1MPR13_Simona_Bļinova sb24037

1.uzdevums

Programma, kas realizē veikala simulāciju.

Kods:

```
class Prece:
  def __init__(self, artikuls, nosaukums, daudzums, cena):
    self.artikuls = artikuls
    self.nosaukums = nosaukums
    self.daudzums = daudzums
    self.cena = cena

  def __str__(self):
    return f'{self.nosaukums} (Artikuls: {self.artikuls}) - {self.daudzums} gab., {self.cena:.2f} EUR'

class Veikals:
  def __init__(self):
    self.krajumi = {} # key: preces nosaukums, vertiba: prece objekts

  def pievienot_preci(self, prece):
    if prece.nosaukums in self.krajumi:
      self.krajumi[prece.nosaukums].daudzums += prece.daudzums
    else:
      self.krajumi[prece.nosaukums] = prece

  def paradit_preces(self):
    print('Pieejamas preces veikalā:')
    for prece in self.krajumi:
      print(prece)

  def ir_prece(self, nosaukums, daudzums):
    return nosaukums in self.krajumi and self.krajumi[nosaukums].daudzums >= daudzums

  def pardod_preci(self, nosaukums, daudzums):
    if self.ir_prece(nosaukums, daudzums):
      prece = self.krajumi[nosaukums]
      prece.daudzums -= daudzums
      return Prece(prece.artikuls, nosaukums, daudzums, prece.cena)
    else:
      print(f'Nav pietiekami daudz preces: {nosaukums}')
```

```
        return None

class Grozs:
    def __init__(self):
        self.pirkumi = []

    def pievienot_pirkumu(self, prece):
        self.pirkumi.append(prece)

    def izdrukat_cekus(self):
        print('Pirkuma čeks:')
        kopa = 0
        for prece in self.pirkumi:
            summa = prece.daudzums * prece.cena
            print(f'{prece.nosaukums} x {prece.daudzums} = {summa:.2f} EUR')
            kopa += summa
        print(f'Kopa jamaksa: {kopa:.2f} EUR')

def galvena_programma():
    veikals = Veikals()

    print('~~~ Preču piegāde veikalām ~~~')
    while True:
        artikuls = input('\nIevadiet artikulu --> ')
        nosaukums = input('Ievadiet preces nosaukumu --> ')
        daudzums = int(input('Ievadiet daudzumu --> '))
        cena = float(input('Ievadiet preces cenu (EUR) --> '))
        veikals.pievienot_preci(Prece(artikuls, nosaukums, daudzums, cena))

        paz = input('\nVai pievienot vēl preces? (j/n)')
        if paz == 'n':
            break

    print(' ')
    veikals.paradit_preces()

    print('\n~~~ Iepirkšanas ~~~')
    grozs = Grozs()

    while True:
        nosaukums = input('\nIevadiet preces nosaukumu --> ')
```

```
        daudzums = int(input('Ievadiet daudzumu --> '))
        if veikals.ir_prece(nosaukums, daudzums):
            prece = veikals.pardod_preci(nosaukums, daudzums)
            grozs.pievienot_pirkumu(prece)
        else:
            print('Nav pieejams!')

        paz = input('\n Vai pievienot vēl preces? (j/n)')
        if paz == 'n':
            break

    print('\n~~~ Grozs ~~~')
    print(' ')
    grozs.izdrukat_cekus()

if __name__ == '__main__':
    galvena_programma()
```

Testa piemērs(1)

```
~~~ Preču piegāde veikalām ~~~

Ievadiet artikulu --> 01
Ievadiet preces nosaukumu --> Apelsīnu sula
Ievadiet daudzumu --> 30
Ievadiet preces cenu (EUR) --> 1.30

Vai pievienot vēl preces? (j/n)j

Ievadiet artikulu --> 02
Ievadiet preces nosaukumu --> Sieriņš
Ievadiet daudzumu --> 100
Ievadiet preces cenu (EUR) --> 0.59

Vai pievienot vēl preces? (j/n)n

Pieejamas preces veikalā:
Apelsīnu sula
Sieriņš

~~~ Iepirkšanas ~~~

Ievadiet preces nosaukumu --> Sieriņš
Ievadiet daudzumu --> 5

 Vai pievienot vēl preces? (j/n)j

Ievadiet preces nosaukumu --> Apelsīnu sula
Ievadiet daudzumu --> 1

 Vai pievienot vēl preces? (j/n)n

~~~ Grozs ~~~

Pirkuma čeks:
Sieriņš x 5 = 2.95 EUR
Apelsīnu sula x 1 = 1.30 EUR
Kopa jamaksa: 4.25 EUR
```

Testa piemērs(2)

```
~~~ Preču piegāde veikalām ~~~

Ievadiet artikulu --> 01
Ievadiet preces nosaukumu --> Zeķes
Ievadiet daudzumu --> 20
Ievadiet preces cenu (EUR) --> 5.99

Vai pievienot vēl preces? (j/n)n

Pieejamas preces veikalā:
Zeķes

~~~ Iepirkšanas ~~~

Ievadiet preces nosaukumu --> Sula
Ievadiet daudzumu --> 2
Nav pieejams!

 Vai pievienot vēl preces? (j/n)j

Ievadiet preces nosaukumu --> Zeķes
Ievadiet daudzumu --> 1

 Vai pievienot vēl preces? (j/n)n

~~~ Grozs ~~~

Pirkuma čeks:
Zeķes x 1 = 5.99 EUR
Kopa jamaksa: 5.99 EUR
```

Testa piemērs(3)

```
~~~ Preču piegāde veikalām ~~~

Ievadiet artikulu --> 01
Ievadiet preces nosaukumu --> Šokolāde tumša
Ievadiet daudzumu --> 5
Ievadiet preces cenu (EUR) --> 2.99

Vai pievienot vēl preces? (j/n)j

Ievadiet artikulu --> 02
Ievadiet preces nosaukumu --> Piens 2%
Ievadiet daudzumu --> 30
Ievadiet preces cenu (EUR) --> 0.95

Vai pievienot vēl preces? (j/n)n

Pieejamas preces veikalā:
Šokolāde tumša
Piens 2%

~~~ Iepirkšanas ~~~

Ievadiet preces nosaukumu --> Šokolāde tumša
Ievadiet daudzumu --> 2

 Vai pievienot vēl preces? (j/n)n

~~~ Grozs ~~~

Pirkuma čeks:
Šokolāde tumša x 2 = 5.98 EUR
Kopa jamaksa: 5.98 EUR
```

2.uzdevums

Programma, kas realizē darbības ar kompleksiem skaitļiem.

Kods:

```
import math

class Kompleksais_skaitlis:
  def __init__(self, a, b):
    self.z = (a, b)

  def __str__(self):
    if self.z[0] == 0 and self.z[1] == 0:
      return '0'
    elif self.z[0] == 0:
      return f'{self.z[1]}i'
    elif self.z[1] == 0:
      return f'{self.z[0]}'
    elif self.z[1] < 0:
      return f'{self.z[0]} - {abs(self.z[1])}i'
    else:
      return f'{self.z[0]} + {self.z[1]}i'


class Darbibas:
  def __init__(self, sk1, sk2):
    self.darb = {}
    self.darb['sk1'] = sk1
    self.darb['sk2'] = sk2

  def saistitais(self):
    self.darb['saist1'] = Kompleksais_skaitlis(self.darb['sk1'].z[0], -self.darb['sk1'].z[1])
    self.darb['saist2'] = Kompleksais_skaitlis(self.darb['sk2'].z[0], -self.darb['sk2'].z[1])

  def modulis(self):
    self.darb['modulis1'] = str(math.sqrt(self.darb['sk1'].z[0]**2 + self.darb['sk1'].z[1]**2))
    self.darb['modulis2'] = str(math.sqrt(self.darb['sk2'].z[0]**2 + self.darb['sk2'].z[1]**2))

  def summa(self):
    a = self.darb['sk1'].z[0] + self.darb['sk2'].z[0]
    b = self.darb['sk1'].z[1] + self.darb['sk2'].z[1]
```

```
        self.darb['summa'] = Kompleksais_skaitlis(a, b)

    def reizinajums(self):
        a = self.darb['sk1'].z[0] * self.darb['sk2'].z[0] - self.darb['sk1'].z[1] * self.darb['sk2'].z[1]
        b = self.darb['sk1'].z[0] * self.darb['sk2'].z[1] + self.darb['sk2'].z[0] * self.darb['sk1'].z[1]
        self.darb['reizinajums'] = Kompleksais_skaitlis(a, b)

    def dalijums(self):
        if self.darb['sk2'].z[0] != 0 and self.darb['sk2'].z[1] != 0:
            saucejs = self.darb['sk2'].z[0] ** 2 + self.darb['sk2'].z[1] ** 2
            a_skait = self.darb['sk1'].z[0] * self.darb['sk2'].z[0] + self.darb['sk1'].z[1] * self.darb['sk2'].z[1]
            b_skait = self.darb['sk2'].z[0] * self.darb['sk1'].z[1] - self.darb['sk1'].z[0] * self.darb['sk2'].z[1]
            self.darb['dalijums'] = Kompleksais_skaitlis(a_skait/saucejs, b_skait/saucejs)
        else:
            self.darb['dalijums'] = '---'

def galvena_programma():
    a1 = float(input('Ievadiet reālo koeficientu 1. skaitlim --> '))
    b1 = float(input('Ievadiet imaginaro koeficientu 1. skaitlim --> '))
    skaitlis1 = Kompleksais_skaitlis(a1, b1)

    a2 = float(input('Ievadiet reālo koeficientu 2. skaitlim --> '))
    b2 = float(input('Ievadiet imaginaro koeficientu 2. skaitlim --> '))
    skaitlis2 = Kompleksais_skaitlis(a2, b2)

    print('')

    darbibas = Darbibas(skaitlis1, skaitlis2)

    darbibas.saistitais()
    darbibas.modulis()
    darbibas.summa()
    darbibas.reizinajums()
    darbibas.dalijums()

    vertibas = []
    for vertiba in darbibas.darb.values():
        vertibas.append(vertiba)

    print('Pirmais skaitlis:', vertibas[0])
    print('Otrais skaitlis:', vertibas[1])
```

```
    print('Pirmā skaitļa saistītais:', vertibas[2])
    print('Otrā skaitļa saistītais:', vertibas[3])
    print('Pirmā skaitļa modulis:', vertibas[4])
    print('Otrā skaitļa modulis:', vertibas[5])
    print('Skaitļu summa:', vertibas[6])
    print('Skaitļu reizinājums:', vertibas[7])
    print('Skaitļu dalījums:', vertibas[8])

if __name__ == '__main__':
    galvena_programma()
```

Testa piemērs(1)

```
Ievadiet reālo koeficientu 1. skaitlim --> 0.5
Ievadiet imaginaro koeficientu 1. skaitlim --> -1
Ievadiet reālo koeficientu 2. skaitlim --> 3
Ievadiet imaginaro koeficientu 2. skaitlim --> 2

Pirmais skaitlis: 0.5 - 1.0i
Otrais skaitlis: 3.0 + 2.0i
Pirmā skaitļa saistītais: 0.5 + 1.0i
Otrā skaitļa saistītais: 3.0 - 2.0i
Pirmā skaitļa modulis: 1.118033988749895
Otrā skaitļa modulis: 3.605551275463989
Skaitļu summa: 3.5 + 1.0i
Skaitļu reizinājums: 3.5 - 2.0i
Skaitļu dalījums: -0.038461538461538464 - 0.3076923076923077i
```

Testa piemērs(2)

```
Ievadiet reālo koeficientu 1. skaitlim --> 1
Ievadiet imaginaro koeficientu 1. skaitlim --> 2
Ievadiet reālo koeficientu 2. skaitlim --> 3
Ievadiet imaginaro koeficientu 2. skaitlim --> 4

Pirmais skaitlis: 1.0 + 2.0i
Otrais skaitlis: 3.0 + 4.0i
Pirmā skaitļa saistītais: 1.0 - 2.0i
Otrā skaitļa saistītais: 3.0 - 4.0i
Pirmā skaitļa modulis: 2.23606797749979
Otrā skaitļa modulis: 5.0
Skaitļu summa: 4.0 + 6.0i
Skaitļu reizinājums: -5.0 + 10.0i
Skaitļu dalījums: 0.44 + 0.08i
```

Testa piemērs(3)

```
Ievadiet reālo koeficientu 1. skaitlim --> 1
Ievadiet imaginaro koeficientu 1. skaitlim --> 1
Ievadiet reālo koeficientu 2. skaitlim --> 1
Ievadiet imaginaro koeficientu 2. skaitlim --> 1

Pirmais skaitlis: 1.0 + 1.0i
Otrais skaitlis: 1.0 + 1.0i
Pirmā skaitļa saistītais: 1.0 - 1.0i
Otrā skaitļa saistītais: 1.0 - 1.0i
Pirmā skaitļa modulis: 1.4142135623730951
Otrā skaitļa modulis: 1.4142135623730951
Skaitļu summa: 2.0 + 2.0i
Skaitļu reizinājums: 2.0i
Skaitļu dalījums: 1.0
```

3.uzdevums

Programma, kas aprēķina n-stūra laukumu.

Kods:

```
class Virsotne:
  def __init__(self, x, y):
    self.koordinatas = (x, y)

class Nsturis:
  def __init__(self, virsotnes):
    self.virsotnes = virsotnes

  def pievienot_virsotni(self, x, y):
    self.virsotnes.append(Virsotne(x, y))

  def laukums(self):
    if len(self.virsotnes) < 3:
      return 'Virsotņu skaits nav pietiekams laukuma aprēķinam.'
    else:
      summa = 0
      for i in range(len(self.virsotnes)-1):
        #print(self.virsotnes[i].koordinatas[0], self.virsotnes[i+1].koordinatas[0])
        s = (self.virsotnes[i].koordinatas[0] + self.virsotnes[i+1].koordinatas[0]) *
(self.virsotnes[i].koordinatas[1] - self.virsotnes[i+1].koordinatas[1])
        summa += s
      abs_summa = abs(summa)
      return abs_summa / 2

def galvena_programma():
  print('N-stūra virsotņu ievade')
  print('')

  v = []
  nsturis = Nsturis(v)

  while True:
    koord_x = float(input('Ievadiet x koordinātu --> '))
    koord_y = float(input('Ievadiet y koordinātu --> '))
    nsturis.pievienot_virsotni(koord_x, koord_y)
    print('')

    turp = input('Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> ')
    print('')
```

```
        if turp == 'p':
            break

    print('')
    print('Laukuma aprēķins')
    print('')

    print(f'Laukums: {nsturis.laukums()}')

if __name__ == '__main__':
    galvena_programma()
```

Testa piemērs(1)

```
N-stūra virsotņu ievade

Ievadiet x koordinātu --> 0
Ievadiet y koordinātu --> 0

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> t

Ievadiet x koordinātu --> 2
Ievadiet y koordinātu --> 0

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> t

Ievadiet x koordinātu --> 2
Ievadiet y koordinātu --> 2

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> t

Ievadiet x koordinātu --> 0
Ievadiet y koordinātu --> 2

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> p


Laukuma aprēķins

Laukums: 4.0
```

Testa piemērs(2)

```
N-stūra virsotņu ievade

Ievadiet x koordinātu --> 0
Ievadiet y koordinātu --> 0

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> t

Ievadiet x koordinātu --> 3
Ievadiet y koordinātu --> 0

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> p


Laukuma aprēķins

Laukums: Virsotņu skaits nav pietiekams laukuma aprēķinam.
```

Testa piemērs(3)

```
N-stūra virsotņu ievade

Ievadiet x koordinātu --> 0
Ievadiet y koordinātu --> 0

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> t

Ievadiet x koordinātu --> 3
Ievadiet y koordinātu --> 0

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> t

Ievadiet x koordinātu --> 0
Ievadiet y koordinātu --> 4

Vēlaties turpināt ievadi? (t - turpināt, p - pabeigt) --> p


Laukuma aprēķins

Laukums: 6.0
```