Izvještaj iz laboratorijske vježbe 1

Uvod u programski jezik Julia

Bakir Činjarević
19705

21. oktobar 2025.

Sadržaj

1 Uvod

Ova laboratorijska vježba ima za cilj upoznavanje sa osnovama programskog jezika Julia, koji je namijenjen numeričkim i naučnim izračunavanjima. Julia kombinuje brzinu izvršavanja sa jednostavnošću sintakse, što je čini idealnom za operaciona istraživanja.

2 Osnovne funkcije

2.1 Zadatak 1

Postavka: Izračunati vrijednosti izraza:

2.1.1 Zadatak 1a

$$3 \cdot \frac{456}{23} + 31.54 + 2^6$$

Rješenje:

```
rezultat_1a = 3 * (456/23) + 31.54 + 2^6
println("1a: ", rezultat_1a)
```

Objašnjenje: Direktno se koriste aritmetički operatori za množenje (*), dijeljenje (/), sabiranje (+) i stepenovanje (^). Julia automatski izvršava operacije po prioritetu.

2.1.2 Zadatak 1b

$$\sin\left(\frac{\pi}{7}\right) \cdot e^{0.3} \cdot (2 + 0.9i)$$

Rješenje:

```
rezultat_1b = sin(pi/7) * exp(0.3) * (2 + 0.9im)
println("1b: ", rezultat_1b)
```

Objašnjenje: Koriste se ugrađene funkcije `sin()` i `exp()` za sinus i eksponencijalnu funkciju. Konstanta π je ugrađena u Juliji. Imaginarni broj se definiše sa `im`.

2.1.3 Zadatak 1c

$$\sqrt{2} \cdot \ln 10$$

Rješenje:

```
rezultat_1c = sqrt(2) * log(10)
println("1c: ", rezultat_1c)
```

Objašnjenje: Funkcija `sqrt()` računa kvadratni korijen, a `log()` prirodni logaritam.

2.1.4 Zadatak 1d

$$\frac{5 + 3i}{1.2 + 4.5i}$$

Rješenje:

```
rezultat_1d = (5 + 3im) / (1.2 + 4.5im)
println("1d: ", rezultat_1d)
```

Objašnjenje: Julia podržava operacije sa kompleksnim brojevima nativno.

2.2 Zadatak 2

Postavka: Dodijeliti varijablama a, b, c i d respektivno vrijednosti:

$$a = \arctan(5) + e^{5.6}, \quad b = \frac{1}{5}\sqrt{\sin\frac{\pi}{3}}, \quad c = \frac{\ln 15 + 1}{23}, \quad d = \sin\frac{\pi}{2} + \cos\pi$$

Rješenje:

```
a = atan(5) + exp(5.6)
b = (1/5) * sqrt(sin(pi/3))
c = (log(15) + 1) / 23
d = sin(pi/2) + cos(pi)

println("a = ", a)
println("b = ", b)
println("c = ", c)
println("d = ", d)
```

Objašnjenje: Koriste se funkcije `atan()`, `exp()`, `sqrt()`, `sin()`, `log()`, `cos()`.

2.2.1 Zadatak 2a

$$(a+b)^c$$

Rješenje:

```
rezultat_2a = (a + b)^c
println("2a: ", rezultat_2a)
```

2.2.2 Zadatak 2b

$$\arccos(b) \cdot \arcsin\left(\frac{c}{11}\right)$$

Rješenje:

```
rezultat_2b = acos(b) * asin(c/11)
println("2b: ", rezultat_2b)
```

2.2.3 Zadatak 2c

$$\sqrt[4]{\frac{a-b}{d}}$$

Rješenje:

```
rezultat_2c = ((a - b) / d)^(1/4)
println("2c: ", rezultat_2c)
```

2.2.4 Zadatak 2d

$$\frac{a}{\sqrt{c}} + \frac{bi}{3+2i}$$

Rješenje:

```
rezultat_2d = a / sqrt(c) + (b * im) / (3 + 2im)
println("2d: ", rezultat_2d)
```

3 Rad sa matricama

Za slijedeće zadatke je potreban paket `LinearAlgebra`:

```
using LinearAlgebra
```

3.1 Zadatak 3

Postavka: Varijabli A dodijeliti matricu:

$$A = \begin{pmatrix} 1 & -4i & \sqrt{2} \\ \ln(-1) & \sin\frac{\pi}{2} & \cos\frac{\pi}{3} \\ \arcsin(0.5) & \arccos(0.8) & e^{0.8} \end{pmatrix}$$

Rješenje:

```
A = [1 -4im sqrt(2);
     log(-1) sin(pi/2) cos(pi/3);
     asin(0.5) acos(0.8) exp(0.8)]
```

Objašnjenje: Matrica se definiše pomoću uglastih zagrada, elementi se odvajaju razmacima, a redovi tačkom-zarezom.

3.1.1 Zadatak 3a - Transponovana matrica

```
A_transpozovana = transpose(A)
# ili A'
println("Transponovana matrica:")
println(A_transpozovana)
```

3.1.2 Zadatak 3b - Zbir matrice A i njene transpozovane

```
zbir = A + transpose(A)
println("Zbir: ", zbir)
```

3.1.3 Zadatak 3c - Proizvod matrice A i njene transpozovane

```
proizvod_1 = A * transpose(A)
println("A * A': ", proizvod_1)
```

3.1.4 Zadatak 3d - Proizvod transpozovane i matrice A

```
proizvod_2 = transpose(A) * A
println("A' * A: ", proizvod_2)
```

3.1.5 Zadatak 3e - Determinanta

```
det_A = det(A)
println("Determinanta: ", det_A)
```

3.1.6 Zadatak 3f - Inverzna matrica

```
inv_A = inv(A)
println("Inverzna matrica: ", inv_A)
```

3.2 Zadatak 4

Postavka: Generisati različite tipove matrica.

3.2.1 Zadatak 4a - Matrica nula

```
matrica_nula = zeros(8, 9)
```

3.2.2 Zadatak 4b - Matrica jedinica

```
matrica_jedinica = ones(7, 5)
```

3.2.3 Zadatak 4c - Jedinična matrica (identitet)

```
jedinicna = Matrix(I, 5, 5)
# ili
jedinicna = I(5)
```

3.2.4 Zadatak 4d - Nasumična matrica

```
nasumicna = rand(4, 9)
```

Objašnjenje: Julia ima ugrađene funkcije za generisanje specijalnih matrica: `zeros()`, `ones()`, `I` (identitet) i `rand()` za nasumične vrijednosti.

3.3 Zadatak 5

Postavka: Za datu matricu izračunati zbir, minimum i maksimum po redovima, kolonama i dijagonalama:

$$a = \begin{pmatrix} 2 & 7 & 6 \\ 9 & 5 & 1 \\ 4 & 3 & 8 \end{pmatrix}$$

Rješenje:

```
a = [2 7 6; 9 5 1; 4 3 8]

# Zbir po redovima
zbir_redovi = sum(a, dims=2)

# Zbir po kolonama
zbir_kolone = sum(a, dims=1)

```

```
# Zbir glavne dijagonale
zbir_glavna_dijag = sum(diag(a))

# Zbir sporedne dijagonale
zbir_sporedna_dijag = sum(diag(reverse(a, dims=2)))

# Minimum i maksimum po redovima
min_redovi = minimum(a, dims=2)
max_redovi = maximum(a, dims=2)

# Minimum i maksimum po kolonama
min_kolone = minimum(a, dims=1)
max_kolone = maximum(a, dims=1)

# Minimum i maksimum dijagonala
min_glavna = minimum(diag(a))
max_glavna = maximum(diag(a))
```

Objašnjenje: Parametar `dims=1` znači operacija po kolonama, `dims=2` po redovima. Funkcija `diag()` izdvaja dijagonalu matrice.

3.4 Zadatak 6

Postavka: Neka je:

$$a = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}, \quad b = \begin{pmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \end{pmatrix}$$

```
a = [1 2 3; 4 5 6; 7 8 9]
b = [1 1 1; 2 2 2; 3 3 3]
```

3.4.1 Zadatak 6a - Sinus svakog elementa

```
c = sin.(a)
```

Objašnjenje: Operator tačka (.) omogućava element-wise operacije (broadcasting).

3.4.2 Zadatak 6b - sin(a) .* cos(b)

```
c = sin.(a) .* cos.(b)
```

3.4.3 Zadatak 6c - Treći korijen matrice

```
c = a^(1/3)  # Matri ni tre i korijen
```

3.4.4 Zadatak 6d - Treći korijen svakog elementa

```
c = a.^(1/3)  # Element-wise tre i korijen
```

3.5 Zadatak 7

Postavka: Formirati vektore.

3.5.1 Zadatak 7a

```
vektor_a = 0:99
# ili
vektor_a = collect(0:99)
```

3.5.2 Zadatak 7b

```
vektor_b = 0:0.01:0.99
```

3.5.3 Zadatak 7c

```
vektor_c = 39:-2:1
# Za vektor kolonu:
vektor_c_kolona = collect(39:-2:1)
vektor_c_kolona = reshape(vektor_c_kolona, :, 1)
```

Objašnjenje: Sintaksa `start:step:stop` kreira raspon. Za kolonu koristimo `reshape()`.

3.6 Zadatak 8

Postavka: Formirati zadatu matricu 8x8 sa blokovima.

Rješenje:

```
# Blok 7-ica (4x4) i nula (4x4)
gornji_red = [7*ones(4,4) zeros(4,4)]
# Blok 3-ica (4x8)
donji_red = 3*ones(4,8)
# Kombinovanje
a = vcat(gornji_red, donji_red)
```

3.6.1 Zadatak 8a

```
b = a + I(8)
```

3.6.2 Zadatak 8b - Svaki drugi red

```
c = b[1:2:end, :]
```

3.6.3 Zadatak 8c - Svaka druga kolona

```
d = b[:, 1:2:end]
```

3.6.4 Zadatak 8d - Svaki drugi red i kolona

```
e = b[1:2:end, 1:2:end]
```

Objašnjenje: Sintaksa `start:step:end` indeksira elemente. `:` znači svi elementi.

4 Funkcije za crtanje

Za crtanje je potreban paket `Plots`:

```
using Plots
```

4.1 Zadatak 1 - Crtanje grafika jedne promjenljive

4.1.1 Zadatak 1a i 1b

```
x = range(-pi, pi, length=101)
y_sin = sin.(x)
y_cos = cos.(x)

plot(x, y_sin, label="sin(x)")
plot!(x, y_cos, label="cos(x)")
```

4.1.2 Zadatak 1c i 1d

```
x = range(1, 10, length=101)
y1 = sin.(1 ./ x)
y2 = cos.(1 ./ x)

plot(x, y1, color=:black, label="sin(1/x)")
plot!(x, y2, marker=:circle, color=:blue,
      label="cos(1/x)", linestyle=:none)
```

Objašnjenje: Funkcija `plot()` crta prvi grafik, `plot!()` dodaje na postojeći grafik.

4.2 Zadatak 2 - 3D grafik

```
x = -8:0.5:8
y = -8:0.5:8
z = [sin(sqrt(xi^2 + yi^2)) for xi in x, yi in y]

surface(x, y, z, xlabel="x", ylabel="y",
        zlabel="z", title="z = sin(sqrt(x^2 + y^2))")
```

Objašnjenje: Funkcija `surface()` crta 3D površinu.

5 Funkcije i metaprogramiranje

5.1 Zadatak 1

Postavka: Funkcija za sabiranje i oduzimanje sa provjerama.

Rješenje:

```
function sabiranje_oduzimanje(arg1=0, arg2=0)
    # Provjera dimenzija ako su matrice
    if isa(arg1, Array) && isa(arg2, Array)
        if size(arg1) != size(arg2)
            return 0, 0
        end
    end

    zbir = arg1 + arg2
    razlika = arg1 - arg2

    return zbir, razlika
end

# Testiranje
rez1, rez2 = sabiranje_oduzimanje(5, 3)
println("Zbir: ", rez1, ", Razlika: ", rez2)
```

5.2 Zadatak 2

Postavka: Funkcija za sabiranje elemenata matrice.

Rješenje:

```
function suma_matrice(mat)
    m, n = size(mat)

    # Ukupna suma
    ukupna_suma = 0
    for i in 1:m
        for j in 1:n
            ukupna_suma += mat[i,j]
        end
    end

    # Suma po redovima
    suma_redovi = zeros(m)
    for i in 1:m
        for j in 1:n
            suma_redovi[i] += mat[i,j]
        end
    end

    # Suma po kolonama
    suma_kolone = zeros(n)
    for j in 1:n
```

```
        for i in 1:m
            suma_kolone[j] += mat[i,j]
        end
    end

    # Suma glavne dijagonale
    suma_glavna = 0
    for i in 1:min(m,n)
        suma_glavna += mat[i,i]
    end

    # Suma sporedne dijagonale
    suma_sporedna = 0
    for i in 1:min(m,n)
        suma_sporedna += mat[i, n-i+1]
    end

    return ukupna_suma, suma_redovi, suma_kolone,
           suma_glavna, suma_sporedna
end
```

5.3 Zadatak 3

Postavka: Funkcija za crtanje proizvoljne funkcije iz stringa.

Rješenje:

```
function crtaj_funkciju(izraz_string)
    x = range(-5, 5, length=100)

    # Evaluacija izraza
    f = eval(Meta.parse("x -> " * izraz_string))
    y = f.(x)

    plot(x, y, label=izraz_string,
         xlabel="x", ylabel="y")
end

# Primjer kori tenja
crtaj_funkciju("sin(x)")
crtaj_funkciju("x^2 + 2*x + 1")
```

Objašnjenje: `Meta.parse()` pretvara string u izraz, a `eval()` ga evaluira.

6 Zaključak

U ovoj vježbi smo se upoznali sa osnovama programskog jezika Julia, uključujući:

- Osnovne aritmetičke operacije i matematičke funkcije
- Rad sa matricama i vektorima

- Linearno-algebarske operacije
- Crtanje 2D i 3D grafika
- Pisanje funkcija i korištenje metaprogramiranja

Julia pokazuje veliku fleksibilnost i brzinu u numeričkim računanjima, što je čini pogodnom za operaciona istraživanja.