1. uzdevums

Sastādīt programmu, kas uzzīmē dambretes galdiņu.

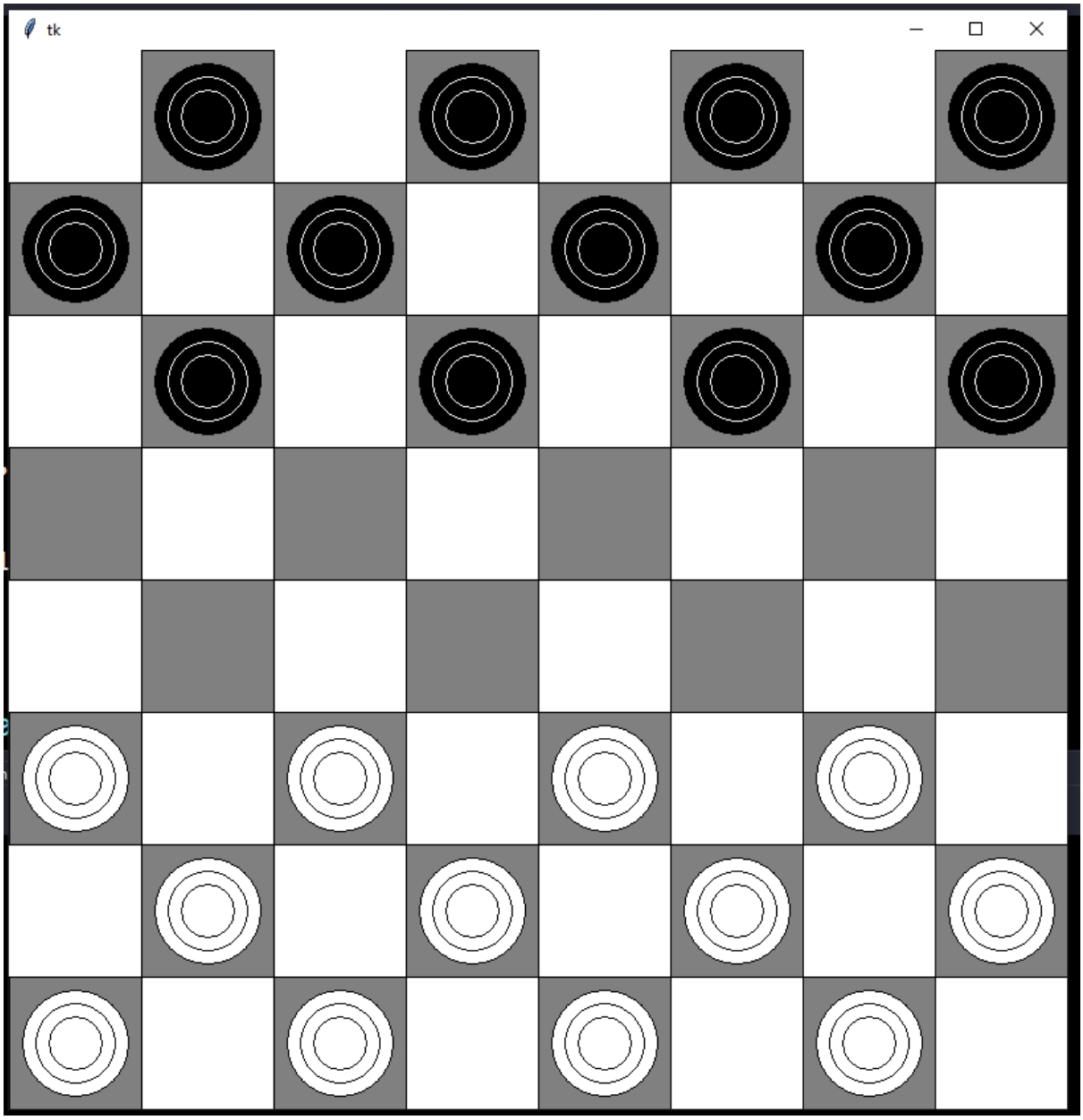
Kods:

```
# Programmas nosaukums: 1. uzd MPR10

# 1. uzdevums MPR10

# Uzdevuma formulējums: Sastādīt programmu, kas uzzīmē dambretes galdiņu.

# Versija 1.0


import tkinter

from tkinter import ttk



# Loga atribūti


logs = tkinter.Tk()

logs.geometry("800x800")


# Kanvas novietošana


kanva = tkinter.Canvas(logs, bg="white", height=900, width=900)

kanva.place(x=-100, y=-100)


kanva.create_rectangle (0,0,900,900, fill = "white")


for i in range (1,9):

    for j in range (1,9):

        if (i + j)%2 == 1:

            kanva.create_rectangle(i*100, j*100, i*100+100, j*100+100, fill="gray")


            if j < 4:
```

```
            kanva.create_oval(i*100+10, j*100+10, i*100+90, j*100+90, fill="black")

            kanva.create_oval(i*100+20, j*100+20, i*100+80, j*100+80, outline="white")
            kanva.create_oval(i*100+30, j*100+30, i*100+70, j*100+70, outline="white")


        if j > 5:
            kanva.create_oval(i*100+10, j*100+10, i*100+90, j*100+90, fill="white")

            kanva.create_oval(i*100+20, j*100+20, i*100+80, j*100+80, outline="black")
            kanva.create_oval(i*100+30, j*100+30, i*100+70, j*100+70, outline="black")



# Obligāta rindiņa, lai logs būtu redzāms visu laiku
logs.mainloop()
```

Testa piemēri:

1)



2. uzdevums

Sastādīt programmu, kas lietotāja ievadīto vārdu izkārto kā parādīts piemērā. Piemērām ievada 1234567890, bet izvada šādi:

1234567890

2345678901

3456789012

4567890123

5678901234

6789012345

7890123456

8901234567

9012345678

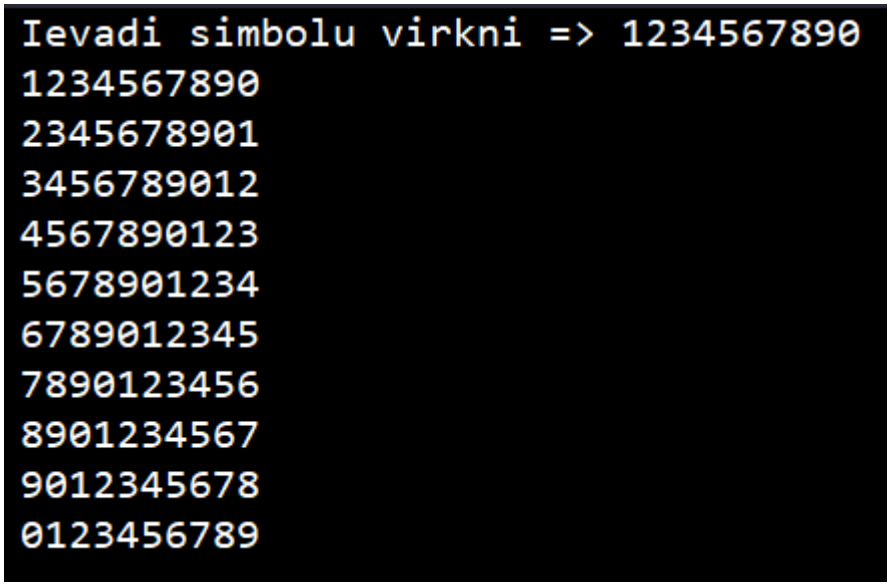
0123456789

Kods:

```
# Programmas nosaukums: 2. uzd MPR10

# 2. uzdevums MPR10

# Uzdevuma formulējums: Sastādīt programmu, kas lietotāja ievadīto vārdu izkārto kā parādīts piemērā.

# Versija 1.0



x=input("Ievadi simbolu virkni => ")

n=len(x)

for i in range (n):

    print(x[i:n]+x[0:i])
```

Testa piemēri:

1)



2)

```
Ievadi simbolu virkni => 1
1
```

3)

```
Ievadi simbolu virkni => abcd
abcd
bcda
cdab
dabc
```

4)

```
Ievadi simbolu virkni => abcdefgh
abcdefgh
bcdefgha
cdefghab
defghabc
efghabcd
fghabcde
ghabcdef
habcdefg
```

3. uzdevums

Sastādīt programmu, kas lietotāja ievadīto vārdu izkārto kā parādīts piemērā. Piemērām, ievada Viesturs, bet izvada šādi:

Viesturs

i r

e u

s t

t s

u e

r i

srutseiV

Kods:

```
# Programmas nosaukums: 3. uzd MPR10

# 3. uzdevums MPR10

# Uzdevuma formulējums: Sastādīt programmu, kas lietotāja ievadīto vārdu izkārto ka parādīts piemērā.

# Versija 1.0


x = str(input("Ievādi vārdu => "))


n = len(x)


print(x)


for i in range (1, n-1):
    sv = x[i]


    sv = sv + (n-2)*" "


    sv = sv + x[n-1-i]
    print(sv)


sv = ""


for i in x:
    sv = i + sv
print(sv)
```

Testa piemēri:

1)

```
Ievādi vārdu => Viesturs
Viesturs
i       r
e       u
s       t
t       s
u       e
r       i
srutseiV
```

2)

```
Ievādi vārdu => Vladislavs
Vladislavs
l         v
a         a
d         l
i         s
s         i
l         d
a         a
v         l
svalsidalV
```

3)

```
Ievādi vārdu => Volters
Volters
o     r
l     e
t     t
e     l
r     o
sretloV
```

4)

```
Ievādi vārdu => Smukums
Smukums
m     m
u     u
k     k
u     u
m     m
smukumS
```

5)

```
Ievādi vārdu => Alus ari ira aluS
Alus ari ira aluS
l               u
u               l
s               a

a               a
r               r
i               i

i               i
r               r
a               a

a               s
l               u
u               l
Sula ari ira sulA
```

6)

```
Ievādi vārdu => Skaidrs
Skaidrs
k     r
a     d
i     i
d     a
r     k
srdiakS
```

4. uzdevums

Sastādīt programmu, kas nodrukā visus pirmskaitļus no 1 līdz N. Skaitli N ievada lietotājs.

Kods:

```
# Programmas nosaukums: 4. uzd MPR10

# 4. uzdevums MPR10

# Uzdevuma formulējums: Sastādīt programmu, kas nodrukā visus pirmskaitļus no 1 līdz N. Skaitli N ievada lietotājs.

# Versija 1.0


import math

import sys


x = int(input("Programmā nodrukā visus pirmskaitļus no 1 līdz N. Ievadiet naturālo skaitļi N ==> "))


if x < 1:

   print("Tas nav naturāls skaitlis")

   sys.exit(0)


if x == 1:

   print("1 nav naturāls skaitlis")

   sys.exit(0)


for a in range (1, x+1):

   if a > 1:

      for i in range (2, a):

         if (a % i) == 0:

            break

      else:

         print(a)
```

Testa piemēri:

1)

```
Programmā nodrukā visus pirmskaitļus no 1 līdz N. Ievadiet naturālo skaitļi N ==> 100
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
```

2)

```
Programmā nodrukā visus pirmskaitļus no 1 līdz N. Ievadiet naturālo skaitļi N ==> 1
1 nav naturāls skaitlis
```

3)

```
Programmā nodrukā visus pirmskaitļus no 1 līdz N. Ievadiet naturālo skaitļi N ==> 0
Tas nav naturāls skaitlis
```

4)

```
Programmā nodrukā visus pirmskaitļus no 1 līdz N. Ievadiet naturālo skaitļi N ==> -151254
Tas nav naturāls skaitlis
```

5)

```
Programmā nodrukā visus pirmskaitļus no 1 līdz N. Ievadiet naturālo skaitļi N ==> 2
2
```

6)

```
Programmā nodrukā visus pirmskaitļus no 1 līdz N. Ievadiet naturālo skaitļi N ==> 130
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
101
103
107
109
113
127
```

5. uzdevums

Sastādīt programmu, kas nodrukā visus laimīgos loterijas biļešu numurus.

Ja zināms, ka loterijas biļešu numuri ir četrciparu skaitļi no 0000 līdz 9999, bet par laimīgiem tie uzskatīti tie numuri, kas apmierina šādus nosacījumus:

1) visi cipari ir dažādi.

2) Pirmo divu un pēdējo divu ciparu veidoto skaitļu summa ir vienāda ar visu četru ciparu reizinājumu.

Kods:

```
# Programmas nosaukums: 5. uzd MPR10

# 5. uzdevums MPR10

# Uzdevuma formulējums: Sastādīt programmu, kas nodrukā visus laimīgos loterijas biļešu numurus.
```

Ja zināms, ka loterijas biļešu numuri ir četrciparu skaitļi no 0000 līdz 9999, bet par laimīgiem tie uzskatīti

tie numuri, kas apmierina šādus nosacījumus:

1) visi cipari ir dažādi.

2) Pirmo divu un pēdējo divu ciparu veidoto skaitļu summa ir vienāda ar visu četru ciparu reizinājumu.

Versija 1.0

```
for a in range(0, 10) :
    for b in range(0, 10) :
        for c in range(0, 10) :
            for d in range(0, 10) :


                if ((a!=0) and (b!=0) and
                    (c!=0) and (d!=0) and
                    (a!=b) and (a!=c) and
                    (a!=d) and (b!=c) and
                    (b!=d) and (c!=d) and
                    (10*a+b+10*c+d==a*b*c*d)) :
                    print (str(a) + str(b) +
                           str(c) + str(d))
```

Testa piemēri:

1)

```
1296
1692
6183
6381
8163
8361
9216
9612
```

6. uzdevums

Sastādīt programmu, kas noskaidro, cik pilnu kvadrātu (1x1 vienība) satur riņķis ar rādiusu R. Skaitli R ievada lietotājs.

Kods:

```
# Programmas nosaukums: 6. uzd MPR10

# 6. uzdevums MPR10

# Uzdevuma formulējums: Sastādīt programmu, kas noskaidro, cik pilnu kvadrātu (1x1 vienība) satur riņķis ar rādiusu R. Skaitli R ievada lietotājs.

# Versija 1.0


import math


r=float(input("Ievadiet riņķa rādiusu ==> "))

r1=int(r)

s=0


for i in range (1, r1):
   x=round(math.sqrt(r*r-i*i))
   if x*x+i*i==r*r:
      s=s+4*x
   else:
      s=s+int(math.sqrt(r*r-i*i))*4


print(str(s) + " pilno kvadrātu 1x1 satur riņķis ar rādiusu " + str(r) + " vienības")
```

Testa piemēri:

1)

```
Ievadiet riņķa rādiusu ==> 5
60 pilno kvadrātu 1x1 satur riņķis ar rādiusu 5.0 vienības
```

2)

```
Ievadiet riņķa rādiusu ==> 1
0 pilno kvadrātu 1x1 satur riņķis ar rādiusu 1.0 vienības
```

3)

```
Ievadiet riņķa rādiusu ==> -12
0 pilno kvadrātu 1x1 satur riņķis ar rādiusu -12.0 vienības
```

4)

```
Ievadiet riņķa rādiusu ==> 5.000001
60 pilno kvadrātu 1x1 satur riņķis ar rādiusu 5.000001 vienības
```

5)

```
Ievadiet riņķa rādiusu ==> 4.9999999999
44 pilno kvadrātu 1x1 satur riņķis ar rādiusu 4.9999999999 vienības
```

6)

```
Ievadiet riņķa rādiusu ==> 4.44
40 pilno kvadrātu 1x1 satur riņķis ar rādiusu 4.44 vienības
```

7)

```
Ievadiet riņķa rādiusu ==> 4.5
44 pilno kvadrātu 1x1 satur riņķis ar rādiusu 4.5 vienības
```

8)

```
Ievadiet riņķa rādiusu ==> 0.7
0 pilno kvadrātu 1x1 satur riņķis ar rādiusu 0.7 vienības
```

9)

```
Ievadiet riņķa rādiusu ==> 1.999999999
0 pilno kvadrātu 1x1 satur riņķis ar rādiusu 1.999999999 vienības
```

10)

```
Ievadiet riņķa rādiusu ==> 2
4 pilno kvadrātu 1x1 satur riņķis ar rādiusu 2.0 vienības
```

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


import math

import sys


def check_squares(a, b):

    n = 0

    if r1*r1 >= (a+1)*(a+1) + b*b >= r2*r2 and r1*r1 >= a*a + (b+1)*(b+1) >= r2*r2 and r1*r1 >= (a+1)*(a+1) + (b+1)*(b+1) >= r2*r2:

        n+=1


    if r1*r1 >= (a+1)*(a+1) + b*b >= r2*r2 and r1*r1 >= (a+1)*(a+1) + (b-1)*(b-1) >= r2*r2 and r1*r1 >= a*a + (b-1)*(b-1) >= r2*r2:

        n+=1


    if r1*r1 >= a*a + (b+1)*(b+1) >= r2*r2 and r1*r1 >= (a-1)*(a-1) + b*b >= r2*r2 and r1*r1 >= (a-1)*(a-1) + (b+1)*(b+1) >= r2*r2:

        n+=1


    if r1*r1 >= a*a + (b-1)*(b-1) >= r2*r2 and r1*r1 >= (a-1)*(a-1) + b*b >= r2*r2 and r1*r1 >= (a-1)*(a-1) + (b-1)*(b-1) >= r2*r2:

        n+=1


    return n
```

```
r1 = float(input("Ievadiet gredzena arējo rādiusu ==> "))
r2 = float(input("Ievadiet gredzena iekšējo rādiusu ==> "))



if r1 < 0 or r2 < 0 :
    print("Dati nav ievadīti atbilstoši nosacījumiem")
    sys.exit(0)



a = math.floor(r1)
b = -math.floor(r1)

number_of_squares = 0

while a >= -r1:
    b = -math.floor(r1)
    while b <= r1:
        if r1*r1 >= a*a + b*b >= r2*r2:


            number_of_squares += check_squares(a, b)
        b+=1
    a-=1

print(str(int(number_of_squares/4)) + " pilno kvadrātu ir gredzenā")
```

Testa piemēri:

1)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet gredzena iekšējo rādiusu ==> 8
0 pilno kvadrātu ir gredzenā
```

2)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet gredzena iekšējo rādiusu ==> -1
Dati nav ievadīti atbilstoši nosacījumiem
```

3)

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

4)

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

5)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet gredzena iekšējo rādiusu ==> 4
0 pilno kvadrātu ir gredzenā
```

6)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet gredzena iekšējo rādiusu ==> 3
24 pilno kvadrātu ir gredzenā
```

7)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet gredzena iekšējo rādiusu ==> 2
44 pilno kvadrātu ir gredzenā
```

8)

```
Ievadiet gredzena arējo rādiusu ==> 5
Ievadiet gredzena iekšējo rādiusu ==> 1
56 pilno kvadrātu ir gredzenā
```

9)

```
Ievadiet gredzena arējo rādiusu ==> 5.5
Ievadiet gredzena iekšējo rādiusu ==> 3.5
24 pilno kvadrātu ir gredzenā
```

10)

```
Ievadiet gredzena arējo rādiusu ==> 2.8
Ievadiet gredzena iekšējo rādiusu ==> 1
8 pilno kvadrātu ir gredzenā
```

11)

```
Ievadiet gredzena arējo rādiusu ==> 2.999999
Ievadiet gredzena iekšējo rādiusu ==> 1
12 pilno kvadrātu ir gredzenā
```

12)

```
Ievadiet gredzena arējo rādiusu ==> 3
Ievadiet gredzena iekšējo rādiusu ==> 1
12 pilno kvadrātu ir gredzenā
```

13)

```
Ievadiet gredzena arējo rādiusu ==> 3.4
Ievadiet gredzena iekšējo rādiusu ==> 1
20 pilno kvadrātu ir gredzenā
```