7. uzdevums

Sastādīt programmu, kas noskaidro, cik pilnu kvadrātu (1x1 vienība) satur gredzens, kura iekšējais rādiuss ir R1, bet ārējais rādiuss ir R2. Skaitļus R1 un R2 ievada lietotājs.

Kods:

```
# Programmas nosaukums: 7. uzd MPR10

# 7. uzdevums MPR10

# Uzdevuma formulējums: Sastādīt programmu, kas noskaidro, cik pilnu kvadrātu (1x1 vienība) satur gredzens, kura iekšējais rādiuss ir R1, bet ārējais rādiuss ir R2. Skaitļus R1 un R2 ievada lietotājs.

# Versija 1.0


# Importēsim math bibliotēku, lai izmantotu math.floor

import math


# Definēsim funkciju, kura pārbauda vai punkts piedēr gredzenām vai nēpiedēr


def bls(r, R, x, y):                    # nosaukums funkcijai ir bls - no angl. belongs (piedēr)
   if (r*r <= x*x + y*y) and (x*x + y*y <= R*R):
      return 1                          # jā punkts ir gredzenā iekšā, tad return 1
   else:
      return 0                          # jā punkts nav gredzenā, tad return 0


ammount = 0          # kvadrātu skaits gredzenā (tas ir kvadrātu skaititājs, sākuma ir 0 kvadrātu).


R = float(input("Ievadiet gredzena arējo rādiusu ==> ")) # Lielāis rādiuss (arējais)

r = float(input("Ievadiet riņķa iekšējo radiusu ==>  ")) # Mazais rādiuss (iekšējais)


for x in range(0, math.floor(R) + 1): # Paitonā range(0,n) = [0,n), tāpēc paņemsim +1
```

for y in range(0, math.floor(R) + 1): # Šeit cikls ir ciklā, lai visas iespējamās kombinācijas līdz R tiek pārbaudītas. (x=0 y=0 ; x=0 y=1 ; x=0 y=2 utt. ------ x=1 y=0 ; x=1 y=1 ; x=1 y=2 utt. -------- līdz beigām)

Pirmkārt pārbaudam visas y pie x=0, pēc tām visas y pie x=1 utt. Vispār pārbaudīsim R*R kvadrātus (pilnā pārlase).

if ((bls(r , R, x, y) == 1) and

(bls(r , R, x, y + 1) == 1) and

(bls(r , R, x + 1, y) == 1) and # Pārbaudam, vai visi četri punkti piedēr gredzenam

(bls(r , R, x + 1, y + 1) == 1)):

ammount = ammount+1 # ja piedēr, tad +1 kvadrātu skaititājam

Sākuma tiek pārbaudīts kvadrāts ar virsotnem (0 0) (0 1) (1 0) (1 1)

Tālak iterācijas y kļūst par 1 un tad tiek pārbaudīts kvadrāts ar virsotnem (0 1) (0 2) (1 1) (1 2) un tā tālāk uz augšu līdz R (līdz radiusa garumam).

Pēc tām x palielināsies par 1 arī (jo cikls ciklā) un tādēļ mes pārbaudisim visas kombinācijas.

print(str(ammount*4) + " pilno kvadrātu ir gredzenā")

Pārbaudījam tikai pirmo kvadrāntu (I). Riņķis ir simetrisks pret (0;0) un tāpēc pareizināsim ar 4 kvadrātu skaitu.

```
# Programmas nosaukums: 7. uzd MPR10
# 7. uzdevums MPR10
# Uzdevuma formulējums: Sastādīt programmu, kas noskaidro, cik pilnu kvadrātu (1x1 vienība) satur gredzens, kura iekšējais rādiuss ir R1, bet ārējais rādiuss ir R2. Skaitļus R1 un R2 ievada lietotājs.
# Versija 1.0

# Importēsim math bibliotēku, lai izmantotu math.floor
import math

# Definēsim funkciju, kura pārbauda vai punkts piedēr gredzenām vai nēpiedēr

def bls(r, R, x, y):                                    # nosaukums funkcijai ir bls - no angl. belongs (piedēr)
    if (r*r <= x*x + y*y) and (x*x + y*y <= R*R):
        return 1                                        # jā punkts ir gredzenā iekšā, tad return 1
    else:
        return 0                                        # jā punkts nav gredzenā, tad return 0


ammount = 0             # kvadrātu skaits gredzenā (tas ir kvadrātu skaitītājs, sākuma ir 0 kvadrātu).


R = float(input("Ievadiet gredzena arējo rādiusu ==> ")) # Lielāis rādiuss (arējais)
r = float(input("Ievadiet riņķa iekšējo radiusu ==>  ")) # Mazais rādiuss (iekšējais)


for x in range(0, math.floor(R) + 1): # Paitonā range(0,n) = [0,n), tāpēc paņemsim +1

    for y in range(0, math.floor(R) + 1): # Šeit cikls ir ciklā, lai visas iespējamās kombinācijas līdz R tiek pārbaudītas. ( x=0 y=0 ; x=0 y=1 ; x=0 y=2 utt. ------ x=1 y=0 ; x=1 y=1 ; x=1 y=2 utt. -------- līdz beigām)


        # Pirmkārt pārbaudam visas y pie x=0, pēc tām visas y pie x=1 utt. Vispār pārbaudīsim R*R kvadrātus (pilnā pārlase).
        if ((bls(r , R, x, y) == 1) and
           (bls(r , R, x, y + 1) == 1) and
           (bls(r , R, x + 1, y) == 1) and    # Pārbaudam, vai visi četri punkti piedēr gredzenam
           (bls(r , R, x + 1, y + 1) == 1)):
            ammount = ammount+1             # ja piedēr, tad +1 kvadrātu skaititājam

            # Sākuma tiek pārbaudīts kvadrāts ar virsotnem (0 0) (0 1) (1 0) (1 1)
            # Tālak iterācijas y kļūst par 1 un tad tiek pārbaudits kvadrāts ar virsotnem (0 1) (0 2) (1 1) (1 2) un tā tālāk uz augšu līdz R (līdz radiusa garumam).
            # Pēc tām x palielināsies par 1 arī (jo cikls ciklā) un tādēļ mes pārbaudisim visas kombinācijas.

print(str(ammount*4) + " pilno kvadrātu ir gredzenā") # Pārbaudījam tikai pirmo kvadrāntu (I). Riņķis ir simetrisks pret (0;0) un tāpēc pareizināsim ar 4 kvadrātu skaitu.
```

Testa piemēri:

1)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet riņķa iekšējo radiusu ==>  0
60 pilno kvadrātu ir gredzenā
```

2)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet riņķa iekšējo radiusu ==>  5
0 pilno kvadrātu ir gredzenā
```

3)

```
Ievadiet gredzena arējo rādiusu ==> 3.5
Ievadiet riņķa iekšējo radiusu ==>  2
8 pilno kvadrātu ir gredzenā
```

4)

```
Ievadiet gredzena arējo rādiusu ==> 4
Ievadiet riņķa iekšējo radiusu ==>  3
0 pilno kvadrātu ir gredzenā
```

5)

```
Ievadiet gredzena arējo rādiusu ==> 4.999
Ievadiet riņķa iekšējo radiusu ==>  3.999
0 pilno kvadrātu ir gredzenā
```

6)

```
Ievadiet gredzena arējo rādiusu ==> 2.10
Ievadiet riņķa iekšējo radiusu ==>  2
0 pilno kvadrātu ir gredzenā
```

Cikla princips: