integer(5);      // objekt tipa z "vrednostjo" 5
```

Prednost primitivnih podatkovnih tipov je, med drugim ta, da lahko nad njimi izvajamo atomarne operacije

```
int a = 5, b = 7;
int c = a + b;
```

Nad objekti tega ne moremo početi:

```
Integer aObj = new Integer(5);
Integer bObj = new Integer(7);
Integer cObj = aObj + bObj;     // NAPAKA!!! Tega ne smemo poceti!
```

Primitivni in ovojni podatkovni tipi

- Ovojne podatkovne tipe bomo uporabljali za **pretvorbo** med nizom in osnovnim podatkovnim tipom (v obe smeri):

```
String odgovorStr = "42";
int    odgovor    = Integer.parseInt(odgovorStr);  // 42

String piStr = "3.14";
double pi    = Double.parseDouble(piStr);         // 3.14
```

in obratno, za pretvorbo primitivnega tipa v niz

```
String aStr = Integer.toString(15);  // "15"
String bStr = Double.toString(2.78); // "2.78"
```

Pretvorba tipov