Operatorji in izrazi

Programiranje 2, Tomaž Dobravec

Operatorji in izrazi

Aritmetični operatorji

- unarni operator (en sam operand): – in +

```
int a=-1;
int b=+15;     // isto kot int b=15;
```

- binarni operator (dva operanda) +, -, *, /, %

```
int a=8, b=2, c;

c = (a + b) * 3;   // c = 30;
c = a / b;         // 4 (celostevilsko deljenje)
c = 33 % 5;        // 3 (ostanek po deljenju s 5)

float x = 3, y = 2, z;
z = x / y;         // z = 1.5 (realno deljenje)
```

Relacijski in logični operatorji

Relacijski operatorji

<	manjše
<=	manjše ali enako
>	večje
>=	večje ali enako
==	enako
!=	različno

Logični operatorji

!	negacija
&&	konjunkcija (logični in)
||	disjunkcija (logični ali)

Relacijski in logični operatorji

- Logični izrazi se preverjajo od leve proti desni. Izračun vrednosti logičnih izrazov se konča takoj, ko je znana končna vrednost.

```
PRIMER:  if (a==2 && zmanjsajStevec() > 0) {…}
```

- Pravilo ARL: Aritmetični operatorji imajo prednost pred relacijskimi, relacijski pa pred logičnimi

`i < a - 1`	**isto kot**	`i < (a-1)`
`a < 5 && b > 3`	**isto kot**	`(a<5) && (b>3)`

Operatorja ++ in --

```
              ... isto kot ...
++i;                              i = i + 1;
--i;                              i = i - 1;

x = ++n;                          n = n + 1; x = n;
y = --i;                          i = i - 1; y = i;

t[++i]=a;                         i = i + 1; t[i] = a;
```

```
              ... isto kot ...
i++;                              i = i + 1;
i--;                              i = i - 1;

x = n++;                          x = n; n = n + 1;
y = i--;                          y = i; i = i - 1;

t[i++]=a;                         t[i] = a;  i = i + 1;
```

Operatorji +=, -=, *=, /= in %=

```
            ... isto kot ...
i+=5;                          i = i + 5;
i-=2;                          i = i - 2;
i/=2;                          i = i / 2;
i*=2;                          i = i * 2;
i%=2;                          i = i % 2;
```

Bitni operatorji

Java pozna naslednje bitne operatorje:

&	in	AND
|	ali	OR
^	ekskluzivni ali	XOR
<<	predznačen pomik bitov v levo	LSHIFT
>>	predznačen pomik bitov v desno	RSHIFT
>>>	nepredznačen pomik bitov v desno	
~	eniški komplement	NOT

Bitni operatorji

- Primer delovanje bitnih operatorjev:

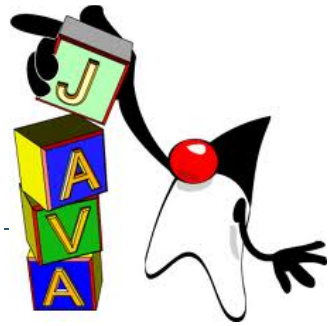
```
 01011     01011     01011     ~01011
&00101    |00101    ^00101
=00001    =01111    =01110     =10100
```

- Množenje in deljenje z 2 s pomočjo bitnih operatorjev:

```
a = b << 1;     // isto kot a=b*2
a = b << 3;     // isto kot a=b*8
a = b >> 1;     // isto kot a=b/2
a = b >> 10;    // isto kot a=b/1024
```

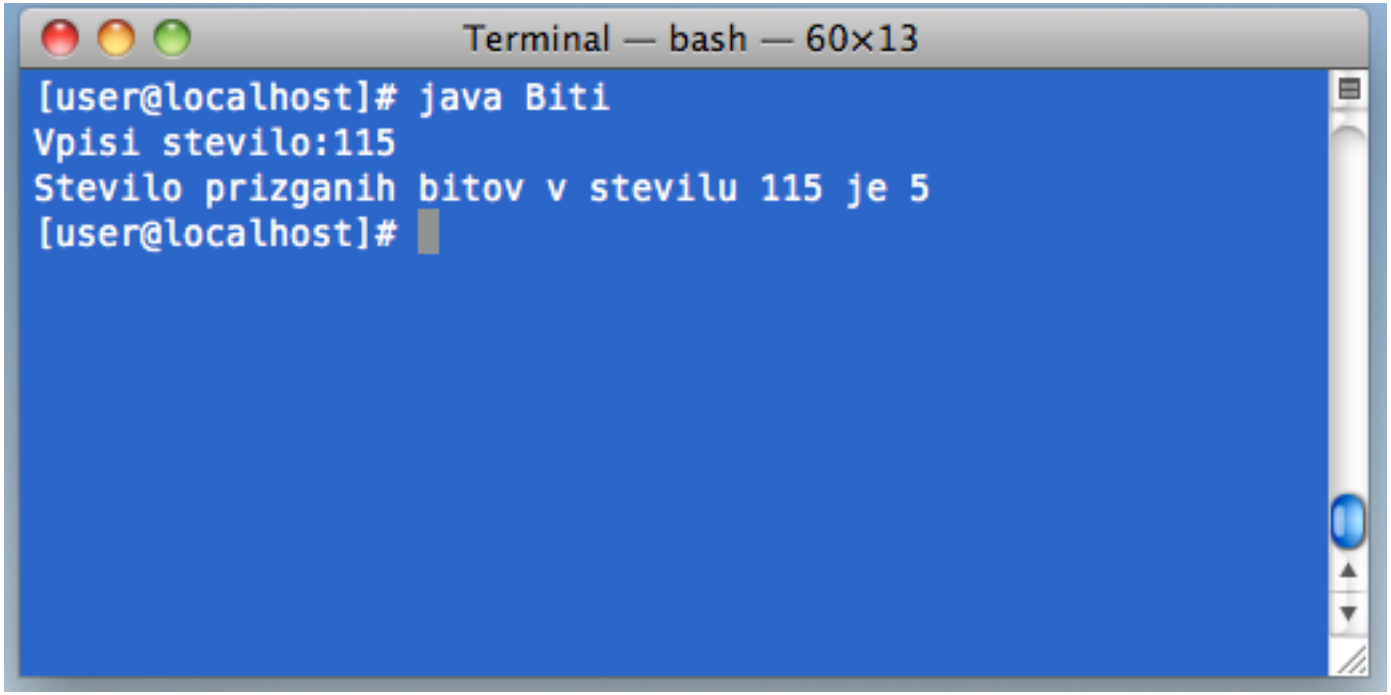
Naloga

Bitni operatorji



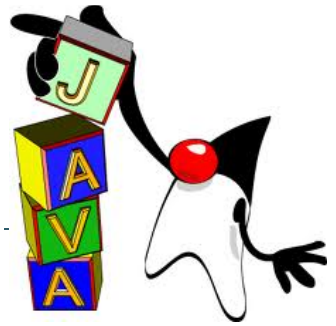
`operatorji/Biti.java`

- Napiši program, ki izpiše število prižganih bitov v prvem argumentu programa.



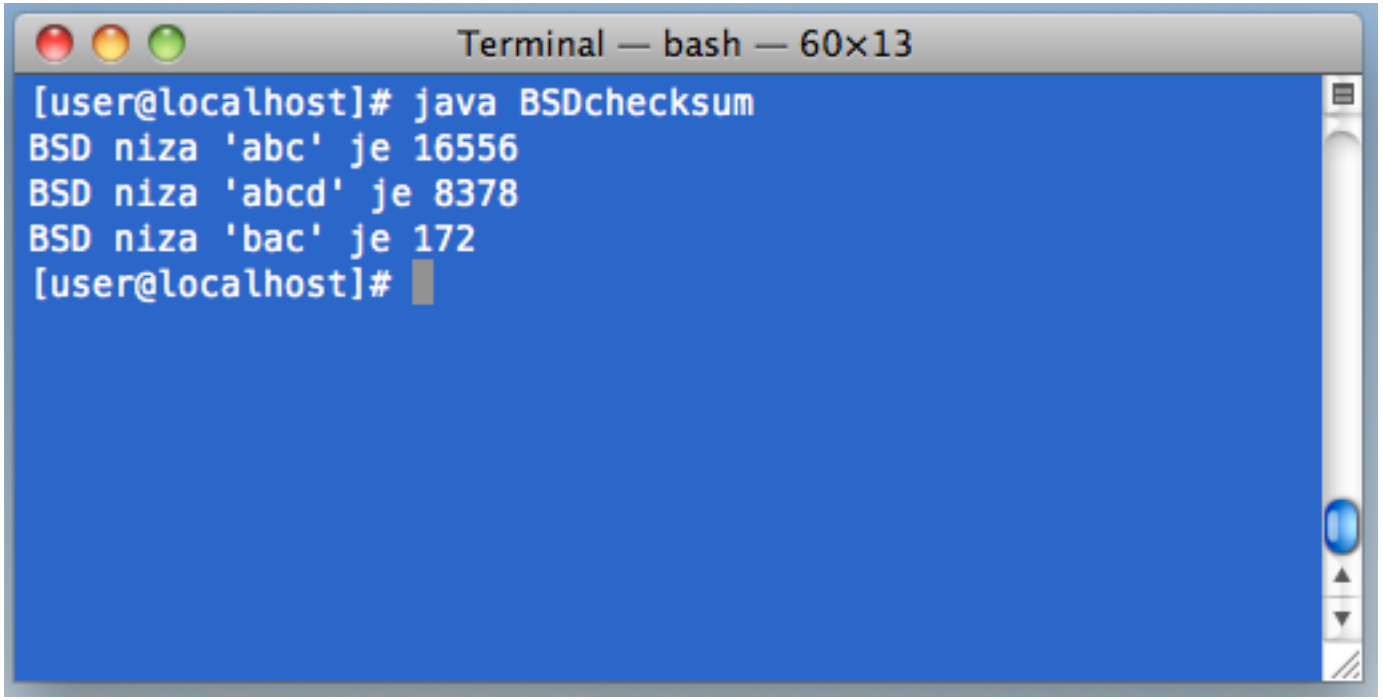
Naloga

Bitni operatorji



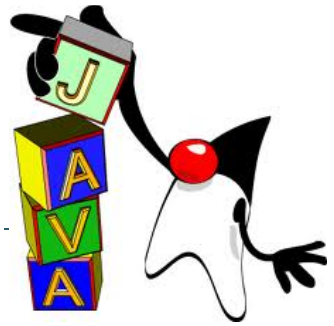
`operatorji/BSDchecksum.java`

- Napiši metodo, ki izračuna BSD kontrolno vsoto podanega niza (pomoč: glej wiki).



Naloga

Dvojiški zapis



`operatorji/Dvojisko.java`

Napiši metodi

```
static String vDvojisko(int x)
```

in

```
static int vDesetisko(String d)
```

za pretvorbo med desetiškim in dvojiškim številskim sistemom. Metodi preveri v `main` metodi na nekaj primerih.

Operator ?

- Namesto

```
if (pogoj)
  rezultat = izraz1;
 else
  rezultat = izraz2;
```

lahko pišemo tudi

```
rezultat = pogoj ? izraz1 : izraz2;
```

Primer:

```
printf("%s", x < 37 ? "OK" : "VROCINA!");
```

Prireditveni stavek

- **Prireditveni stavek vrne vrednost.**
- Rezultat prirejanja `b = a` je `a`.

Uporaba:

- `c = b = a;`
- ```
  while ((c=fis.read()) != -1) {
    …
  }
  ```
  ```

Prioriteta in asociativnost operatorjev

Vprašanje 1: Se bo v izrazu

```
d = a + b * c;
```

najprej izračunal seštevek ali zmnožek?

Odgovor: najprej zmnožek, saj ima * večjo prioriteto kot +.

Prioriteta in asociativnost operatorjev

Vprašanje 2: Se bo v izrazu

```
d =  8 / 4 / 2;
```

najprej delilo z 2 ali s 4?

Odgovor: najprej s 4, saj je / levo-asociativen.

Prioriteta in asociativnost operatorjev

Operatorji	Asociativnost
(), [], .	L
!, ~, ++, --, +, -, (type), new	D
*, /, %	L
+, -	L
<<, >>, >>>	L
<, <=, >=, >, instanceof	L
==, !=	L
&	L
^	L
|	L
&&	L
||	L
?:	D
=, + =, − =, * =, / =, %=, &=, ^=, | =, <<=, >>=, >>>=	D

Tabela: Prioriteta in asociativnost operatorjev v *javi*