

UPORABA IKT V NARAVOSLOVJU IN TEHNIKI

Predavanje 9

Splet – programiranje v JavaScript

doc.dr. Mira Trebar

Vsebina

- Algoritem
 - JavaScript - uvod
 - Spremenljivke
 - Izrazi
 - Operatorji
 - Stavki
 - Objekti
 - Primeri
-
- Vir: <http://www.w3schools.com/>

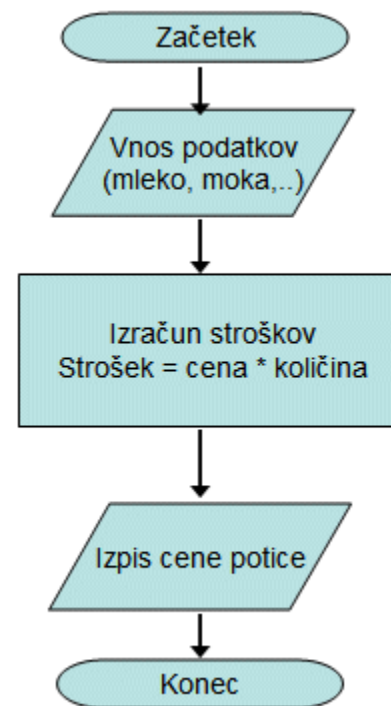
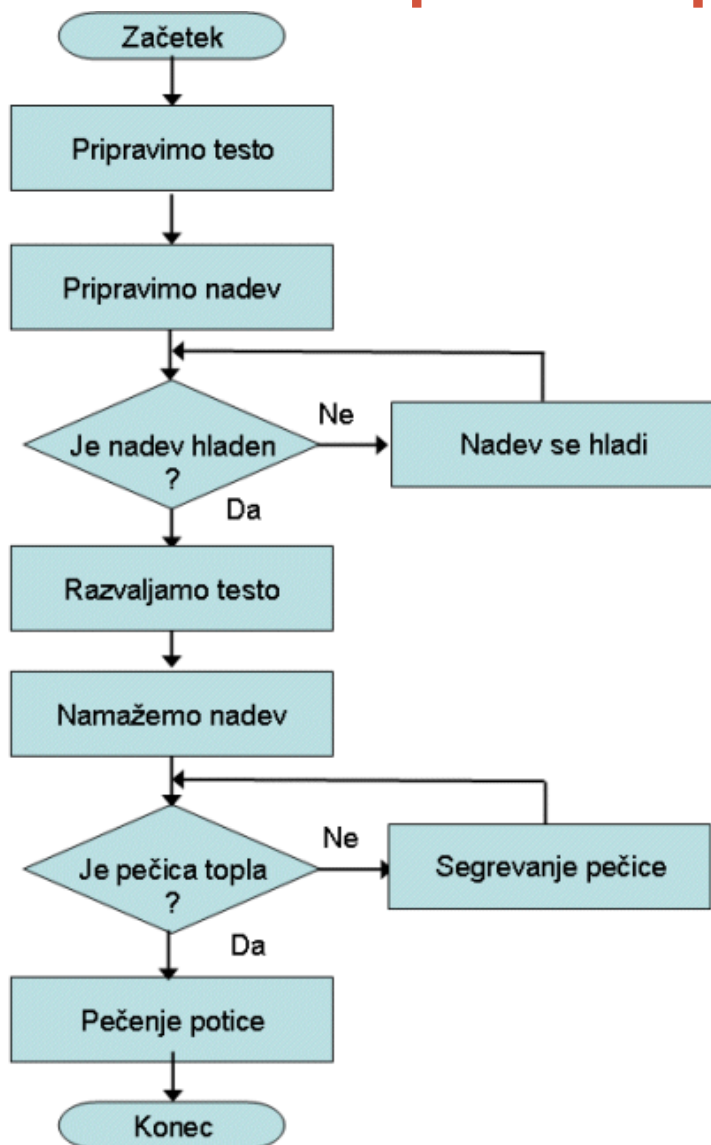
Uvod - problem

- Kako narediti jabolčni zavitek?
(Zaporedje slik z opisi)
- Kako narediti potico?
(Diagram poteka)
- Koliko stane potica?
(Program: Pyton, C++, ...)



Cena potice = ?

Uvod - Priprava potice, cena potice



Uvod - Algoritmi

- **Algoritem** je množica korakov, ki določajo izvedbo določene naloge
- Z algoritmi (postopki) se srečujemo vsak dan
 - pri kuhanju jih imenujemo recepti,
 - na pralnem stroju izberemo algoritem, po katerem nam opere oblačila,
 - pri iskanju cilja govorimo o napotkih,
 - pri igranju inštrumenta so to note,
 - računanje
 - seštevanje, odštevanje, množenje, deljenje
 - najmanjši skupni delitelj
 - ugotavljanje ali je število praštevilo ali ne
 - iskanje rešitev kvadratne enačbe

AJDOVI ŽGANCİ S KISLO REPO

ajdovi žganci s krompirjem in kuhana kislá repa

| | | |
|------------------------|-------|-----------|
| ZAHTEVNOST | | |
| RECEPT PRISPEVAL | | Pitončica |
| ŠTEVILKA RECEPTA | | 12376 |
| KOLIČINA | | 6 oseb |
| ČAS PRIPRAVE + KUHANJA | | |
| DRŽAVA IZVORA | | Slovenija |
| DATUM VNOSA | | 9.3.2009 |
| PRIKAZANO | | 3216-krat |
| OCENA | | |

Sestavine:

Za žgance:
30 dag krompirja
sol
50 dag ajdove moke

Za kisló repo:
50 dag kislé repe
1 manjša čebula
olje
moka



Postopek:

Krompir olupimo in ga narežemo na kolesca, debela cca. 3 - 5 mm. Solimo, dodamo vodo in kuhamo še 15 minut, ko že zavre. Vode mora biti dovolj, še čez krompir. Nato dodamo ajdovo moko - stresemo "kupček" čez krompir in na sredini naredimo luknjo do dna posode, da pride voda ven. Pokrijemo in kuhamo še 15 do 20 minut. Nato s kühalnico mečkamo, dokler moka ne vpije vse vode in se pomeša s krompirjem. Moka se ne sme več videti. Pred postrežbo žgance nadrobimo na manjše koščke.

Algoritem – definicija (1)

- Računalniška znanost – osnovni koncept je algoritem (Norman Gibbs in Allen Tucker, 1986)
- Računalniški strokovnjaki morajo razviti algoritme za reševanje problemov.
- **Algoritem** - postopek za reševanje matematičnega problema v končnem številu korakov, ki redno vključujejo ponavljanje določene operacije.
- Za izvedbo neke naloge je uporabljena metode "korak za korakom".
 - K 1: Nekaj izvedi
 - K 2: Nekaj izvedi
 - ...
 - K N: Konec postopka

Algoritem – definicija (2)

- Kategorije:

- **Zaporedne operacije:** zaporedni ukaz izvede eno natančno določeno nalogo. Ko je naloga končana, se algoritem pomakne na naslednjo operacijo:

Primer: Dodaj 1 skodelico moka v maso za testo.

Postavi vrednost x na 1.

- **Pogojne operacije:** ukazi tipa "postaviti vprašanje". Postavimo vprašanje in naslednji ukaz je izbran na osnovi odgovora.

Primer: Če je masa za testo preveč suha, dodaj pol.skodelice mleka.

Če je x različen od 0, potem postavi $y=1/x$,sicer izpiši sporočilo o napaki, da deljenje z 0 ni možno.

- **Ponavljajoče operacije:** ukazi tipa "zanka". Povedo, da ne gremo na naslednji ukaz, ampak se vrnemo nazaj in ponovimo izvajanje prejšnjega bloka ukazov.

Primer: Ponavljaj prejšnji dve operaciji, dokler ni masa za testo gladka.

Algoritmi – formalna definicija

- Algoritem je urejena množica izvršljivih in nedvoumnih korakov, ki definirajo ustavljen proces.
 - **urejena množica**
 - jasen mora biti vrstni red izvajanja korakov
 - ni potrebno, da se koraki izvršujejo eden za drugim
 - **izvršljivi koraki**
 - pomeni, da vsak korak lahko dejansko naredimo
 - **nedvoumni koraki**
 - imamo dovolj informacij o procesu, da popolnoma in na en sam način izvedemo potrebne akcije
 - izvajanje algoritma ne potrebuje kreativnosti, dovolj je sposobnost sledenja
 - **ustavljen proces**
 - algoritem se mora prej ali slej končati
 - kaj pa deljenje $1:3 = 0,3333\dots$



Algoritem – postopek seštevanja

Podano je število $m \geq 1$ in dve m -bitni pozitivni števili:

$A_{m-1}A_{m-2}\dots A_0$ in $B_{m-1}B_{m-2}\dots B_0$.

Izračun vsote:

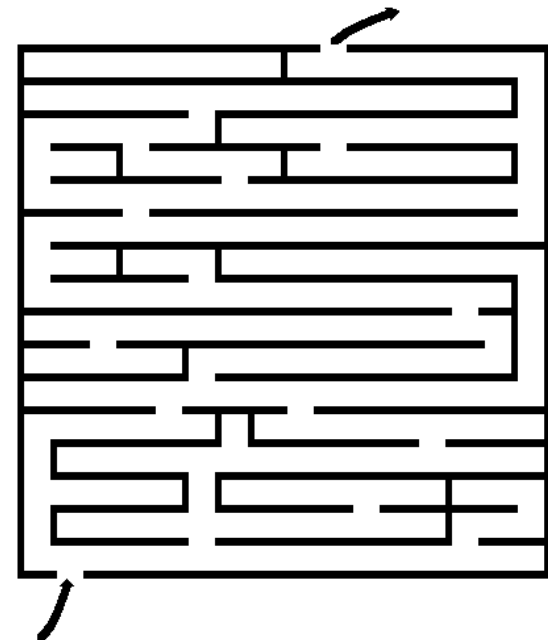
$$C_m C_{m-1} C_{m-2} \dots C_0 = A_{m-1} A_{m-2} \dots A_0 + B_{m-1} B_{m-2} \dots B_0$$

• Algoritem:

- K1: Postavi prenos = 0
- K2: Postavi indeks $i = 1$
- K3: Dokler je i manjši ali enak $m-1$ ponavljaj ukaze v K4 do K6
- K4: Prištej dva bita A_i in B_i k trenutni vrednosti prenos za izračun C_i
- K5: Če je $C_i \geq 10$, potem izračunaj $(C_i - 10)$, postavi prenos = 1, sicer pa je nova vrednost prenos = 0
- K6: Prištej 1 k i , (dejansko se pomaknemo eno mesto levo)
- K7: postavi $C_m = \text{prenos}$
- K8: Izpiši odgovor $C_m C_{m-1} C_{m-2} \dots C_0$
- K9: Stop

Algoritem - primeri

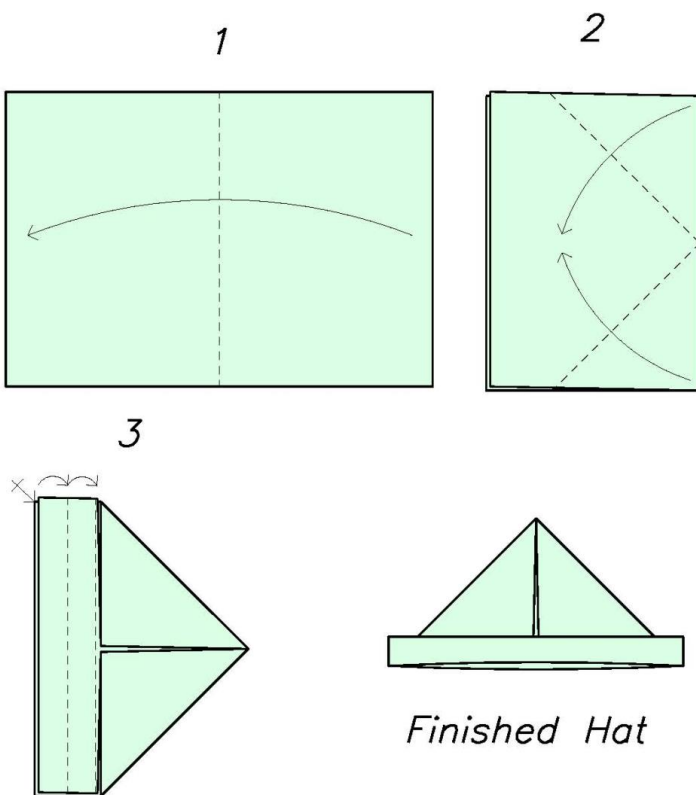
- Na predavanjih smo se srečali že z nekaj algoritmi:
 - delovanje centralne procesne enote (CPE)
 - jemanje ukaza iz pomnilnika,
 - dekodiranje in
 - izvrševanje ukaza.
 - postopek zagona ob vklopu računalnika
 - najprej se zažene zagonski program,
 - sledi prenos datotek operacijskega sistema iz diska v pomnilnik,
 - nazadnje se zažene operacijski sistem.
 - Navodila za uporabo programske opreme
 - Protokoli za komunikacijo med računalniki v omrežju



Algoritem - primer

- Origami

- algoritem
- nedvoumni koraki ???



- osnovni koraki (ukazi)
 - **sintaksa** – določa zapis in obliko ukazov
 - **semantika** – določa pomen ukazov

| sintaksa | semantika |
|----------|-------------------------|
| --- | pregib v dolino |
| -.-.- | pregib na hrib |
| ← | pregibanje proti nam |
| ↷ | pregibanje stran od nas |
| → | pritisk, stisk |
| ← + | ponavljanje korakov |
| ↻ | obračanje modela |

Algoritmi v računalništvu

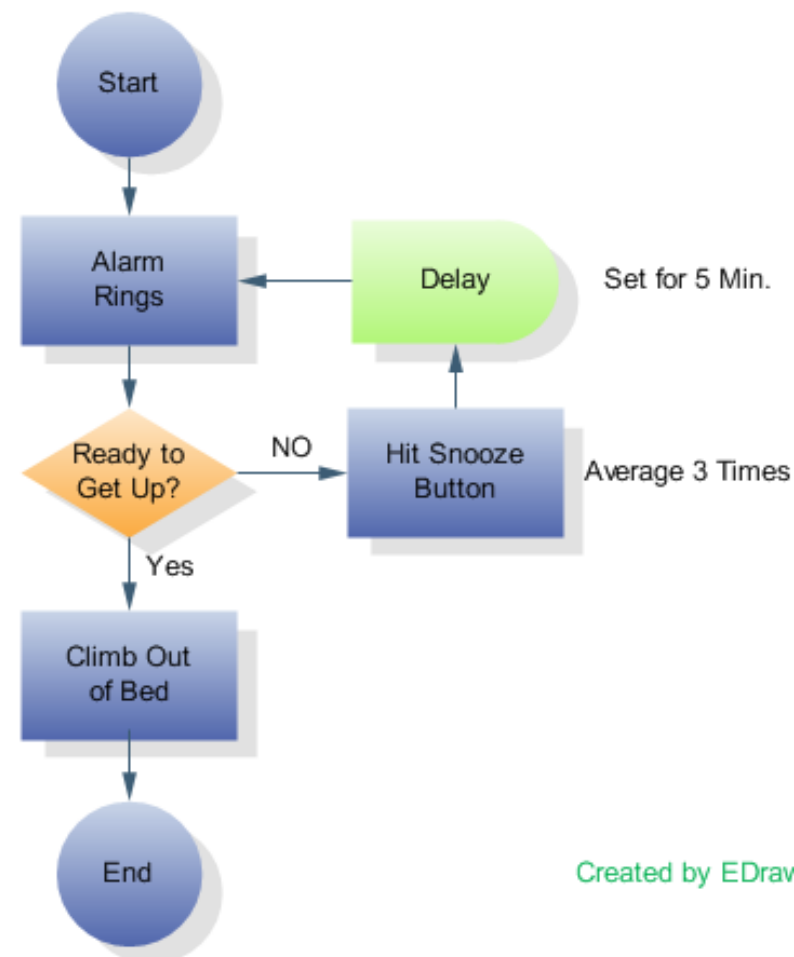
- Opisovanje algoritmov v računalništvu
 - diagram poteka
 - psevdo koda

1. ura zveni
2. dokler ne moreš vstati, ponavljaj
 - a. pritisni na gumb za dremež
 - b. čakaj
 - c. ura zveni
3. zlezi iz postelje

Primer programa:

```

pripravljenVstati = uraZvoni()
while pripravljenVstati != true:
    pritisniGumbDremez()
    cakaj(5)
    pripravljenVstati = uraZvoni()
zleziIzPostelje()
  
```



JavaScript - uvod

- Skriptni jezik:
 - Je vključen v dokument HTML in se interpretira
 - Ukazi se izvajajo brez predhodnega prevajanja
- Razlikovanje črk ('case sensitive')
- Omogoča interaktivnost spletnih strani
- Uporaba:
 - Dodajanje programskih funkcionalnosti
 - Odziva se na dogodke
 - Branje in spreminjanje vsebine elementov HTML
 - Potrditev obrazcev
 - Generiranje piškotkov ('cookies')
 - Komunikacija s strežnikom



JavaScript – definicija

- Program je del kode HTML:

```
<script>
```

```
    document.write(„Pozdravljeni!“)
```

```
</script>
```

```
<noscript>
```

```
    Omogoči JavaScript, da bo stran v celoti vidna!
```

```
</noscript>
```

- Program je zapisan v samostojni datoteki z imenom *test.js*

```
<script src="test.js">
```

```
</script>
```

Spremenljivke

- ni deklarirana,
- nedoločena spremenljivka ('undefined'):
 $y=x$; x ni definirana
- številski tip (ang. number) - (64-bitna plavajoča vejica (standard IEEE 754)
 $y = x - 2$; - ni število - **NaN** - (ang. Not a Number)
 $a = 1/0$ - neskončno (**Infinity**)
 $y = 10$; - število
- Boolov tip :
 $x = \text{true}$; - x =pravilno,
 $y = \text{false}$; - y =napačno
- znakovni niz (ang. char):
 - $y = "a"$; " " - dvojne navednice ; ' ' - enojne navednice

Spremenljivke - primeri

- HTML in JavaScript

```

<html>
<head>
<title>JavaScript-spremenljivke</title>
</head>
<body>
<script>
  var x,y;
  y=x; document.write("<br /> y=" + y);
  y=x-2; document.write("<br /> y=" + y);
  a=1/0; document.write("<br /> a=" + a);
  y=10; document.write("<br /> y=" + y);
  x=true; document.write("<br /> x=" + x);
  y="a"; document.write("<br />y= " + y);

  document.write(z);
  var z="Dober dan";
  document.write("<br /> z=" + z);
</script>
</body></html>

```

Rezultati - Izpisi

```

y=undefined
y=NaN
a=Infinity
y=10
x=true
y= a

```

???

Izrazi

- Komentar:
 // opis ali /* opis */
- Ukazi/Stavki/Izrazi:
 - `var a, B;` - spremenljivka
 - `a=B+3;` - priredilni operator
 - ...
- **Operand** - ena ali več vhodnih vrednosti.
- **Operator** – znak, skupina znakov za označevanje operacije.
- **Izraz** je kombinacija operandov in operatorjev.
- Prednost in vrstni red izvajanja: operacije, operatorji, oklepaji
 - $a+b*c-d$ ali $(a+b)*(c-d)$

Operatorji

- Aritmetični operatorji (številski tip):
a + b, a - b, a * b, a / b, a % b, ++, --
- Primerjalni operatorji (Boolov tip)
==, ===, !=, !==, <, >, <=, >=
- Logični operatorji (Boolov tip)
&& - and, || - or, ! - not
- Operatorji z znakovnimi nizi:
vsi primerjalni operatorji, in konkatencija (+)
- Priredilni in kombinirani operatorji
=, +=, -=, *=, /=, %=
- Posebni operatorji
typeof() – določa tip operanda (spremenljivka, rezultat operacije)

Operatorji - primeri

1. Število in Boolova spremenljivka

`4 + false = 4` in `4 + true = 5`

2. Število in znak

Seštevanje: `4 + "5" = 45`

Množenje: `1 * "5" = 5`

3. Operator + pomeni konkatencijo

```
<script>
```

```
var x=10, y=20;
```

```
document.write("<br />" + x-y);
```

```
document.write("<br />" + (x-y));
```

```
var z=x-y;
```

```
document.write("<br />" + z);
```

```
</script>
```

4. Primerjalni operatorji

```
<script>
```

```
var a=3,b="3";
```

```
document.write(a==b);
```

```
document.write(a===b);
```

```
<script>
```

3) NaN

-10

-10

4) true

false

JavaScript – stavki (1)

Tipi:

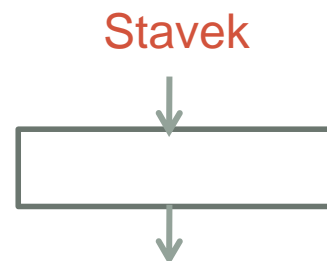
- priredilni stavek (`a=b;`)
- prazen stavek (`;`)
- deklaracija (`var a;`)
- sestavljeni stavek

Ločevanje med stavki:

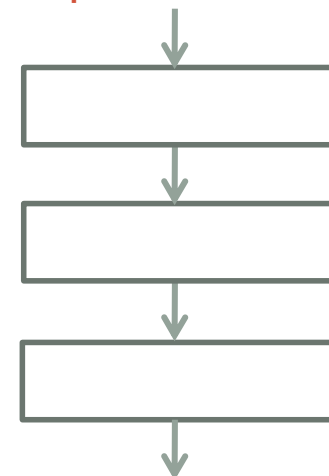
- stavek v eni vrstici
- podpičje na koncu stavka
- zaviti oklepaj in zaklepaj za sestavljeni stavek

Diagram poteka:

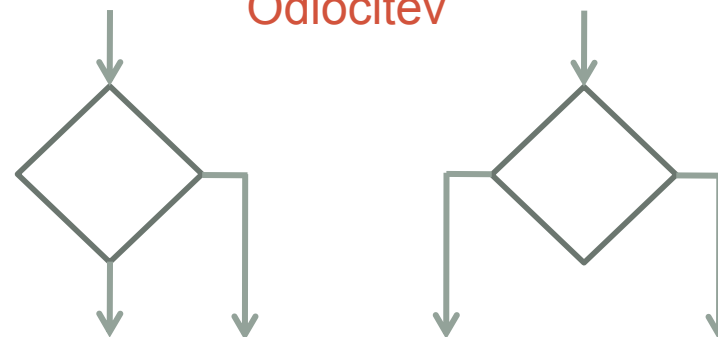
- pravokotnik,
- romb,
- puščica



Zaporedni stavki



Odločitev



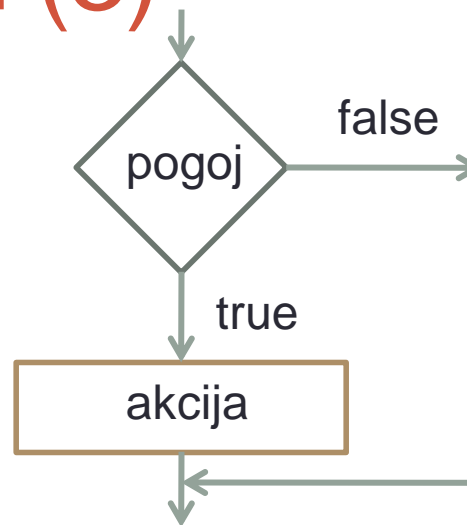
JavaScript – stavki (2)

- Pogojni
 - if (pogoj) akcija;
 - If (pogoj) akcija else alternativa;
- Ponavljalni
 - for (inicijalizacija; pogoj; šteje) izvajaj;
 - while (pogoj) akcija;
 - do akcija while (pogoj);
- Drugi stavki
 - break;
 - continue;
- Vnos podatka - objekt:
 - prompt("Vnesite število:");

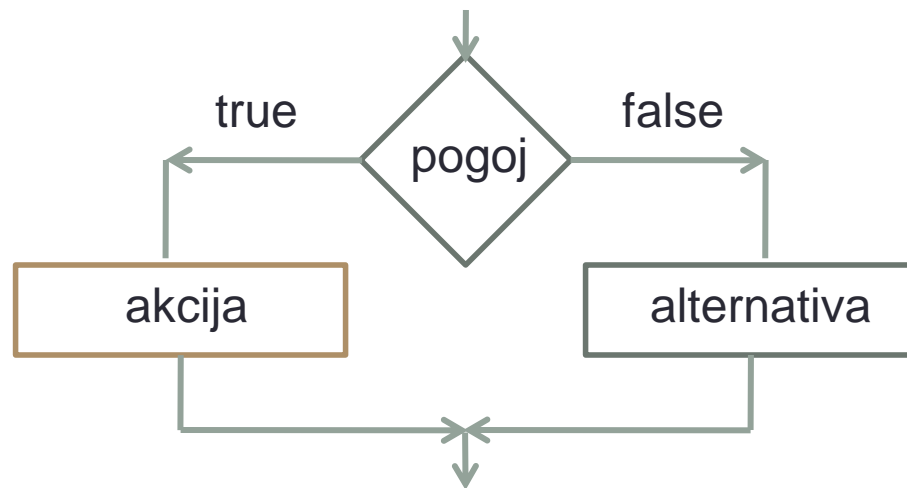
JavaScript – stavki (3)

- Pogojni stavek

```
if (pogoj)  
    akcija;
```



```
if (pogoj)  
    akcija;  
else  
    alternativa;
```



JavaScript – stavki (4)

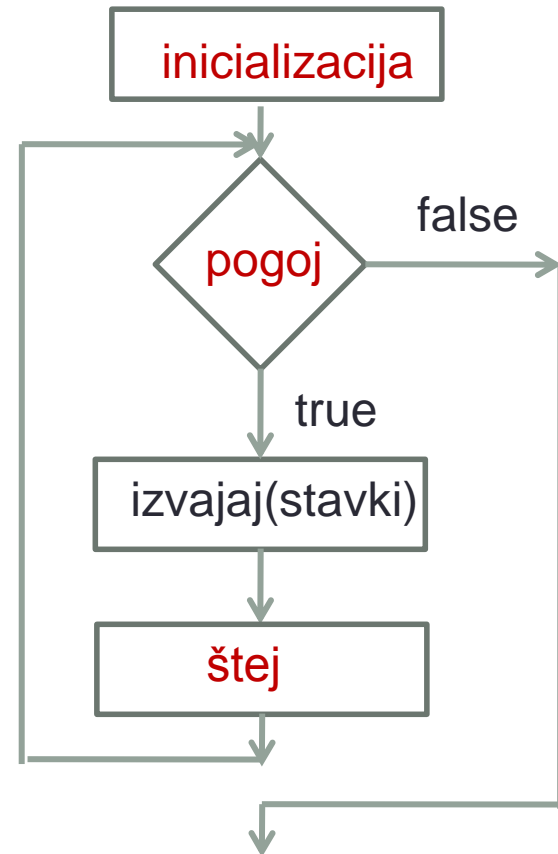
- Ponavljalni stavek (for)

```
for (inicijalizacija; pogoj;  
    štej)  
    izvajaj;
```

Primer:

```
var a=5, b=1;  
for (i=1; i<= a; i++)  
    b=b+1;
```

Rezultat: b=6



JavaScript – stavki (5)

- Zaustavitev izvajanja v ponavljalnem stavku:
break;

```
for (i = 1; i <= 5; i++)  
    if (i==3) break;
```

Primer: Poštevanka

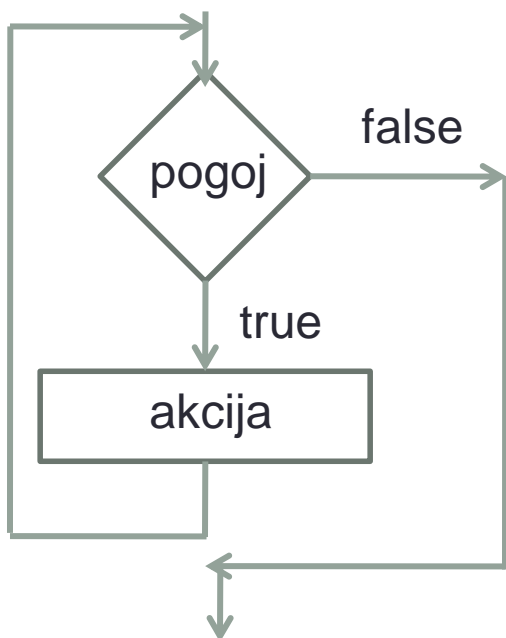
```
var i, j;  
for (j = 1; j <=5; j++) {  
    for (i = 1; i <= 5; i++)  
        document.write(" " + (i * j) );  
    document.write("<br />");  
}
```

| | | | | |
|---|----|----|----|----|
| 1 | 2 | 3 | 4 | 5 |
| 2 | 4 | 6 | 8 | 10 |
| 3 | 6 | 9 | 12 | 15 |
| 4 | 8 | 12 | 16 | 20 |
| 5 | 10 | 15 | 20 | 25 |

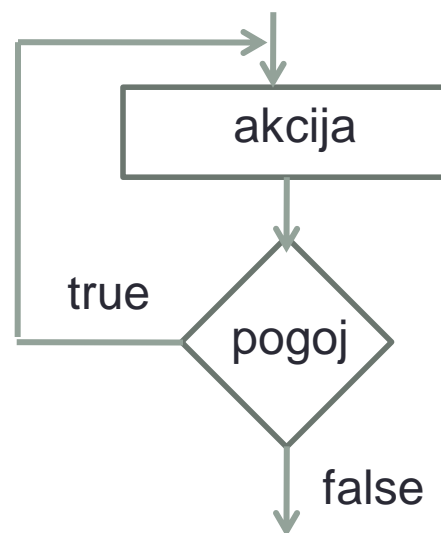
JavaScript – stavki (6)

- Ponavljalni stavek (while)

while (pogoj)
akcija;



do
akcija;
while (pogoj);



JavaScript – objekti (1)

- **Objekt** – posebna vrsta podatkov, ki ima:
 - Lastnosti
 - Postopki (metode)
- Zapis objekta:
 - **imeObjekta.imeLastnosti**
 - **imeObjekta.imePostopka(seznamParametrov)**
- Objekti brskalnika
 - **objekt window** - !!! Ni obvezno !!!!
 - lastnosti: document
 - postopki: alert() , confirm() , prompt()
 - **objekt document**
(window.document == document)

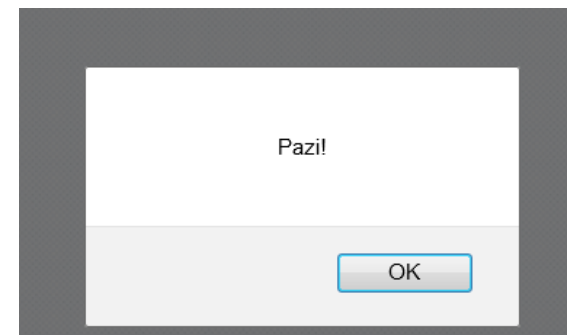
JavaScript – objekti (2)

```

<script>
window.alert("Pazi!"); // se prikažeta eden za drugim
alert("Glej!");

var a, b;
a=window.prompt("Vnesi a");
b=prompt("Vpiši b");
window.document.write("<br /> b=" + b);
document.write("<br /> b=" + b);
</script>

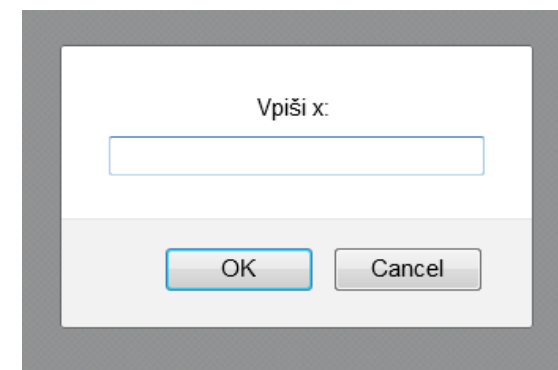
```



```

<script>
var vnos, vsota=0;
do
{
  vnos = prompt("Vnesi vrednost ( ničla konča )");
  vsota += vnos;
} while (vnos!=0);
document.write("<br /> Vsota je " + vsota);
</script>

```



JavaScript – objekti (3)

- Tabela

| A | B | SUMA |
|----|----|------|
| 10 | 20 | 30 |
| 35 | 32 | 67 |

```
<body>
<table border="1" cellpadding="5">
<tr align="right" bgcolor="yellow">
<th>A</th>
<th>B</th>
<th>SUMA</th>
</tr>
<script>
  var v_1,o_1;
  var v_2,o_2;
  var vr_1,vr_2;

  v_1=10;
  v_2=35;
  o_1=20;
  o_2=32;
  document.write("<tr><td>" + v_1 + "</td><td>" + o_1 + "</td>");
  vr_1= v_1 + o_1;
  document.write("<td>" + vr_1 + "</td></tr>");

  document.write("<tr><td>" + v_2 + "</td><td>" + o_2 + "</td>");
  vr_2= v_2 + o_2;
  document.write("<td>" + vr_2 + "</td></tr>");
</script>
</table>
</body>
```

JavaScript – primer (1)

- Največji skupni delitelj dveh števil:
 - Vnos števila x in y
 - Odštevanje manjšega števila od večjega števila, dokler sta števili različni.
 - Uporaba ponavljalnega stavka:
 - `while (pogoj)`
 - `akcija;`
 - Prekinitev trenutnega izvajanja - **stavek** *continue*;
 - Izpis rezultata
- Naloga:
 - Diagram poteka
 - Rešitev v HTML in JavaScript

JavaScript – primer (2)

- Izpis deliteljev števila n .
 - Primer: $n=20$ - izpis: 2 4 5 10
 - Rešitev: število n delimo z $i=2,3, \dots, n-1$,
če je ostanek $=0$, potem je i delitelj števila
 - Izpis rezultata
- Naloga:
 - Diagram poteka
 - Rešitev v HTML in JavaScript