8. Numerička integracija

Data je funkcija:

$$f(x) = \sin x$$

1. Naći vrednost određenog integrala funkcije $f(x)$ na intervalu $x \in \left[0, \frac{3\pi}{2}\right]$:

```
f = @(x) sin(x);
a = 0;
b = 3*pi/2;

I = integral(f, a, b)
```

Rezultat:

```
I =
    1.0000
```

1. Trapezna metoda

Zadatak 1. Napisati trapeznu metodu za izračunavanje vrednosti određenog integrala proizvoljne funkcije nad proizvoljnim intervalom.
Pokušati prvo ručno jednu iteraciju trapezne metode nad funkcijom $f(x) = \sin x$ na intervalu $x \in \left[0, \frac{3\pi}{2}\right]$:

```
f = @(x) sin(x);
a = 0;
b = 3*pi/2;
```

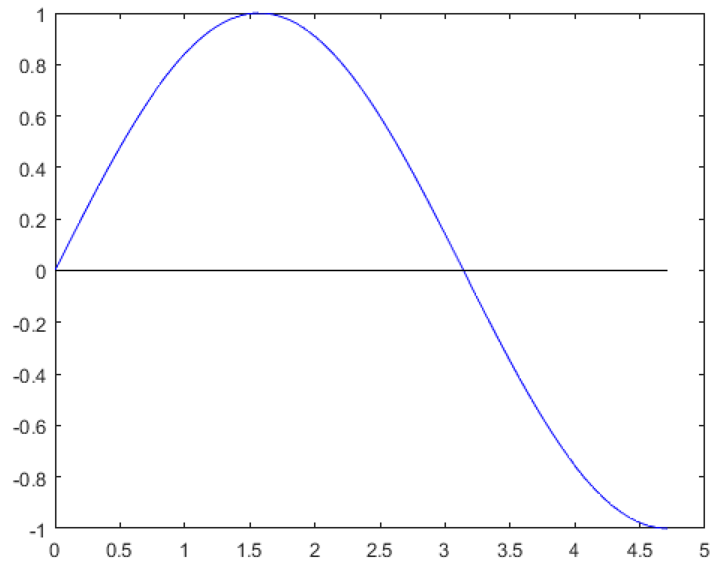
1. Definisati broj podintervala:

```
intervals = 4;
```

2. Nacrtati funkciju i x-osu nad intervalom i zadržati grafik:

```
x = linspace(a, b, 100);
fX = f(x);
plot(x, fX, 'blue', [a b], [0 0], 'black'), hold on
```

Rezultat:



Slika 1. Grafik funkcije

3. Izračunati širinu podintervala, na osnovu nje naći vrednosti nezavisno promenljive x u krajevima podintervala, a na osnovu njih naći vrednosti funkcije $f(x)$ u krajevima intervala:

```
width = (b - a)/intervals;
x = a:width:b;
fX = f(x);
```

4. Postaviti vrednost integralne sume na 0:

```
I = 0
```

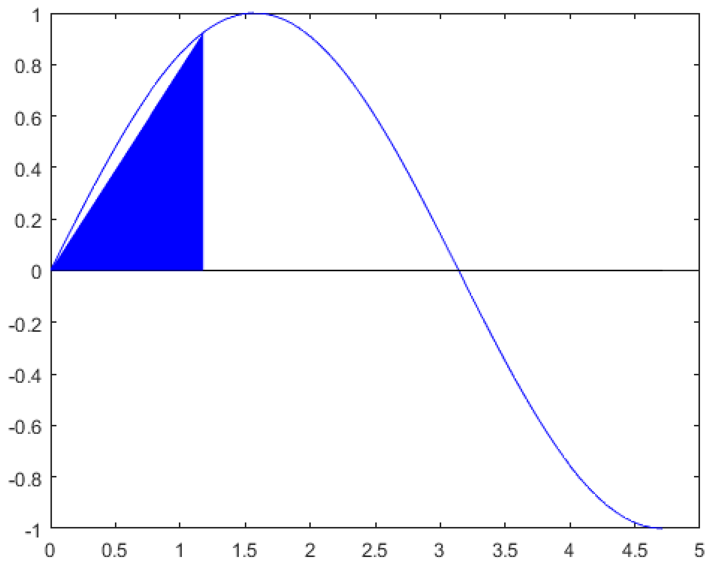
Rezultat:

```
I =
     0
```

5. Na osnovu vrednosti funkcije u krajevima 1. podintervala i nacrtati trapeznu površ između x-ose i funkcije u 1. podintervalu:

```
x1 = x(1);
x2 = x(2);
fX1 = fX(1);
fX2 = fX(2);
area([x1 x2], [fX1 fX2], 'FaceColor', 'blue', 'LineStyle', 'none')
```

Rezultat:



Slika 2. 1. podinterval

6. Naći vrednost nacrtane površine i dodati je na integralnu sumu:

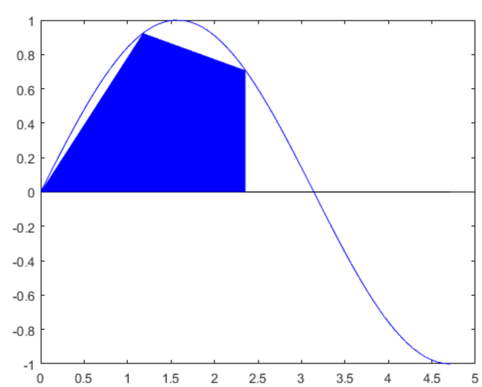
```
I = I + (fX1 + fX2)*width/2
```

Rezultat:
```
I =
    0.5442
```

7. Ponoviti korake 5 i 6 za 2, 3 i 4. podinterval:

2. podinterval

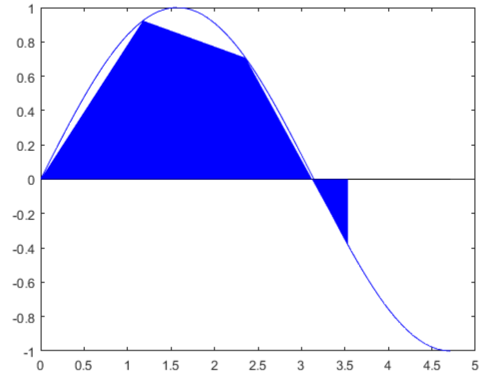
```
x1 = x(2);
x2 = x(3);
fX1 = fX(2);
fX2 = fX(3);
.
.
.
```



Rezultat:
```
I =
    1.5049
```

3. podinterval

```
x1 = x(3);
x2 = x(4);
fX1 = fX(3);
fX2 = fX(4);
.
.
.
```

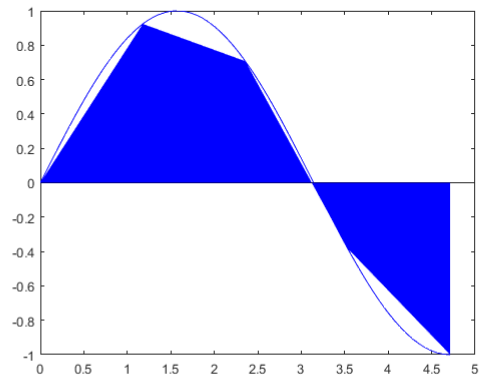


Rezultat:

```
I =
    1.6960
```

4. podinterval

```
x1 = x(4);
x2 = x(5);
fX1 = fX(4);
fX2 = fX(5);
.
.
.
```



Rezultat:

```
I =
    0.8816
```

Uporediti korake:

```
x1 = x(1);
x2 = x(2);
fX1 = fX(1);
fX2 = fX(2);
.
.
.

x1 = x(2);
x2 = x(3);
fX1 = fX(2);
fX2 = fX(3);
.
.
.

x1 = x(3);
x2 = x(4);
fX1 = fX(3);
fX2 = fX(4);
.
.
.

x1 = x(4);
x2 = x(5);
fX1 = fX(4);
fX2 = fX(5);
.
.
.
```

Proširiti indekse:

```
x1 = x(1);
x2 = x(1 + 1);
fX1 = fX(1);
fX2 = fX(1 + 1);
.
.
.

x1 = x(2);
x2 = x(2 + 1);
fX1 = fX(2);
fX2 = fX(2 + 1);
.
.
.

x1 = x(3);
x2 = x(3 + 1);
fX1 = fX(3);
fX2 = fX(3 + 1);
.
.
.

x1 = x(4);
x2 = x(4 + 1);
fX1 = fX(4);
fX2 = fX(4 + 1);
.
.
.
```

8. Primetiti šta je promenljivo. Koraci 5 i 6 se mogu zapisati jednom *for* petljom, pri čemu promenljivi indeksi zavise od indeksa *for* petlje:

```
I = 0;
for it = 1:intervals
    x1 = x(it);
    x2 = x(it + 1);
    fX1 = fX(it);
    fX2 = fX(it + 1);
    area([x1 x2], [fX1 fX2], 'FaceColor', 'blue', 'LineStyle', 'none')

    I = I + (fX1 + fX2)*width/2
end
```

9. Sada je moguće definisati funkciju koja sadrži prethodni algoritam. Na početku funkcije zatvoriti eventualno postojeći prethodni grafik:

```
function I = integrateTrapezoid(f, a, b, intervals, plotSpeed)
    close
    .
    .
    .
end
```

10. Pauzirati algoritam u zavisnosti od tražene brzine iscrtavana postupka:

```
    .
    .
    .

    for it = 1:intervals
        .
        .
        .

        pause(1/plotSpeed)
    end
end
```

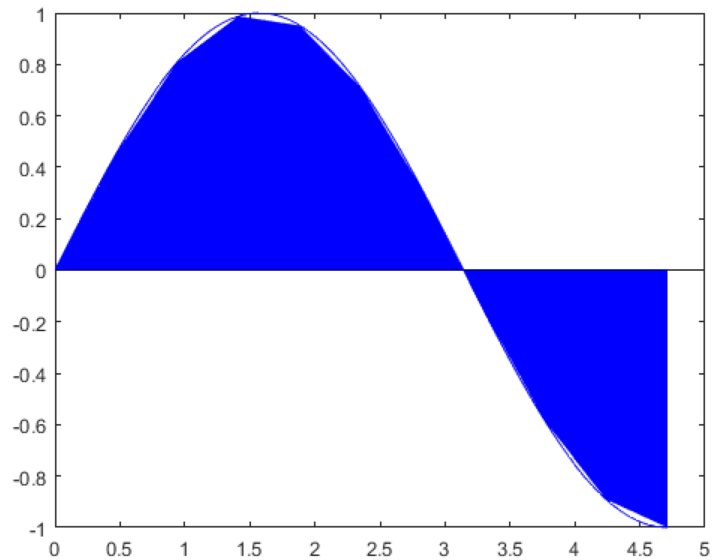
11. Testirati funkciju `integrateTrapezoid` na primeru:

```
f = @(x) sin(x);
a = 0;
b = 3*pi/2;

I = integrateTrapezoid(f, a, b, 10, 10.0)
```

Rezultat:

```
I =
    0.9814
```



Slika 6. Trapezna metoda

12. Napraviti 2 podfunkcije u funkciji `integrateTrapezoid`. Ako je prosleđena brzina iscrtavanja postupka 0 ili manja, pozvati varijantu funkcije bez naredbi za iscrtavanje i pauziranje algoritma:

```
function I = integrateTrapezoid(f, a, b, intervals, plotSpeed)
    if plotSpeed <= 0
        I = integrateTrapezoidWithoutPlot(f, a, b, intervals);
        return
    end
    I = integrateTrapezoidWithPlot(f, a, b, intervals, plotSpeed);
end
```

13. Testirati funkciju `integrateTrapezoid` na primeru:

```
f = @(x) sin(x);
a = 0;
b = 3*pi/2;

I = integrateTrapezoid(f, a, b, 1000, 0.0)
```

Rezultat:

```
I =
    1.0000
```

2. Simpsonova metoda

Zadatak 2. Napisati Simpsonovu 1/3 metodu za izračunavanje vrednosti određenog integrala proizvoljne funkcije nad proizvoljnim intervalom.
Pokušati prvo ručno jednu iteraciju Simpsonove 1/3 metode nad funkcijom $f(x) = \sin x$ na intervalu $x \in \left[0, \frac{3\pi}{2}\right]$:

```
f = @(x) sin(x);
a = 0;
b = 3*pi/2;
```

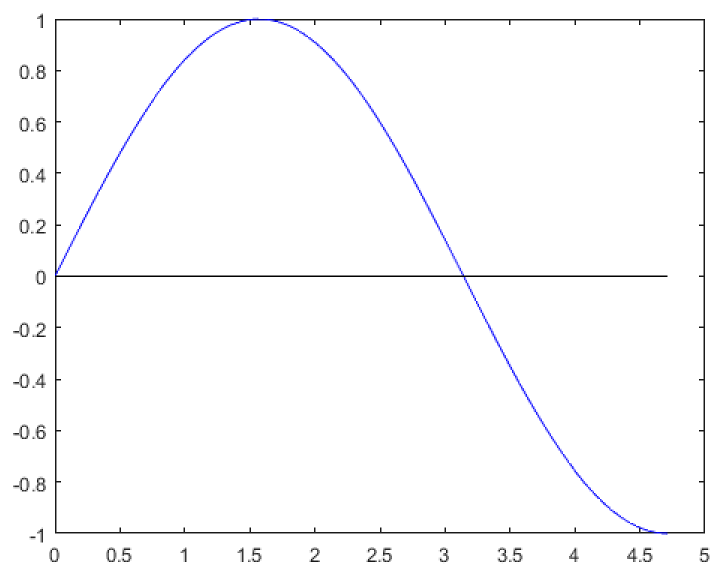
1. Definisati broj podintervala:

```
intervals = 4;
```

2. Nacrtati funkciju i x-osu nad intervalom i zadržati grafik:

```
x = linspace(a, b, 100);
fX = f(x);
plot(x, fX, 'blue', [a b], [0 0], 'black'), hold on
```

Rezultat:



Slika 7. Grafik funkcije

3. Izračunati širinu podintervala, na osnovu nje naći vrednosti nezavisno promenljive x u krajevima podintervala, a na osnovu njih naći vrednosti funkcije $f(x)$ u krajevima intervala:

```
width = (b - a)/intervals;
x = a:width:b;
fX = f(x);
```

4. Postaviti vrednost integralne sume na 0:

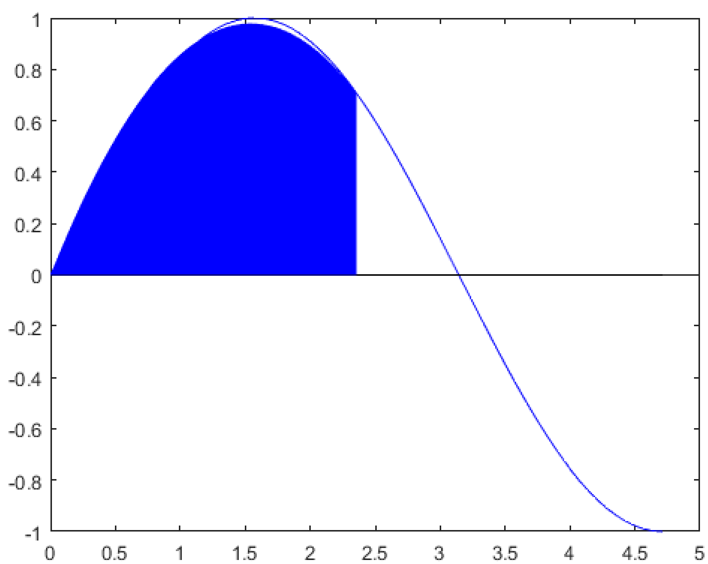
```
I = 0
```

Rezultat:
```
I =
     0
```

5. Na osnovu vrednosti funkcije u krajevima i između prva 2 podintervala i nacrtati površ između x-ose i aproksimirane funkcije polinomom 2. stepena na prva 2 podintervala:

```
x1 = x(1);
x2 = x(2);
x3 = x(3);
fX1 = fX(1);
fX2 = fX(2);
fX3 = fX(3);
p = polyfit([x1 x2 x3], [fX1 fX2 fX3], 2);
xP = linspace(x1, x3, 100);
fXP = polyval(p, xP);
area(xP, fXP, 'FaceColor', 'blue', 'LineStyle', 'none')
```

Rezultat:



Slika 8. Prva 2 podintervala

6. Naći vrednost nacrtane površine i dodati je na integralnu sumu:

```
I = I + (fX1 + 4*fX2 + fX3)*width/3
```

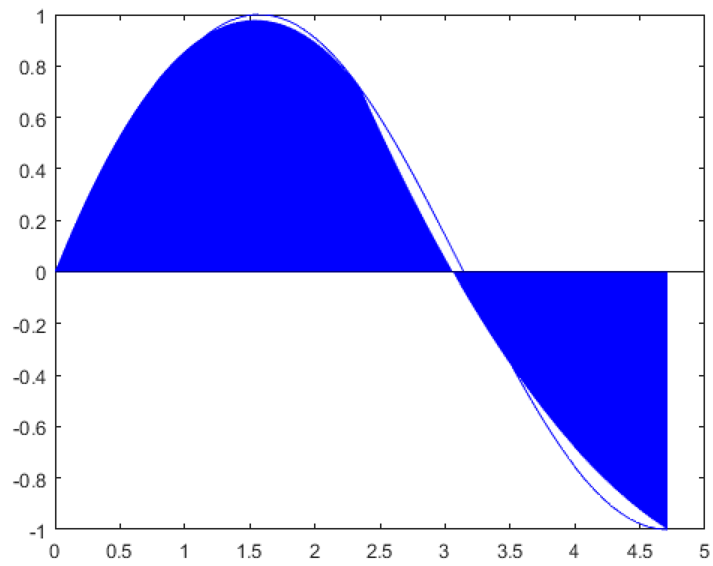
Rezultat:
```
I =
    1.7289
```

7. Ponoviti korake 5 i 6 za druga 2 podintervala:

```
x1 = x(3);
x2 = x(4);
x3 = x(5);
fX1 = fX(3);
fX2 = fX(4);
fX3 = fX(5);
.
.
.
```

Rezultat:

```
I =
    1.0128
```



Slika 9. Druga 2 podintervala

Uporediti korake:

```
x1 = x(1);
x2 = x(2);
x3 = x(3);
fX1 = fX(1);
fX2 = fX(2);
fX3 = fX(3);
.
.
.

x1 = x(3);
x2 = x(4);
x3 = x(5);
fX1 = fX(3);
fX2 = fX(4);
fX3 = fX(5);
.
.
.
```

Proširiti indekse:

```
x1 = x(1);
x2 = x(1 + 1);
x3 = x(1 + 2);
fX1 = fX(1);
fX2 = fX(1 + 1);
fX3 = fX(1 + 2);
.
.
.

x1 = x(3);
x2 = x(3 + 1);
x3 = x(3 + 2);
fX1 = fX(3);
fX2 = fX(3 + 1);
fX3 = fX(3 + 2);
.
.
.
```

10. Primetiti šta je promenljivo. Koraci 5 i 6 se mogu zapisati jednom *for* petljom, pri čemu promenljivi indeksi zavise od indeksa *for* petlje:

```
I = 0;
for it = 1:2:intervals
     x1 = x(it);
     x2 = x(it + 1);
     x3 = x(it + 2);
     fX1 = fX(it);
     fX2 = fX(it + 1);
     fX3 = fX(it + 2);
     p = polyfit([x1 x2 x3], [fX1 fX2 fX3], 2);
     xP = linspace(x1, x3, 100);
     fXP = polyval(p, xP);
     area(xP, fXP, 'FaceColor', 'blue', 'LineStyle', 'none')

     I = I + (fX1 + 4*fX2 + fX3)*width/3;
end
```

11. Sada je moguće definisati funkciju koja sadrži prethodni algoritam. Na početku funkcije zatvoriti eventualno postojeći prethodni grafik:

```
function I = integrateSimpson(f, a, b, intervals, plotSpeed)
    close
    .
    .
    .
end
```

14. Pauzirati algoritam u zavisnosti od tražene brzine iscrtavana postupka:

```
    .
    .
    .

    for it = 1:2:intervals
        .
        .
        .

        pause(1/plotSpeed)
    end
end
```

15. Testirati funkciju `integrateTrapezoid` na primeru:

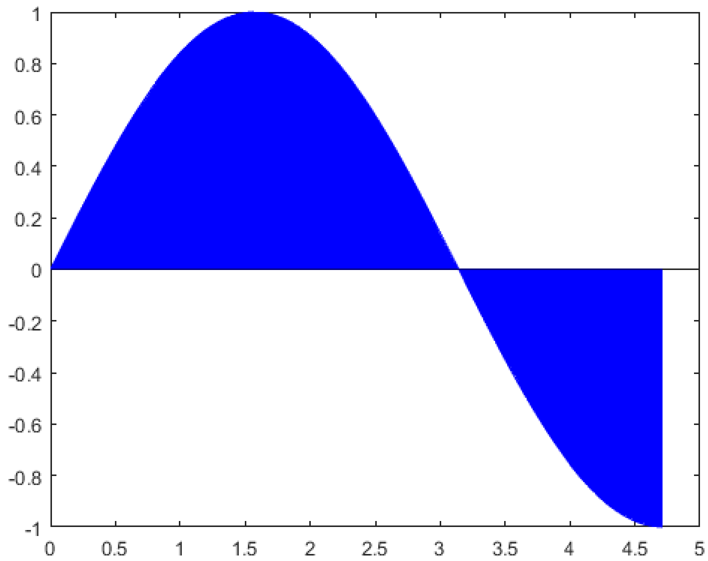
```
f = @(x) sin(x);
a = 0;
b = 3*pi/2;

I = integrateSimpson(f, a, b, 10, 10.0)
```

Rezultat:

```
I =
```

```
    1.0003
```



Slika 10. Simpsonova metoda

16. Napraviti 2 podfunkcije u funkciji `integrateSimpson`. Ako je prosleđena brzina iscrtavanja postupka 0 ili manja, pozvati varijantu funkcije bez naredbi za iscrtavanje i pauziranje algoritma:

```
function I = integrateSimpson(f, a, b, intervals, plotSpeed)
    if plotSpeed <= 0
        I = integrateSimpsonWithoutPlot(f, a, b, intervals);
        return
    end
    I = integrateSimpsonWithPlot(f, a, b, intervals, plotSpeed);
end
```

17. Testirati funkciju `integrateSimpson` na primeru:

```
f = @(x) sin(x);
a = 0;
b = 3*pi/2;

I = integrateSimpson(f, a, b, 100, 0.0)
```

Rezultat:

```
I =
    1.0000
```