Java

Java

uvod

- Just Another Vague Acronim - JAVA
- 1991. tvorac Jave – James Gosling, *Sun Microsystems*
  - Stvorio jednostavan, platformski nezavistan jezik
  - Namenjen pokretanju elektronskih uređaja (Interaktivna TV, inteligentne rerne, telefoni,..)
- 1994. Java se ugrađuje u web browser WebRunner
- 1995. obavljuje se kod i dokumentacija Jave na Internetu

Java
osnovne karakteristike

- Objektna orijentacija
  - podržava sve koncepte objektno orijentisanog programiranja
  - sintaksa slična C++ ali su izbačeni složeni koncepti (pointeri)
- Prenosivost
  - Java programi se prevode u byte kod koji nije mašinski jezik nijednog konkretnog računara, već se izvršava na JVM
  - Java Virtuelna Mašina - je virtuelni računar koji može biti simuliran na bilo kom računaru
- Prirodna prilagođenost Internetu
  - Java programi mogu da se izvršavaju u Web browserima
  - Poseduju sigurnosne mehanizme
  - Mogu da se distribuiraju i izvršavaju na različitim mašinama
  - Podržava konkurentno programiranje
  - Omogućene klase za korisnički interfejs (API) koji omogućuje jedinstven izgled i korišćenje aplikacija

Java kao platforma

- Dizajniran da što manje zavisi od specifičnih karakteristika konkretnog računarskog sistema
- Jednom napisan i preveden program se izvršava na bilo kojoj platformi koja podržava Javu
- Interpretirani jezik, bajt-kod
- Java virtuelna mašina (JVM)
- Dve vrste Java programa
  - Apleti
    - izvršavaju se u okviru WWW čitača
    - automatska distribucija i instalacija
    - ograničene mogućnosti apleta iz razloga bezbednosti
  - Aplikacije

Java od početka

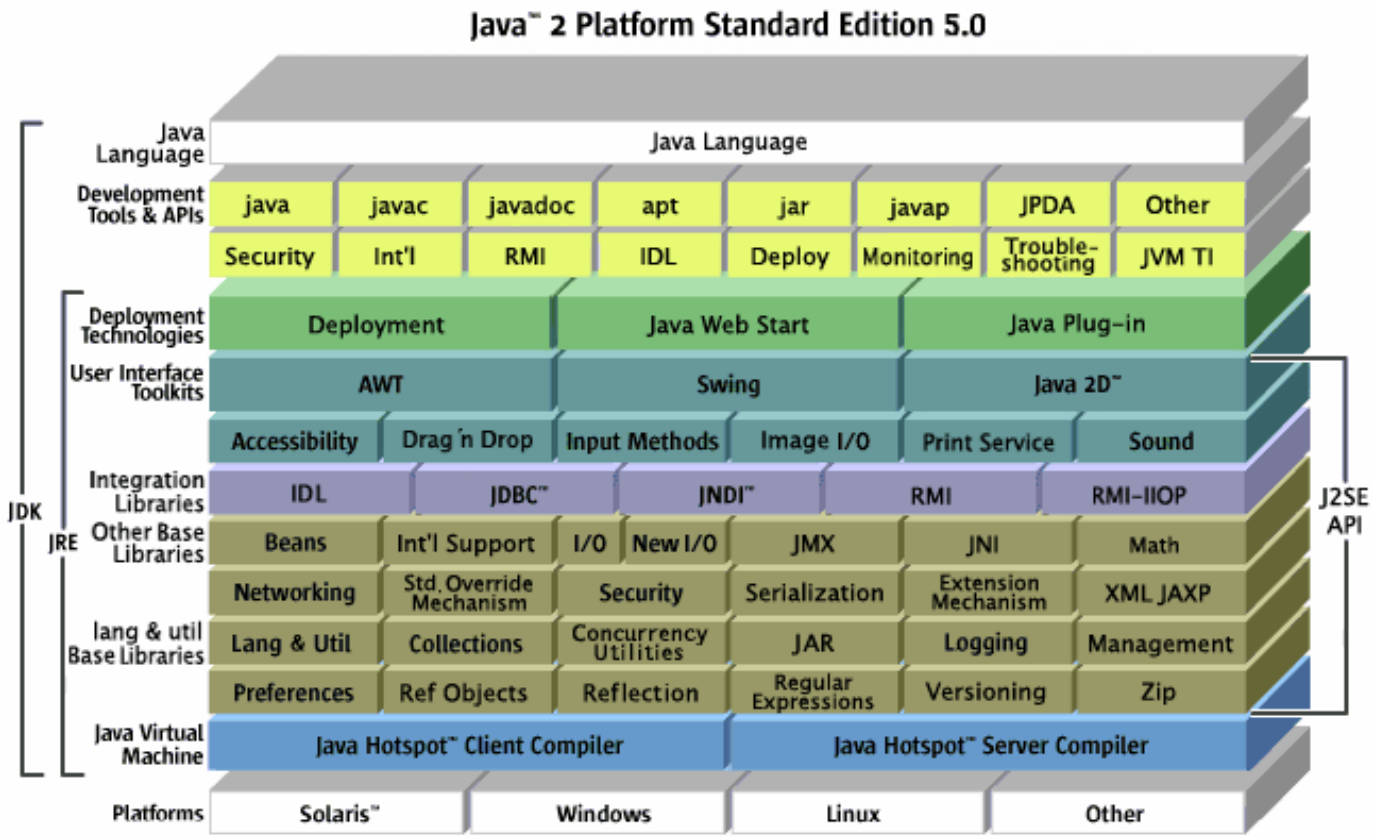
- Verzije
  1.0
  1.1 i
  Java 2 platforma (1.2,1.3,1.4…)

- Free download
  java.sun.com, www.sun.com

- Dokumentacija

Java 2 Platforma

- Javina platforma se sastoji od tri elementa: Java programskog jezika + Java API biblioteka + Javine virtalne mašine.
- Java 2 platforma je skup programa i sistemskih resursa koji su specifični za dati operativni sistem (Windows, Linux, UNIX, Mac, ...), a omogućuje prevođenje i izvršavanje Java programa.



Java 2 Platforma

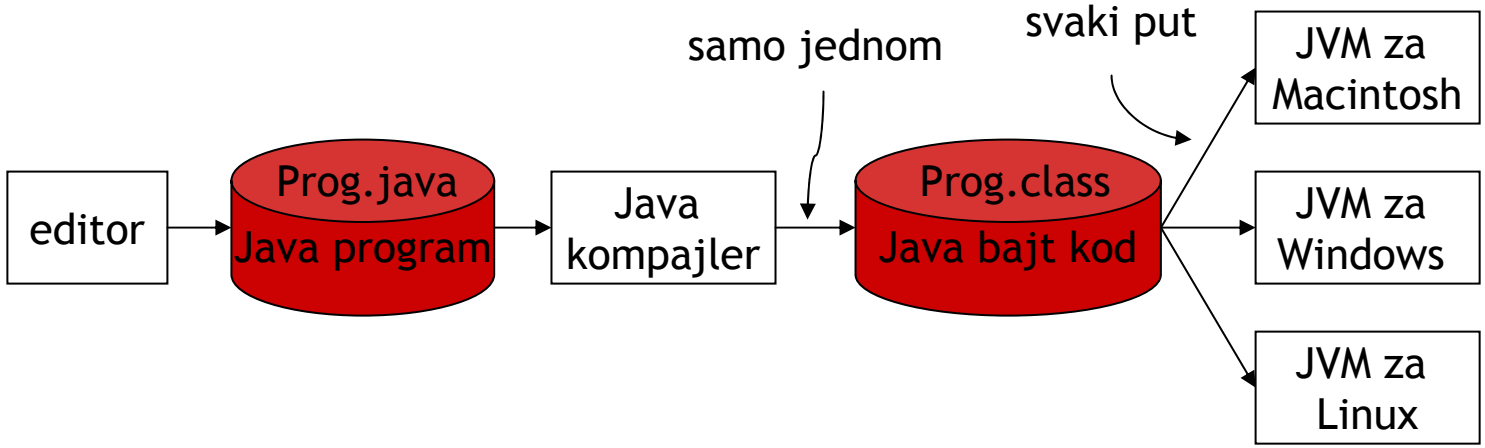
- **Java API (Application Programming Interface)** je skup već gotovih softverskih komponenti napisanih u Java programskom jeziku koji su podeljeni u pakete (eng. package), a svaki se sastoji od klasa (eng. class) koje enkapsuliraju (eng. encapsulate) određenu funkcionalnost.
- **java.applet** - Javini programi se mogu izvršavati unutar Web preglednika (npr. Internet Explorer, Mozilla, Netscape); ovaj paket enkapsulira tu funkcionalnost.
- **java.awt, javax.swing** - enkapsulira kreiranje korisničkog interfejsa, grafike i slike.
- **java.io**-input/output funkcionalnost za Java 2 platformu
- **java.lang** – podrazumevani paket, osnovni za prevođenje i interpretiranje Javinih programa
- **java.net** - namenjena Internet aplikacijama.
- **java.security** - sigurnost na Internetu.
- **java.sql** - rad sa bazama podataka.

Kompajleri i interperteri

- Program pisan u nekom od viših programskih jezika potrebno je prevesti na mašinski jezik, ne bi li bio izvršen. To prevođenje vrši **kompajle**r (compiler) odgovarjućeg programskog jezika. Nakon što je program jednom preveden, program u mašinskom jeziku se može izvršiti neograničen broj puta, ali, naravno, samo na određenoj vrsti računara.
- Postoji alternativa. Umesto kompajlera, koji odjednom prevodi čitav program, moguće je koristiti **interpreter**, koji prevodi naredbu po naredbu prema potrebi. Interpreter je program koji se ponaša kao CPU s nekom vrstom dobavi-i-izvrši ciklusa. Da bi izvršio program, interpreter radi u petlji u kojoj uzastopno čita naredbe iz programa, odlučuje šta je potrebno za izvršavanje te naredbe, i onda je izvršava (oni se mogu koristiti za izvršavanje mašinskog programa pisanog za jednu vrstu računara na sasvim različitom računaru).

Java kompajler i interpreter

- Projektanti Jave su se odlučili za upotrebu kombinacije kompajliranja i interpretiranja. Programi pisani u Javi se prevode u mašinski jezik virtuelnog računara, tzv. **Java Virtual Machine**.
- Mašinski jezika za Java Virtual Machine se zove **Java bytecode**. (Sun Mycrosystems, začetnik Jave, razvio je CPU koji izvršava Java bajt kod u originalu, bez interpretiranja).
- Sve što je računaru potrebno da bi izvršio Java bajt kod jeste interpreter. Takav interpreter oponaša Java virtual machine i izvršava program.



Hello, world

```
public class HelloWorld {
    public static void main(String[ ] args) {
        System.out.println("Hello, World");
    } // kraj main metode
} // kraj klase HelloWorld
```

case sensitive
vodite računa o velikim
i malim slovima

- svaka Java aplikacija mora sadržati barem jednu klasu s metodom main(String[] args)
- počinje svoje izvršavanje pozivom metode **main**
- ovako napisan program se prevodi izvršavajući

  javac HelloWorld.java
- Ako nema grešaka prevodilac javac kreira datoteku HelloWorld.class koja sadrži bytecode instrukcije za JVM.
- JVM se pokreće sa

  java HelloWorld

Hello, world

- System.out.println("Hello, World");

  Klasa **System** nalazi se u paketu **java.lang**. Njeno puno ime je **java.lang.System**. Međutim, java.lang je jedini paket za koji se ne mora navesti puno ime klase. **out** je statička varijabla članica klase **System** tipa **PrintStream**; **println** je jedna od metoda klase **PrintStream**. Efekat cele naredbe je ispis stringa "Hello, World" na konzoli.

- Za API pogledati dokumentaciju Jave koja ide uz JDK (Java Development Kit)

  dirjave\docs\api\index.html

Java od početka

- Potrebni i skoro dovoljni uslovi
  1. prihvatiti OO način razmišljanja
     (‘misliti na javi’)
  2. na početku i na ‘kraju’ čitati dokumentaciju
     (‘ne bežati od engleskog’)

- U Javi je sve čime se manipuliše objekat neke klase♀, odnosno referenca na objekat

sledi

- Imena promenljivih i rezervisane reči
- Tipovi podataka
- Operatori
- Upravljačke strukture

Identifiers and Variables

- First character: letter, a dollar sign ($), or an underscore (_)
- A character other than the first character in an identifier may be a letter, $, _ or a digit.
- None of the Java language keywords (or reserved words) can be used as identifiers.

- Declaring variables

```
<modifier> <dataType> <variableName> = <initialValue>;
private int id = 10;
int id;
```

Java

Rezervisane reči

Abstract	assert	boolean	break	byte
case	catch	char	class	const
continue	default	do	double	else
enum	extends	false	final	finally
float	for	goto	if	implements
import	instanceof	int	interface	long
native	new	null	package	private
Protected	public	return	short	static
true	strictfp	super	switch	synchronized
this	throw	throws	transient	try
void	volatile	while		

domaćice

The words goto and const are reserved to the extent that programmers cannot use them, but they have no specific meaning in the Java language.

tipovi podataka

- Strogo tipiziran jezik
  - Svaki podatak u svakom trenutku se zna kom tipu pripada
  - Prilikom deklaracije promenljive obavezno se navodi i njen tip
- Dva tipa podatka
  - Prosti tipovi podataka
    - int, char, byte, float, boolean, …
    - Svaki prost tip podatka ima tačno definisanu veličinu bez obzira na platformu na kojoj se izvršava Java kod
  - Složeni tipovi podataka
    - Objekti
    - Nizovi

Java primitivni tipovi podataka

Tip Podataka	Opis	Bitova	Označen	Opseg
boolean	Logicki tip	1	NA	`true` ili `false`
char	Karakter (Unicode)	16	Unsigned	‘\u0000’ – ‘\uFFFF’ 0 – 2^{15}-1
byte	Ceo broj	8	Signed	-128 – 127 -2^7 – 2^7 -1
short	Ceo broj	16	Signed	-32768 - 32767 -2^{15} – 2^{15} -1
int	Ceo broj	32	Signed	-2147483648 – 2147483647 -2^{31} – 2^{31} -1
long	Ceo broj	64	Signed	-2^{63} – 2^{63} -1
float	Pokretni zarez	32	Signed	-2^{31} – 2^{31} -1
double	Eksponencijalni zapis	64	Signed	-2^{63} – 2^{63} -1

```
boolean finished = 0;
```

```
if (finished = 0) ⇏ if (finished)
```

true and false are the only two valid values for a boolean type.

Java

inicijalizacija

Tip Podataka	Default vrednost
boolean	False
char	'\u0000'
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d

domaćice

Default inicijalizacija važi samo za podatke članove objekta,
ali ne i za lokalne varijable metoda (kompajler javlja grešku).

Literals

Karakteri

```
char question = '\u4567';
\n: Used to denote new line
\r: Used to denote a return
\t: Used to denote a tab
\b: Used to denote a backspace
\f: Used to denote a form feed
\': Used to denote a single quote
\": Used to denote a double quote
\\: Used to denote a backslash
```

Celi brojevi `43, 053, 0x2b, 0x2B, 0X2b, 0X2B`

(43 u dekadnom, oktalnom i heksadekadnom yapisu)

Realni brojevi: `12.33, 1.25E+8, 1.2534f, 156d`

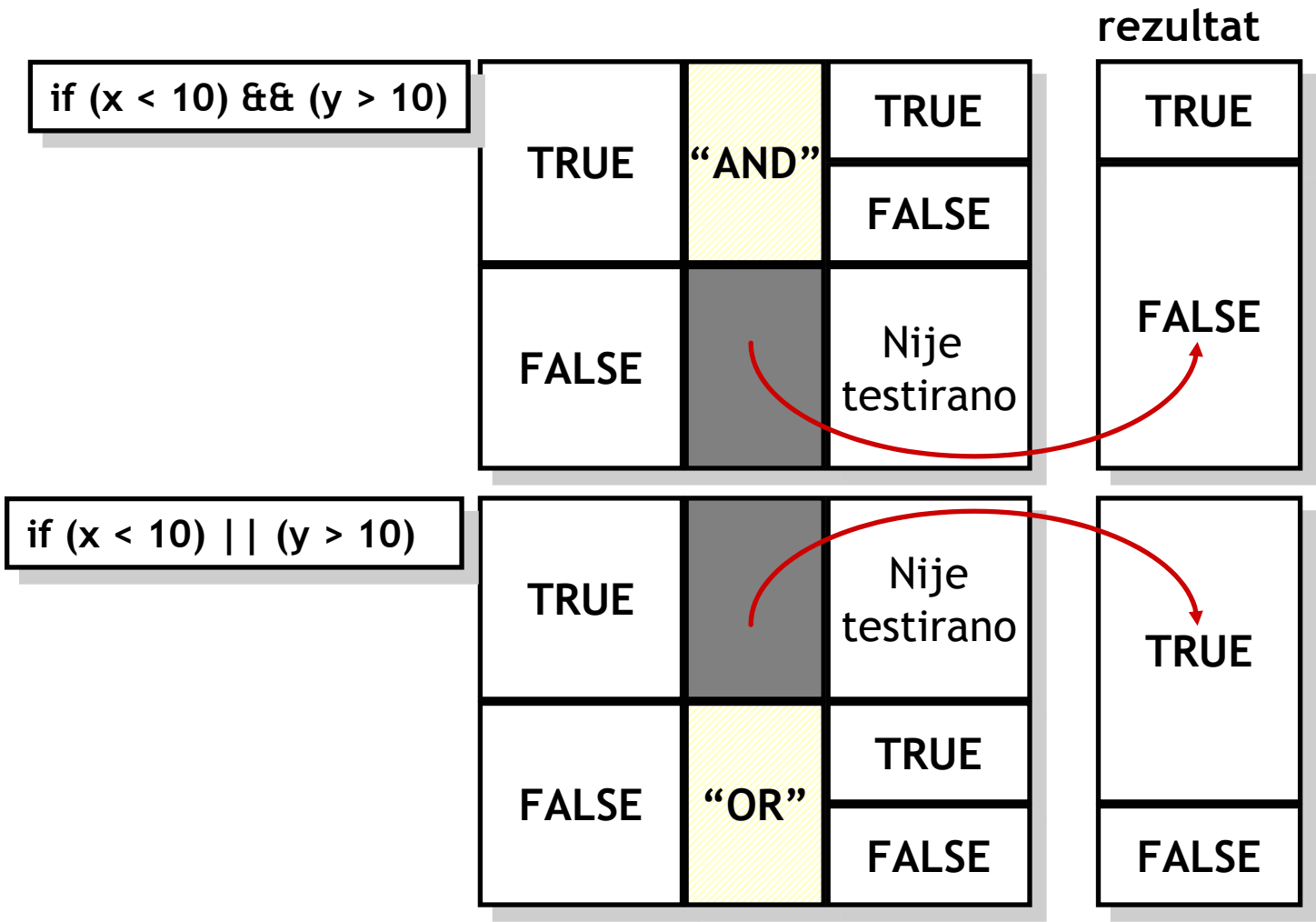
(The suffix f or F indicating a floating-point number; The suffix d or D indicating double number)

If the suffix f or d is not used, the literal is interpreted as of type double.

Java operatori

Tip operatora	Operator	Opis
Aritmetički operatori	++, -- +, -, *, /, %	Inkrementiranje , dekrementiranje (unarni operatori) Sabiranje, Oduzimanje, Množenje, Deljenje, Deljenje po modulu
Operatori poređenje	==, !=, <, >, <= , >= , %	Jednakost, Nejednakost, Manje od, Veće od, Manje ili jednako, veće ili jednako, deljenje po modulu
Logički operatori	&&, II, ^, !	Logičko AND, Logičko uključujuće OR, Logičko isključujuće XOR, Negacija
Operatori za rad sa bitovima	&, ^, ^- <<, >>	Bitsko AND, Bitsko OR, Bitsko XOR, Pomeranje bitova u levu stranu, pomeranje bitova u desnu stranu
Operatori dodele vrednosti	= +=, -=, *=, /=, %=	Dodela vrednosti Dodela vrednosti sa primenom aritmetičke operacije
Spec. operatori	+ Instanceof ?:	Konkatenacija stringova Operator za proveru pripadnosti nekog objekta klasi

Java operatori

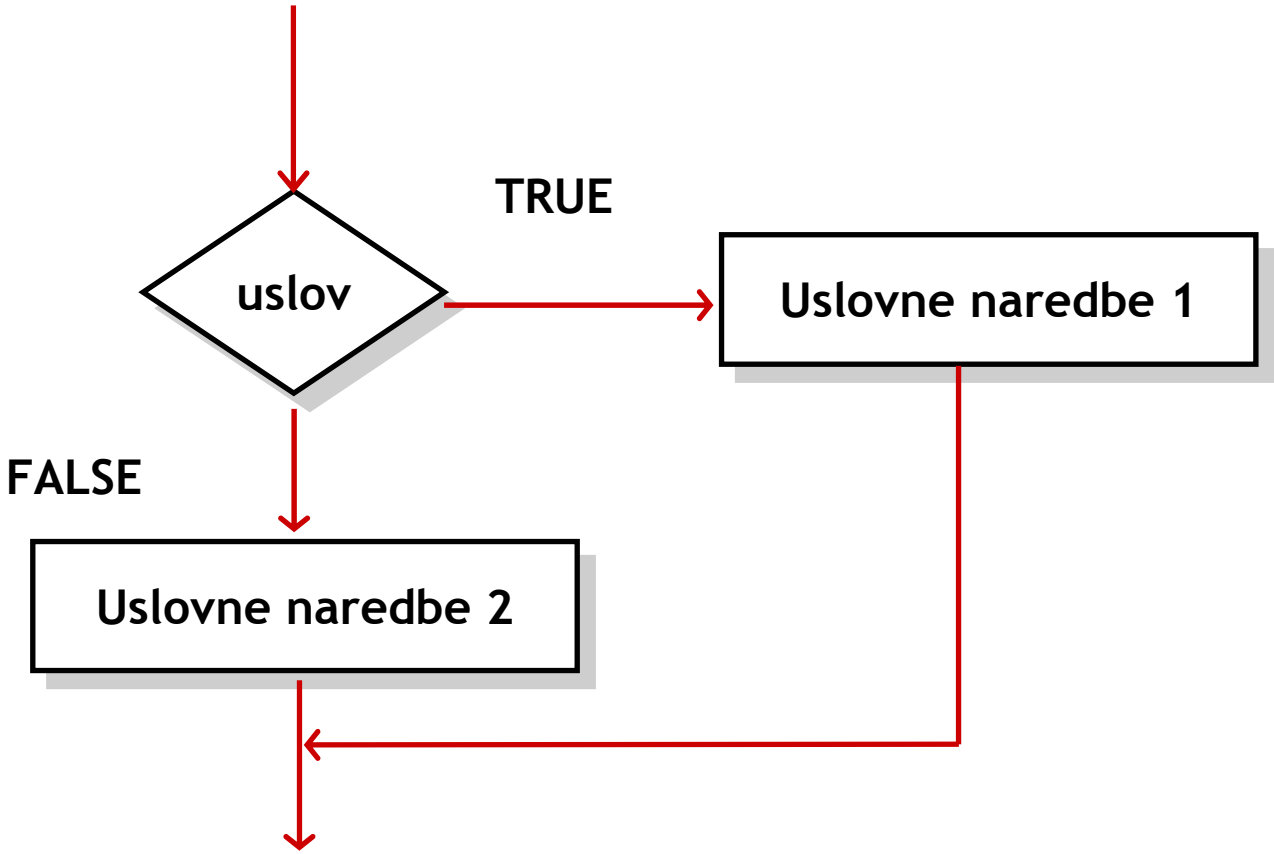


Java

upravljačke strukture

- Naredbe izbora (selekcije)
  - `if/else,`
  - `switch`
- Naredbe iteracije (petlje)
  - `for,`
  - `while,`
  - `do while`
- Naredbe za obradu izuzetaka
  - `try/catch/finally`

if

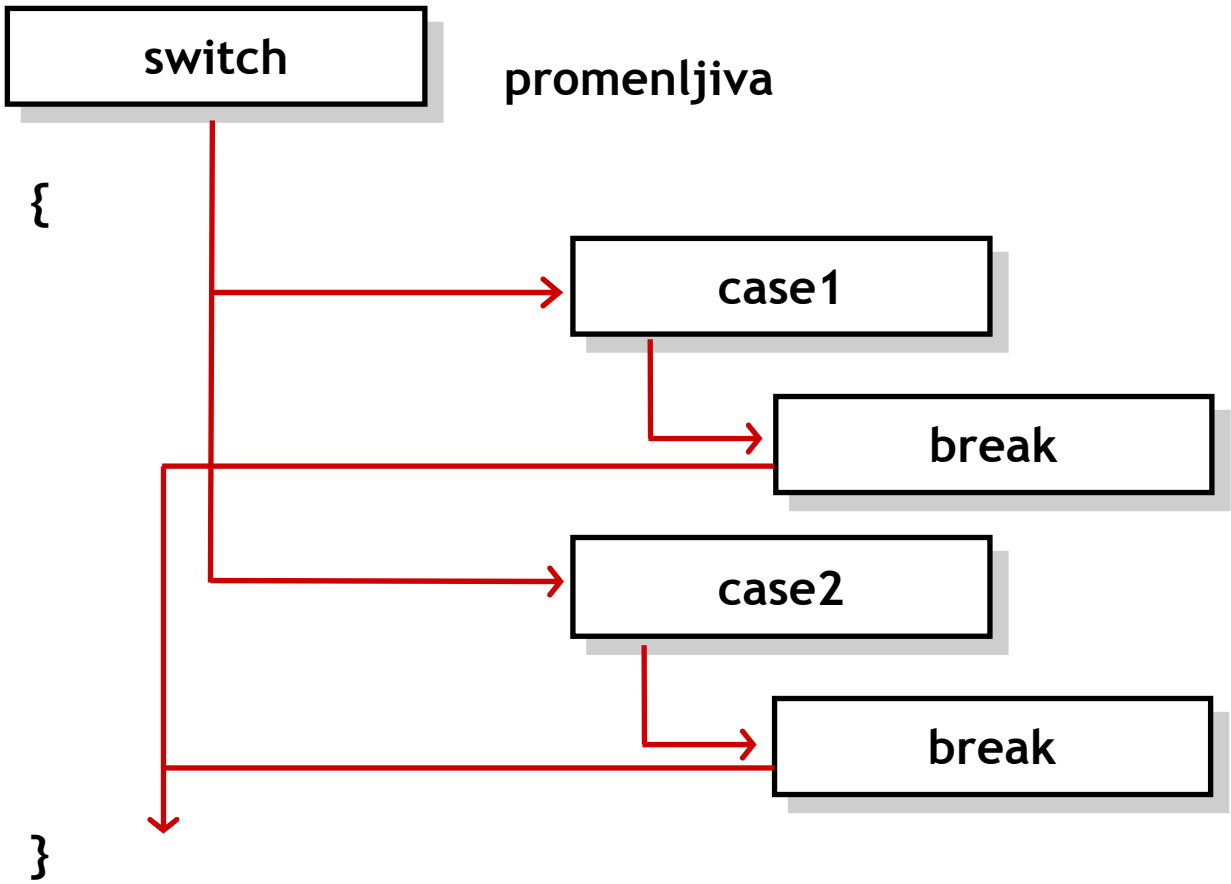


if

```
if (uslov)
 { uslovne_naredbe_1
  }
else
 { uslovne_naredbe_2
  }
```

- if,else - ključne reči
- izraz - može biti tačan ili netačan
- uslovne_naredbe_1 - izvršavaju se ako je izraz tačan
- uslovne_naredbe_2 - izvršavaju se ako je izraz netačan

switch

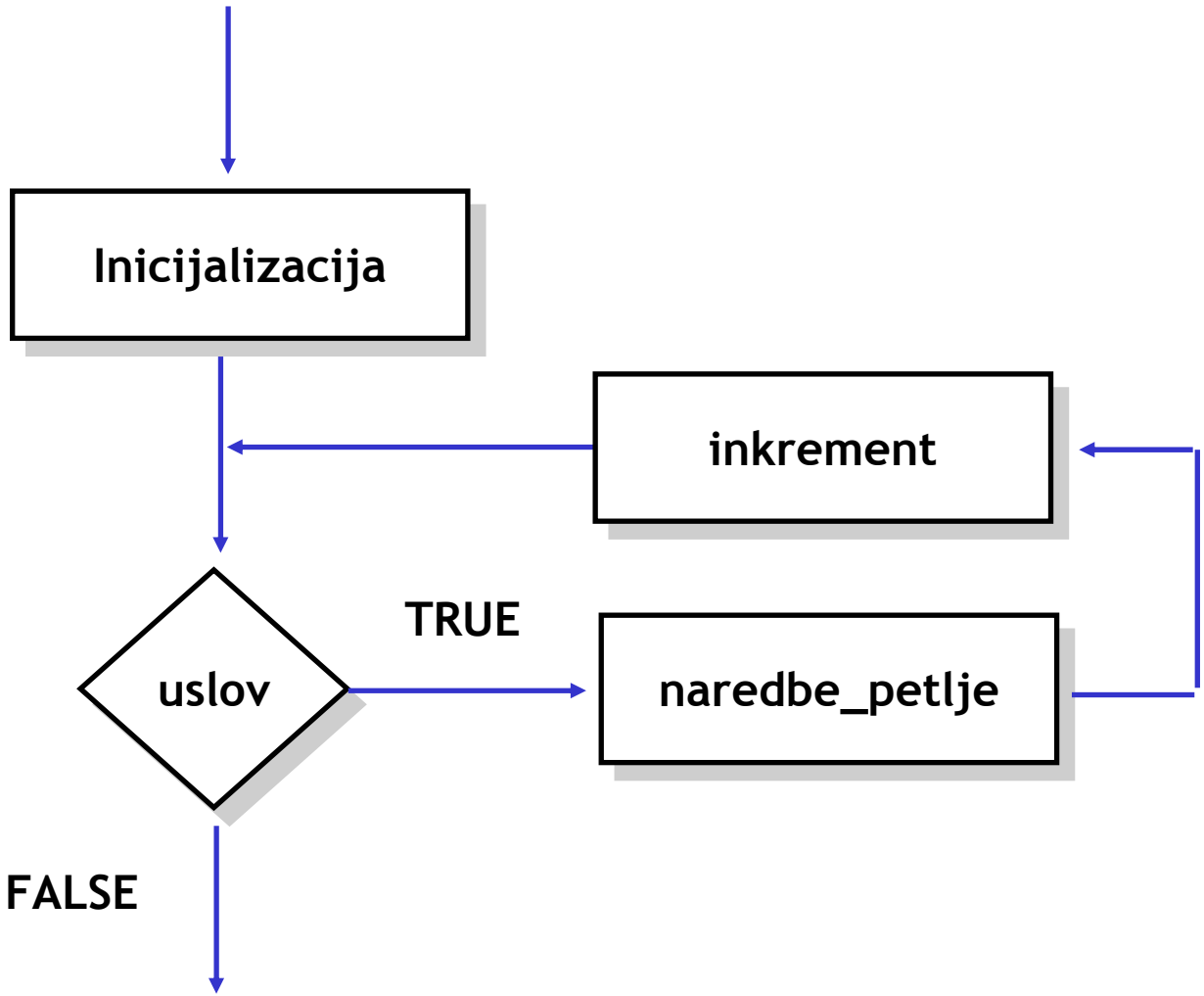


switch

```
switch (promenljiva)
 { case(vrednost1):
        uslovne_naredbe_1
    break;
   case(vrednost2):
        uslovne_naredbe_2
    break;
    ...
   default: uslovne_naredbe
 }
```

- **switch -** identifikuje promenljivu na osnovu koje program odlučuje gde dalje da ide
- **case -** početak svake grane
- **break -** obeležava kraj svake grane
- **default -** ako nijedna promenljiva (char, int, short, byte) naredbe switch ne odgovara ni jednoj vrednosti case

for petlja


Inicijalizacija
inkrement
uslov
TRUE
naredbe_petlje
FALSE

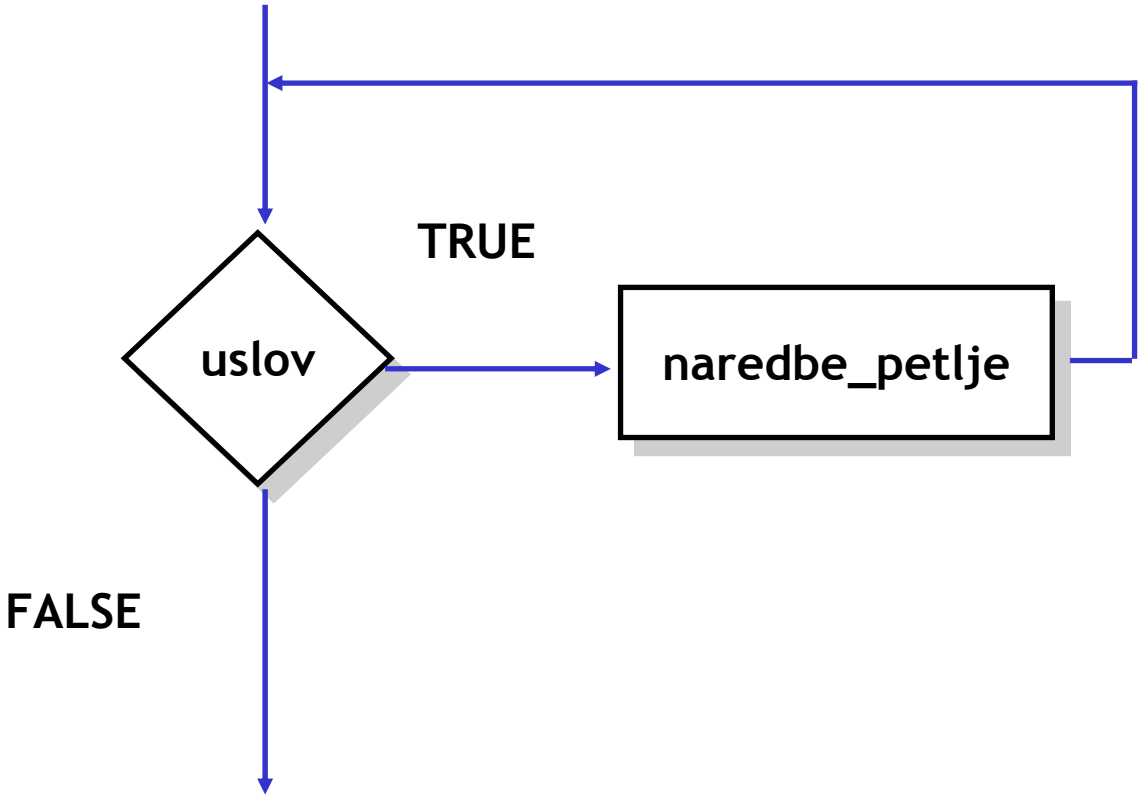
for petlja

```
for ( inicijalizacija ; izraz ; inkrement )
 {
   naredbe_petlje;
  }
```

- **Inicijalizacija -** postavlja početnu vrednost izraza
- **izraz -** testira se na početku svakog prolaza kroz petlju
- **Inkrement -** izvršava se na kraju svakog prolaza kroz petlju.

Java

while



while

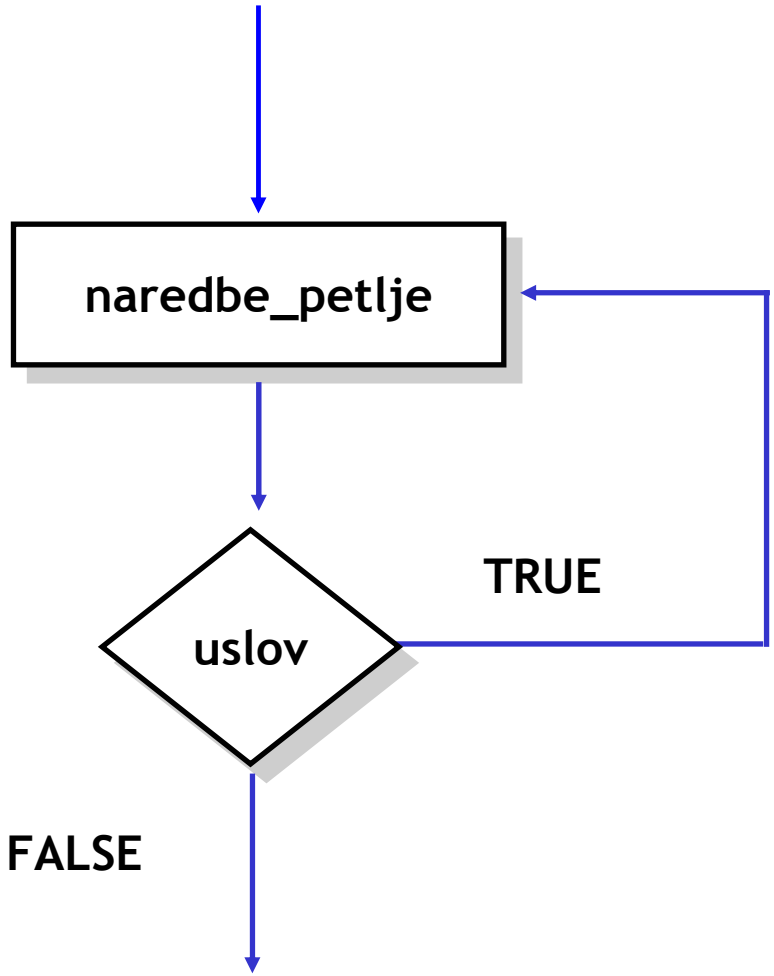

```
while( uslov )
 {
   naredbe_petlje;

  }
```

- **izraz -** testira se na početku svakog prolaza u petlju
- **naredbe_petlje -** izvršavaju se ako je uslov tačan.

Java

do/while



do/while

```
do
 {
   naredbe_petlje;

 } while (uslov)
```

- **naredbe_petlje -** izvršavaju se sigurno jedanput i svaki sledeći put za koji je uslov zadovoljen.
- **izraz -** testira se posle prvog a pre svakog novog ulaska u petlju

try/catch/finaly

```
try {
      // izvršne_naredbe_1;
    } catch (Neki_izuzetak e)
       {
          // naredbe_za_obradu;
       }
finally {
          // izvršne_naredbe_2;
       }
```

- **try** - grupa naredbi u kojima se može pojaviti izuzetak ili abnormalni prekid (break, continue)
- **catch** - Označava početak obrade izuzetaka koji je tipa Neki_izuzetak ili podklase tog tipa
- **finally** - Naredbe koje se izvršavaju bez obzira na to da li je došlo do pojave izuzetaka ili je normalno izvršena grupa naredbi u `try`