1. **ZADATAK**
2. Napišite program unutar kojeg ćete u varijablu inicijalizirati listu proizvoljnog sadržaja (lista od barem 3 vrijednosti). Elemente liste spremite u pojedinačne varijable u jednoj liniji. Nakon što elemente liste imate spremljene u zasebnim varijablama, ispišite vrijednosti tih varijabli.

lista = [1, 2, 3, 4]

a, b, c, d = lista

print(a, b, c, d)

1. U dvije varijable inicijalizirajte dva različita proizvoljna broja. Nakon inicijalizacije ispišite vrijednosti varijabli na zaslon. Zamijenite vrijednosti tih dviju varijabli te ponovo ispišite njihove nove vrijednosti.

a = 5

b = 10

print(a, b)

a, b = b, a

print(a, b)

1. S tipkovnice učitajte broj proizvoljne vrijednosti. Za tako učitan broj provjerite zadovoljava li on uvjet: 10 < broj < 30, ako je uvjet zadovoljen, ispišite na ekran: "Zadovoljava!", ako pak uvjet nije zadovoljen, ispišite na ekran: "Ne zadovoljava!". Ovaj zadatak potrebno je riješiti korištenjem operatora usporedbe, bez korištenja logičkih operatora i logičkih izraza.

broj = input("Unesite broj: ")

broj = int(broj)

if 10 < broj < 30:

print("Zadovoljava!")

else:

print("Ne zadovoljava!")

1. Kreirajte listu te ju popunite s 5 proizvoljnih vrijednosti (brojeva). Nakon toga učitajte proizvoljnu vrijednost (broj) s tipkovnice. Unutar petlje for provjerite nalazi li se vrijednost koju ste učitali s tipkovnice unutar liste. Ako tražena vrijednost (broj) postoji u listi, ispišite na ekran "Postoji!", u suprotnom ispišite na ekran "Ne postoji!".

lista = [1, 2, 3, 4, 5, 6, 7]

broj = input("Unesite broj: ")

broj = int(broj)

for e in lista:

if e == broj:

print("Postoji!")

break

else:

print("Ne postoji!")

1. S tipkovnice učitajte broj proizvoljne vrijednosti. Pomoću ternarnog operatora detektirajte je li učitani broj paran ili neparan. Ako je učitani broj paran, ispišite na ekran: "Paran!", ako pak je neparan, ispišite na ekran: "Neparan!".

broj = input("Unesite broj: ")

broj = int(broj)

print("Paran!" if broj % 2 == 0 else "Neparan!")

1. Implementirajte dvije funkcije. Prva funkcija neka se zove paran() i ona neka ispisuje na ekran: "Paran!", druga funkcija neka se zove neparan() i ona neka ispisuje na ekran: "Neparan!". S tipkovnice učitajte broj proizvoljne vrijednosti te pomoću ternarnog operatora detektirajte je li učitana vrijednost parna ili neparna te ovisno o parnosti pozovite jednu od dviju funkcija koje ste prethodno kreirali.

def paran():

print("Paran!")

def neparan():

print("Neparan")

broj = input("Unesite broj: ")

broj = int(broj)

(paran if broj % 2 == 0 else neparan)()

1. Implementirajte funkciju prototipa rezultat(var1, var2), funkcija može primiti dvije vrijednosti. Tako kreirana funkcija neka u pozivajući dio programa vraća tri vrijednosti dobivene sljedećim aritmetičkim operacijama: var1\*var2, var1+var2, 10\*var1+var2. U glavnom dijelu programa pozovite prethodno implementiranu funkciju te njene povratne vrijednosti spremite u tri zasebne varijable proizvoljnog imena i ispišite ih.

def rezultat(var1, var2):

a = var1 \* var2

b = var1 + var2

c = 10 \* var1 + var2

return a, b, c

var1, var2, var3 = rezultat(5, 10)

print(var1, var2, var3)

1. Napišite program unutar kojeg će biti inicijalizirane dvije varijable proizvoljnog imena. U prvu varijablu spremite niz znakova: "Hello World!", a u drugu varijablu spremite listu sadržaja [1, 2, 3, 4]. Ispišite na ekran niz znakova i listu u obrnutom poretku na dva načina. Najprije vrijednosti ispišite pomoću petlje, a nakon toga vrijednosti ispišite na način koji je

objašnjen u poglavlju 1.1. i 1.2.

nizZnakova = "Hello World!"

lista = [1, 2, 3, 4]

print("Niz znakova (petlja): ", end="")

i = len(nizZnakova) - 1

while i >= 0:

print(nizZnakova[i], sep="", end="")

i -= 1

print("\nLista (petlja): ", end="")

i = len(lista) - 1

while i >= 0:

print(lista[i], sep="", end="")

i -= 1

print("Niz znakova:", nizZnakova[::-1])

print("Lista:", lista[::-1])

1. Napišite funkciju proizvoljnog imena koja prima proizvoljnu pozitivnu cjelobrojnu vrijednost. Rezultat je zbroj svih brojeva od 1 do primljene vrijednosti. Na primjer, ako funkcija primi vrijednost 5, u pozivajući dio programa mora se vratiti vrijednost 1+2+3+4+5 = 15. Ovo je potrebno implementirati pomoću rekurzivne funkcije.

def zbrojiBrojeveDo(x):

if x == 1:

return 1

else:

return x + zbrojiBrojeveDo(x - 1)

zbroj = zbrojiBrojeveDo(5)

print(zbroj)

1. Napišite program koji će s tipkovnice učitati jednu proizvoljnu cjelobrojnu vrijednost. Za tako učitani cijeli broj prebrojite i ispišite koliko se parnih znamenki nalazi u njemu. Prebrojavanje parnih znamenki napravite pomoću rekurzivne funkcije. Na primjer, za broj 12345, potrebno je na zaslon ispisati vrijednost 2.

def brojParnihZnamenki(x):

if x == 0:

return 0

elif x % 2 == 0:

return 1 + brojParnihZnamenki(x // 10)

else:

return brojParnihZnamenki(x // 10)

x = input("Unesite broj: ")

x = int(x)

broj = brojParnihZnamenki(x)

print(broj)

1. Napišite bezimenu funkciju koja prima 3 vrijednosti (pretpostavimo da su sve tri vrijednosti pozitivne i strogo veće od 0). Ta bezimena funkcija izračunava i vraća rezultat matematičke formule: (a\*b + c) / c. U glavnom programu ispišite rezultat matematičke formule za sljedeće vrijednosti: a = 5, b = 10; c = 2.

fun = lambda a, b, c: (a \* b + c) / c

rezultat = fun(5, 10, 2)

print(rezultat)

1. Nadogradite prethodni zadatak tako da prethodno implementirana bezimena funkcija može primiti 3 vrijednosti u rasponu od <-∞, ∞>. Ako izračun nije moguće provesti, ispišite poruku: "Err!", u suprotnom ispišite rezultat matematičke formule. Napomena: obradite slučaj dijeljenja s 0.

fun = lambda a, b, c: (a\*b+c) / c if (c != 0) else "Err!"

rezultat = fun(5, 10, 0)

print(rezultat)

1. Napisi funkciju koja prima 3 vrijednosti. Ta funkcija izračunava i vraća rezultat matematičke funkcije (a\*b+c)/c. Generirati slučajne brojeve za parametre a, b, c u rasponu od <-1000, 1000> koje moze primiti funkcija. Ako izračun nije moguće provesti ispisati "Err!", a u suprotnom ispišite rezultat matematičke formule

import random

def formula(a, b, c):

try:

result = (a \* b + c) / c

return result

except ZeroDivisionError:

return "Err!"

a = random.randint(-1000, 1000)

b = random.randint(-1000, 1000)

c = random.randint(-1000, 1000)

print("Generirani parametri:")

print("a =", a)

print("b =", b)

print("c =", c)

result = formula(a, b, c)

print("Rezultat izračuna:", result)

1. Isprobajte korištenje ugniježđenih funkcija. Vanjska funkcija imena izracun() neka prima dva parametra. Unutar vanjske funkcije kreirajte dvije unutarnje funkcije imena parniZbroj() i neparniZbroj(). Funkcija parniZbroj() neka vraća rezultat matematičke formule 2\*a+5\*b, a funkcija neparniZbroj() neka vraća rezultat matematičke formule a\*b-10. Koja će se od dviju kreiranih unutarnjih funkcija pozvati neka se odredi na temelju parnosti zbroja primljenih parametara a+b. U glavnom programu na zaslon ispišite rezultat.

def izracun(a, b):

def parniZbroj(a, b):

return 2 \* a + 5 \* b

def neparniZbroj(a, b):

return a \* b - 10

if (a + b) % 2 == 0:

return parniZbroj(a, b)

else:

return neparniZbroj(a, b)

rezultat = izracun(5, 15)

print(rezultat)

1. Implementirajte generatorsku funkciju imena danUTjednu(), funkcija mora vratiti generator s danima u tjednu. Nakon što svi dani u tjednu budu otpušteni, ne kreće se ponovo s otpuštanjem vrijednosti od ponedjeljka. U glavnom programu pozivom implementirane generatorske funkcije ispišite sve dane u tjednu.

def danUTjednu():

dani = ["Ponedjeljak",

"Utorak",

"Srijeda",

"Četvrtak",

"Petak",

"Subota",

"Nedjelja"]

for e in dani:

yield e

g = danUTjednu()

for e in g:

print(e)

1. Implementirajte generatorsku funkciju imena jedanDvaTri() koja vraća generator koji pak vraća vrijednosti: "Jedan", "Dva", "Tri", "Jedan", "Dva", "Tri", "Jedan", "Dva", "Tri" i tako u beskonačnost. U glavnom programu ispišite proizvoljan broj puta sekvencu: "Jedan", "Dva", "Tri".

def jedanDvaTri():

brojac = ["Jedan",

"Dva",

"Tri"]

while True:

for e in brojac:

yield e

g = jedanDvaTri()

print(next(g))

print(next(g))

print(next(g))

print(next(g))

print(next(g))

print(next(g))

print(next(g))

print(next(g))

1. Napišite program koji će od korisnika tražiti dvije vrijednosti. Prva vrijednost neka bude neki proizvoljan cijeli broj, a druga vrijednost neka bude znamenka od 0 do 9. Kreirajte rekurzivnu funkciju koja će odrediti broj pojavljivanja zadane znamenke u zadanom broju. Prototip funkcije: prebroji(broj, znamenka)

def prebroji(broj, znamenka):

if broj == 0:

return 0

elif (broj % 10) == znamenka:

return 1 + prebroji(broj // 10, znamenka)

else:

return prebroji(broj // 10, znamenka)

rezultat = prebroji(1223455, 2)

print(rezultat)

1. Napišite program koji će s tipkovnice učitati jednu proizvoljnu cjelobrojnu vrijednosti. Za tako učitani broj, odredite pomoću rekurzivne funkcije je li on prost broj. Ako je učitani broj prost, ispišite na zaslon poruku: "Prost je!", u suprotnom ispišite "Nije prost!".

def jeProst(broj):

def jeProstInner(broj, i):

if broj == i:

return 1

if broj % i == 0:

return 0

return jeProstInner(broj, i + 1)

return jeProstInner(broj, 2)

if jeProst(7) == 1:

print("Prost je!")

else:

print("Nije prost!")

1. Kreirajte listu od 5 proizvoljnih vrijednosti koje će predstavljati temperaturu izraženu u stupnjevima Celzijevim. Napišite bezimenu funkciju koja će uz pomoć funkcije map() generirati vrijednost temperature izražene u Farenhajtima. Formula: ( 𝑥 × 9/5) + 32 , x je vrijednost izražena u stupnjevima Celzijevim. Tako dobivenu listu ispišite na ekran.

celsius = [0, 5, 10, 15.8]

fahrenheit = map(lambda x: (float(9) / 5) \* x + 32,

celsius)

print(list(fahrenheit))

1. Napišite generatorsku funkciju imena bottles(brojBoca, nazivNapitka). Ova funkcija mora vratiti generator koji otpušta stihove pjesme: "99 Bottles of Beer". Svaki stih ove pjesme ima za jedan napitak manje od prethodnog stiha, tako dugo dok se ne istroše svi napitci. Predefinirana vrijednost za broj boca je 99, dok je predefinirana vrijednost za naziv napitka "beer". U nastavku slijedi primjer ispisa za dani parametar

brojBoca = 4.

4 bottles of beer on the wall, 4 bottles of beer. Take one down and pass it around, 3 bottles of beer on the wall.

3 bottles of beer on the wall, 3 bottles of beer. Take one down and pass it around, 2 bottles of beer on the wall.

2 bottles of beer on the wall, 2 bottles of beer. Take one down and pass it around, 1 bottle of beer on the wall.

1 bottle of beer on the wall, 1 bottle of beer. Take one down and pass it around, no more bottles of beer on

the wall.

No more bottles of beer on the wall, no more bottles of beer. Go to the store and buy some more, 99 bottles of beer on the wall.

def bottles(brojBoca=99, nazivNapitka="beer"):

brojBocaNaPocetku = brojBoca

while brojBoca >= 0:

if brojBoca >= 2:

yield (str(brojBoca) + " bottles of " +

nazivNapitka + " on the wall, " +

str(brojBoca) + " bottles of " +

nazivNapitka + ".\n" +

"Take one down and pass it around, " +

str(- 1) + " bottles of " +

nazivNapitka + " on the wall.\n")

elif brojBoca == 1:

yield (str(brojBoca) + " bottle of " +

nazivNapitka + " on the wall, " +

str(brojBoca) + " bottle of " +

nazivNapitka + ".\n" +

"Take one down and pass it around, " +

"no more bottles of " +

nazivNapitka + " on the wall.\n")

else:

yield ("No more bottles of " + nazivNapitka +

" on the wall, no more bottles of " +

nazivNapitka + ".\n" +

"Go to the store and buy some more, "

+ str(brojBocaNaPocetku) +

" bottles of " +

nazivNapitka + " on the wall.")

brojBoca -= 1

g = bottles(4)

for e in g:

print(e)

1. **ZADATAK – MATRICE**
2. - Generirati matricu A dimenzija 4x5 sa slučajno odabranim cijelim brojevima od 0 do 100

- Potencirati svaki element matrice potencijom 2

- Generirati matricu B dimenzija 3x5 slučajno odabranim brojevima od 1 do 10

- Pomnožiti matricu A i B (dopuniti matricu B nulama kako bi dimenzije dviju matrica bile iste)

- Rezultat spremiti u varijablu “rezultat” i svaki element u dobivenoj matrici povećati za 2

- U prvi redak rezultata postaviti sve nule umjesto postojećih elemenata

- Transponirati dobivenu matricu rezultat

- U varijablu b spremiti vektor čiji su elementi sume elemenata stupca matrice

- Ispisati dobiveni vektor b

• Napomena: Obavezno koristiti naredbe za rad s matricama i rješenje spremiti kao zadatak2.py i pozivati iz naredbenog retka.

import numpy as np

A = np.random.randint(0, 101, size=(4, 5))

print("Matrica A:")

print(A)

A\_squared = np.power(A, 2)

print("\nMatrica A sa svakim elementom potenciranim na 2:")

print(A\_squared)

B = np.random.randint(1, 11, size=(3, 5))

print("\nMatrica B:")

print(B)

B\_padded = np.pad(B, ((0, 1), (0, 0)), 'constant', constant\_values=0)

print("\nMatrica B nakon dodavanja nula:")

print(B\_padded)

rezultat = np.dot(A\_squared, B\_padded.T)  # Transponiraj B kako bi omogućio množenje

print("\nRezultat množenja matrica A i B:")

print(rezultat)

rezultat += 2

print("\nRezultat nakon povećanja svakog elementa za 2:")

print(rezultat)

rezultat[0, :] = 0

print("\nRezultat nakon postavljanja prvog retka na nule:")

print(rezultat)

rezultat\_transposed = rezultat.T

print("\nTransponirani rezultat:")

print(rezultat\_transposed)

b = np.sum(rezultat\_transposed, axis=0)

print("\nVektor b (sume stupaca):")

print(b)

1. Generirati matricu A dimenzija (3x3) i matricu B dimenzija (3x4) sa slučajno odabranim brojevima od 0 do 10

- Dopuniti matricu A tricama kako bi dimenzije dviju matrica bile iste

- Zbrojiti matrice A i B, rezultat spremiti u varijablu “rezultat” i svaki element u dobivenoj matrici povećati za 2

- Zbrojiti sve elemente u matrici „rezultat“ i dobivenu varijablu nazvati „suma“

- Kreirati matricu C dimenzija (4x4) u kojoj će svi elementi biti jednaki varijabli „suma“

- Smanjiti sve elemente na dijagonali matrice C za 20

- Kvadrirati sve elemente prvog retka matrice C

\*Napomena: Obavezno koristiti naredbe za rad s matricama i rješenje spremiti kao zadatak2.py i pozivati iz naredbenog retka

import numpy as np

A = np.random.randint(0, 10, size=(3, 3))

print("\nMatrica A: ")

print(A)

B = np.random.randint(0, 10, size=(3, 4))

print("\nMatrica B: ")

print(B)

A = np.insert(A, 3, np.ones((1, 3), dtype=int) \* 3, axis=1)

print("\nMatrica A dopunjena 3: ")

print(A)

rezultat = A + B

print("\nZbroj matrica A i B: ")

print(rezultat)

rezultat\_2 = rezultat + 2

print("\nSvaki element uvećan za 2: ")

print(rezultat\_2)

suma = rezultat.sum()

print("\nSuma elemenata u zbrojenoj matrici matrica A i B: ")

print(suma)

C = np.ones((4, 4), dtype=int) \* suma - np.eye(4, 4, dtype=int) \* 20

print("\nSmanjili elemente na dijagonali matrice C za 20: ")

print(C)

C[0, :] = C[0, :] \*\* 2

print("\nKvadrirali 1. redak matrice C: ")

print(C)

1. a) Generirati matrice x i y:

[10 10 5 10 10]

[10 10 5 10 10]

x= [ 5 5 5 5 5]

[10 10 5 10 10]

[10 10 5 10 10]

[8 8 8]

[3 3 3]

y= [3 3 3]

[8 8 8]

[8 8 8]

b) Spojiti matrice x i y tako da novonastala matrica z bude dimenzija (5,8)

c) Dopuniti matricu z nulama tako da ona bude kvadratna

d) Na dijagonali matrice z pomnožiti sve elemente sa 3

e)Sve elemente u parnim stupcima matrice kvadrirati

f)Na sve pozicije koje imaju vrijednost 0 postaviti broj koji predstavlja sumu svih elemenata u tom stupcu matrice

import numpy as np

print("\nMatrica x: ")

x = np.ones((5,5))\*10

x[2,:]=5

x[:,2]=5

print("\n",x.astype(int))

print("\nMatrica y: ")

y = np.ones((5,3))\*8

y[1:3,:]=3

print("\n",y.astype(int))

print("\nNovonastala matrica z: ")

z = np.concatenate((x,y),axis=1)

print("\n",z.astype(int))

print("\nKvadratna matrica z dopunjena nulama: ")

z = np.insert(z,5,np.zeros((3,8)),axis=0)

print("\n",z.astype(int))

print("\nDijagonala matrice z pomnožena s 3")

np.fill\_diagonal(z,np.diag(z)\*3)

print("\n",z.astype(int))

print("\nKvadrirani parni stupci matrice z: ")

z[:,1:9:2]=z[:,1:9:2]\*\*2

print("\n",z.astype(int))

print("\nSuma elemenata stupca na sve 0")

z[5:9,:]=np.sum(z[:,0:8],axis=0)

print("\n",z.astype(int))

1. Generirati matrice x(5x5) i y(10x4):

[100 100 100 100 10]

[100 100 100 10 100]

x= [ 5 5 5 5 5]

[100 10 100 100 100]

[ 10 100 100 100 100]

[ 9 9 9 10]

[ 9 9 9 10]

y = [ 8 8 3 8]

[ ... ]

[ 8 8 3 8]

a) Spojiti matrice xiy tako da novonastala matrica z bude dimenzija (15x5)

b) Sve elemente u parnim stupcima matrice z podijeliti s 10

c) Sve elemente u neparnim retcima matrice z postaviti na nulu

d) U matricu z dodati još jedan stupac čiji su elementi sume po retcima

e) Obavezno koristiti naredbe za rad s matricama i zadatak pokrenuti iz naredbenog retka

import numpy as np

print("\nMatrica x: ")

x = np.ones((5,5))\*100-np.fliplr(np.eye(5,5)\*90)

x[2,:]=5

print("\n",x)

print("\nMatrica y: ")

y = np.ones((10,4))\*8

y[:,2]=3

y[0:2,:]=9

y[0:2,3]=10

print("\n",y)

print("\nDopunjena matrica y")

y = np.insert(y,4,np.zeros((1,10)), axis=1)

print("\n",y)

print("\nMatrica z (spoj x i y): ")

z = np.concatenate((x,y),axis=0)

print("\n",z)

print("\nParni stupci matrice podijeljeni s 10: ")

z[:,1:5:2]=z[:,1:5:2]/10

print("\n",z)

print("\nSvi elementi u neparnim retcima postaviti na 0")

z[0:15:2]=0

print("\n",z)

print("\nStupac čiji su elementi sume redaka")

suma\_redaka = np.sum(z,axis=1)

print("\n",suma\_redaka)

print("\nMatrica s dodatnim retkom")

z = np.insert(z,5,suma\_redaka,axis=1)

print("\n",z)

1. **ZADATAK – OPTIMIZACIJE**
2. Definirati vektor a u intervalu [-10, 10] s korakom 0.1.

Definirati vektor b=a^2. Na istom grafu (u istom prostoru-figure) nacrtati:

-vektor b (zelenom bojom, crtkana linija)

-sin(b) (crvenom bojom)

-dodati labele za svaki

-Naslov slike treba biti "Ispitni zadatak"

-Naslov x-ose "x-osa" (interval[-1, 10])

-Naslov y-ose "y-osa" (interval [-2, 9])

import numpy as np

import matplotlib.pyplot as plt

# Definiranje vektora a u intervalu [-10, 10] s korakom 0.1

a = np.arange(-10, 10.1, 0.1)

# Definiranje vektora b = a^2

b = a\*\*2

# Crtanje vektora b zelenom bojom, crtkanom linijom

plt.plot(a, b, 'g--', label='b = a^2')

# Crtanje sinusa od vektora b crvenom bojom

plt.plot(a, np.sin(b), 'r', label='sin(b)')

# Postavljanje naslova slike

plt.title('Ispitni zadatak')

# Postavljanje naslova x-ose i y-ose

plt.xlabel('x-osa (interval [-10, 10])')

plt.ylabel('y-osa (interval [-2, 9])')

# Postavljanje intervala x-ose i y-ose

plt.xlim(-10, 10)

plt.ylim(-2, 9)

# Dodavanje legende

plt.legend()

# Prikazivanje grafa

plt.show()

1. Definirati vektor a u intervalu [0, 10] s korakom 0.1.

Definirati vektor b=a^3+a^2+a. Na istom grafu (u istom prostoru-figure) nacrtati:

-vektor b (zelenom bojom, crtkana linija)

-cos(b) (crvenom bojom)

-dodati labele za svaki

-Naslov slike treba biti "Ispitni zadatak"

-Naslov x-ose "x-osa" (interval[-1, 10])

-Naslov y-ose "y-osa" (interval [-2, 9])

import numpy as np

import matplotlib.pyplot as plt

# Definiranje vektora a u intervalu [0, 10] s korakom 0.1

a = np.arange(0, 10.1, 0.1)

# Definiranje vektora b = a^3 + a^2 + a

b = a\*\*3 + a\*\*2 + a

# Crtanje vektora b zelenom bojom, crtkanom linijom

plt.plot(a, b, 'g--', label='b = a^3 + a^2 + a')

# Crtanje cosinusa od vektora b crvenom bojom

plt.plot(a, np.cos(b), 'r', label='cos(b)')

# Postavljanje naslova slike

plt.title('Ispitni zadatak')

# Postavljanje naslova x-ose i y-ose

plt.xlabel('x-osa (interval [-1, 10])')

plt.ylabel('y-osa (interval [-2, 9])')

# Postavljanje intervala x-ose i y-ose

plt.xlim(-1, 10)

plt.ylim(-2, 9)

# Dodavanje legende

plt.legend()

# Prikazivanje grafa

plt.show()

1. Definirati vektor a u intervalu [0, 10] s korakom 0.1.

Definirati vektor b=a^2. Na istom grafu (u istom prostoru-figure) nacrtati:

-vektor b (zelenom bojom, crtkana linija)

-sin(b) (crvenom bojom)

-dodati labele za svaki

-Naslov slike treba biti "Ispitni zadatak"

-Naslov x-ose "x-osa" (interval[-1, 10])

-Naslov y-ose "y-osa" (interval [-2, 9])

import numpy as np

import matplotlib.pyplot as plt

#Definiranje vektora a u intervalu [0, 10] s korakom 0.1

a = np.arange(0, 10.1, 0.1)

#Definiranje vektora b = a^2

b = a\*\*2

#Crtanje vektora b zelenom bojom, crtkanom linijom

plt.plot(a, b, 'g--', label= 'b = a^2')

#Crtanje sinusa od vektora b crvenom bojom

plt.plot(a, np.sin(b), 'r', label= 'sin(b)')

#Postavljanje naslova slike

plt.title('Ispitni zadatak')

#Postavljanje naslova x-ose i y-ose

plt.xlabel('x-osa (interval [-1,11])')

plt.ylabel('y-osa (interval [-2, 10])')

#Postavljanje intervala x-ose i y-ose

plt.xlim(-1, 10)

plt.ylim(-2, 9)

# Dodavanje legendi

plt.legend()

# Prikazivanje grafa

plt.show()

1. Nacrtati funkciju kao na slici:

Slika 1 zubasta

Slika 2

Slika 3

Napomena: funkcija1 i funkcija2 neka budu različitih boja, crtkane linije.

import matplotlib.pyplot as plt

import numpy as np

t = np.arange(-10, 10, 1)

f1 = np.array([-1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1])

f2 = np.array([1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1, 1, -1])

# Graf 1

plt.subplot(3, 1, 1)

plt.plot(t, f1, color="g", linestyle="--")

plt.xlim([-11, 11])

plt.ylim([-2, 2])

plt.legend(["funkcija 1"], loc='upper right') # Postavljanje legende izvan granica grafa

# Graf 2

plt.subplot(3, 1, 2)

plt.plot(t, f2, color="r", linestyle="--")

plt.xlim([-11, 11])

plt.ylim([-2, 2])

plt.legend(["funkcija 2"], loc='upper right') # Postavljanje legende izvan granica grafa

# Graf 3

plt.subplot(3, 1, 3)

plt.plot(t, f1, color="g", linestyle="--")

plt.plot(t, f2, color="r", linestyle="--")

plt.xlim([-11, 11])

plt.ylim([-2, 2])

plt.legend(["funkcija 1", "funkcija 2"], loc='upper right') # Postavljanje legende izvan granica grafa

# Dodajemo naslov

plt.suptitle('2. zadatak')

plt.tight\_layout()

plt.show()

1. Nacrtati funkciju kao na slici:

parabola

parabola

parabola

Napomena: funkcija1 i funkcija2 neka budu različitih boja, crkane linije.

import numpy as np

import matplotlib.pyplot as plt

t = np.arange(-100,100,0.5)

f1 = t \*\* 2

f2 = -t \*\* 2

f31 = t \*\* 2 - 20

f32 = -t \*\* 2 + 20

#Graf 1

plt.subplot(2,2,1)

plt.plot(t, f1, color = "g", linestyle = "--", label = "funkcija f")

plt.xlim([-10,10])

plt.ylim([-5,20])

plt.legend()

#Graf 2

plt.subplot(2,2,3)

plt.plot(t, f2, color = "r", linestyle = "--", label = "funkcija g")

plt.xlim([-10,10])

plt.ylim([-20,5])

plt.legend()

#graf 3

plt.subplot(1,2,2)

plt.plot(t, f31, color = "g", linestyle = "--", label="f")

plt.plot(t, f32, color = "r", linestyle = "--", label = "g")

plt.xlim(-10,10)

plt.ylim([-20,20])

plt.legend()

plt.title("2. zadatak")

plt.tight\_layout()

plt.show()