

# Näidis projekti valmimisest

## Sissejuhatuseks

See näidis on mõeldud selleks, et anda mõtteid, kuidas efektiivsemalt projekti kirjutamisega tegeleda. Teeme läbi väikese näidise algusest lõpuni – idee faasist kuni toimiva graafilise rakenduseni. Lisatud on ka kopeeritavad koodijupid, mida võid Thonny abil jooksutada.

Näidise teemaks on sõbralik miljonimäng, kus ei kaota kohe esimese valega, vaid loendame õiged vastused.

## Planeerimise faas

Esimene asi, millega tegeleda, oleks see, et tuleks mõelda läbi kõik osad, mida projekti valmimiseks on tarvis teha. Eriti oluline on see sel juhul, kui töö käib suurema meeskonnaga – see aitab paremini väikesed ülesanded inimeste vahel ära jagada.

Suure ülesande väikesteks juppideks jaotamine on hea ka teisel põhjusel. Suur projekt, mis algul tundub jube raske, võib osadeks jaotatuna hoopis jõukohasem tunduda. Motivatsiooni hoidmiseks on see kindlasti kasulik. :)

Nüüd siis näidise kallale! Mõtleme läbi, mida on tarvis, et miljonimäng toimiks.

- Peab saama küsimusele vastata.
  - 4 varianti, 1 õige.
- Peab saama küsimusi kuskilt sisse lugeda.
  - Teeme nii, et küsimused tulevad eraldi failist.
- Mäng peab küsima hulga küsimusi.
- Mäng peab teatama, milline on tulemus.
- Küsimus ning lõpptulemus peab olema esitatud graafiliselt.
  - Nagu õiges miljonimängus küsitakse – valides a, b, c või d.

## Teostuse faas ilma graafikata

Nüüd on aeg koodi kirjutamise kallale asuda! Eelmises peatükis jagasime mängu 5-ks osaks, millest praegu katame esimesed 4. Selle asemel, et kogu mängu suure kompotina kirjutama hakata, proovime seda hoopis teha nende väiksemate etappidena.

## Küsimusele vastamine

Esimeseks ülesandeks on küsimusele vastamise tööle saamine. Mängija peab saama valida 4 varaindi vahel, millest 1 on õige. Tundub ju palju väiksem ja lihtsam ülesanne, kui "miljonimängu kirjutamine"? :)

Teeme alguses katsetuseks lihtsa implementatsiooni ilma igasuguste klasside ja funktsioonideta. Teeme asja ainult nii palju huvitavamaks, et segame vastusevariandid juhuslikult.

```

from random import shuffle

küsimus = 'Mis aastal Tartu Ülikool rajati?'
õige = '1632'
vale1 = '1991'
vale2 = '2006'
vale3 = '2015'

variandid = [õige, vale1, vale2, vale3]
shuffle(variandid)

print(küsimus)
print('Võimalikud vastused on:')
for variant in variandid:
    print(variant)

vastus = input('Sisesta vastus: ')
if vastus == õige:
    print('Õige vastus!')
else:
    print('Vale vastus!')

```

Nii! Esimene ülesanne viiest ongi lahendatud. Et enda elu hiljem lihtsamaks teha, muudame küsimuse küsimise funktsiooniks, mis tagastab, kas kasutaja andis õige vastuse.

```

from random import shuffle

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandid = [õige, vale1, vale2, vale3]
    shuffle(variandid)

    print(küsimus)
    print('Võimalikud vastused on:')
    for variant in variandid:
        print(variant)

    vastus = input('Sisesta vastus: ')
    if vastus == õige:
        return True
    else:
        return False

küsimus = 'Mis aastal Tartu Ülikool rajati?'
v1 = '1632'
v2 = '1991'
v3 = '2006'
v4 = '2015'

vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
if vastas_õigesti:
    print('Õige vastus!')
else:
    print('Vale vastus!')

```

See uus koodijupp teeb praegu täpselt sama, mida eelmine. Erinevus on selles, et hiljem on selle abil hoopis lihtsam suuremat hulka küsimusi küsida.

## Küsimuste sisse lugemine

Esiteks defineerime ära, kuidas me soovime küsimusi failis hoida. Ütleme, et küsimused on failis "küsimused.txt".

Määrame nende formaadiks "küsimus;õigevastus;valevastus1;valevastus2;valevastus3". Ehk üks näide sellise faili sisust oleks järgmine.

```
Mis on elevant?;imetaja;kala;lind;kahepaikne
Mis aastal Tartu Ülikool rajati?;1632;1991;2006;2015
Mitu maakonda on Eestis?;15;3;99;2
```

Siin oli ka näha eelmises peatükis küsitava küsimuse vorm failis.

Selle ülesande lahenduseks võib olla näitks järgnev koodijupp. Siin segame samuti küsimused, et põnevam oleks.

```
from random import shuffle

küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)
```

Proovime ruttu ka läbi mõelda, mis kujul meil need küsimused peale sisse lugemist nüüd on.

Muutuja *küsimused* on siin list. Me lisame sinna ridu, millel on rankendatud *.split()* funktsiooni, mis tagastab samuti listi. Seega meil on list listidest, mille sees on sõned.

Sellele samale failile saame väärtuseks järgneva

```
[['Mitu maakonda on Eestis?', '15', '3', '99', '2'], ['Mis on
elevant?', 'imetaja', 'kala', 'lind', 'kahepaikne'], ['Mis aastal
Tartu Ülikool rajati?', '1632', '1991', '2006', '2015']]
```

## Hulga küsimuste küsimine.

Läheneme sellele ülesandele samuti nagu eelmistele – alustame lihtsalt ning siis täiendame. Proovimegi esimese asjana esimese ja teise ülesande tulemused kokku sobitada.

Alustame lihtsalt ning küsime lihtsalt esimese sisse loetud küsimuse. Sellele vastav kood oleks järgmine.

```
from random import shuffle

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandid = [õige, vale1, vale2, vale3]
    shuffle(variandid)

    print(küsimus)
    print('Võimalikud vastused on:')
    for variant in variandid:
        print(variant)

    vastus = input('Sisesta vastus: ')
    if vastus == õige:
        return True
    else:
        return False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

#Võtame välja esimese küsimuse.
#Näiteks võib see olla: ['Mitu maakonda on Eestis?', '15', '3', '99', '2']
esimene = küsimused[0]

#Võtame listi elemendid eraldi muutujatesse,
#et need küsimise funktsioonile edasi anda.
küsimus, v1, v2, v3, v4 = esimene

vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
if vastas_õigesti:
    print('Õige vastus!')
else:
    print('Vale vastus!')
```

Siin on näha, et väga palju me muutma ei pidanudki. Saime peaaegu et niisama mõlema ülesande lahendused üksteisele otsa panna. Muutsime ainult seda, et käsitsi küsimuse/vastuste sisestamise asemel võtame failist esimese juhusliku.

Nüüd on meil olemas peaaegu toimiv mäng. Lisame nüüd selle, et tsükliga küsitakse kasutajalt kõik sisse loetud küsimused.

```

from random import shuffle

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandidid = [õige, vale1, vale2, vale3]
    shuffle(variandidid)

    print(küsimus)
    print('Võimalikud vastused on:')
    for variant in variandidid:
        print(variant)

    vastus = input('Sisesta vastus: ')
    if vastus == õige:
        return True
    else:
        return False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

#Käib üle kõik küsimused.
for element in küsimused:
    küsimus, v1, v2, v3, v4 = element

    vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
    if vastas_õigesti:
        print('Õige vastus!')
    else:
        print('Vale vastus!')

```

Nüüd siis küsib arvuti kasutajalt järjest kõik küsimused. Käiakse küsimused *for* tsükliga üle.

## Tulemuse meeles pidamine.

Viimane funktsioon, mille mängule lisame, on tulemuse näitamine. Tahame kasutajale käigupealt näidata, mitu küsimust on õigesti vastatud.

Selleks defineerime lihtsalt uue muutja *punktide\_arv* ning salvestame sinna jooksvalt tulemuse.

```
from random import shuffle

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandid = [õige, vale1, vale2, vale3]
    shuffle(variandid)

    print(küsimus)
    print('Võimalikud vastused on:')
    for variant in variandid:
        print(variant)

    vastus = input('Sisesta vastus: ')
    if vastus == õige:
        return True
    else:
        return False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

punktide_arv = 0
#Käib üle kõik küsimused.
for element in küsimused:
    küsimus, v1, v2, v3, v4 = element

    vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
    if vastas_õigesti:
        print('Õige vastus!')
        punktide_arv += 1
    else:
        print('Vale vastus!')
    print('Praegune punktide arv on', punktide_arv)

print('Mäng läbi! Kogusid', punktide_arv, 'punkti.')
```

Nüüd on meil olemas täitsa mängitav mäng! Järgmises peatükis proovime lahendust täiendada graafikaga.

# Graafikaga

## Planeerimine

Esiteks oleks graafika lisamisel mõistlik samamoodi alustada plaanist nagu enne esimese nelja punkti jaoks alguses.

Proovime eraldi valmis kirjutada kõik tarviliku graafika ning seejärel proovime selle sobitada varasemasse tekstipõhisesse lahendusse.

Meil on tarvis kirjutada:

- Vaade õige vastuse puhul õnnitlemiseks (näitab ka punktisumma).
- Vaade, mis teavitab valest vastusest (näitab ka punktisumma).
- Vaade lõpus punktide summa näitamiseks.
- Vaade küsimuse küsimiseks.
  - Nähtaval küsimus ja 4 varianti vastamiseks (a,b,c,d).

# Teostus

## Õige puhul õnnitlemine

Alustame kõige lihtsamast. Kirjutame funktsiooni selleks, et kasutajat õnnitleda.

Teeme näiteks nii, et on kirjas "Õige vastus!", punktisumma ning tekst "Vajuta ENTER, et jätkata."

```
import pygame, sys

# Initsialiseerime Pygame ja tekitame akna.
pygame.init()
lava = pygame.display.set_mode([640,480])

def õnnitle(punkte):
    #Tumesinine taust.
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Tekstid, hallika värviga.
    txt1 = font.render("Õige vastus!", 1, (200,200,200))
    txt2 = font.render(str(punkte)+' punkti!', 1, (200,200,200))
    txt3 = font.render("Vajuta ENTER, et jätkata!", 1, (200,200,200))
    #Joonistame numbritega määratud koordinaatidele tekstid.
    lava.blit(txt1, (100, 100))
    lava.blit(txt2, (100, 200))
    lava.blit(txt3, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                if i.key == pygame.K_RETURN:
                    ootame = False
```

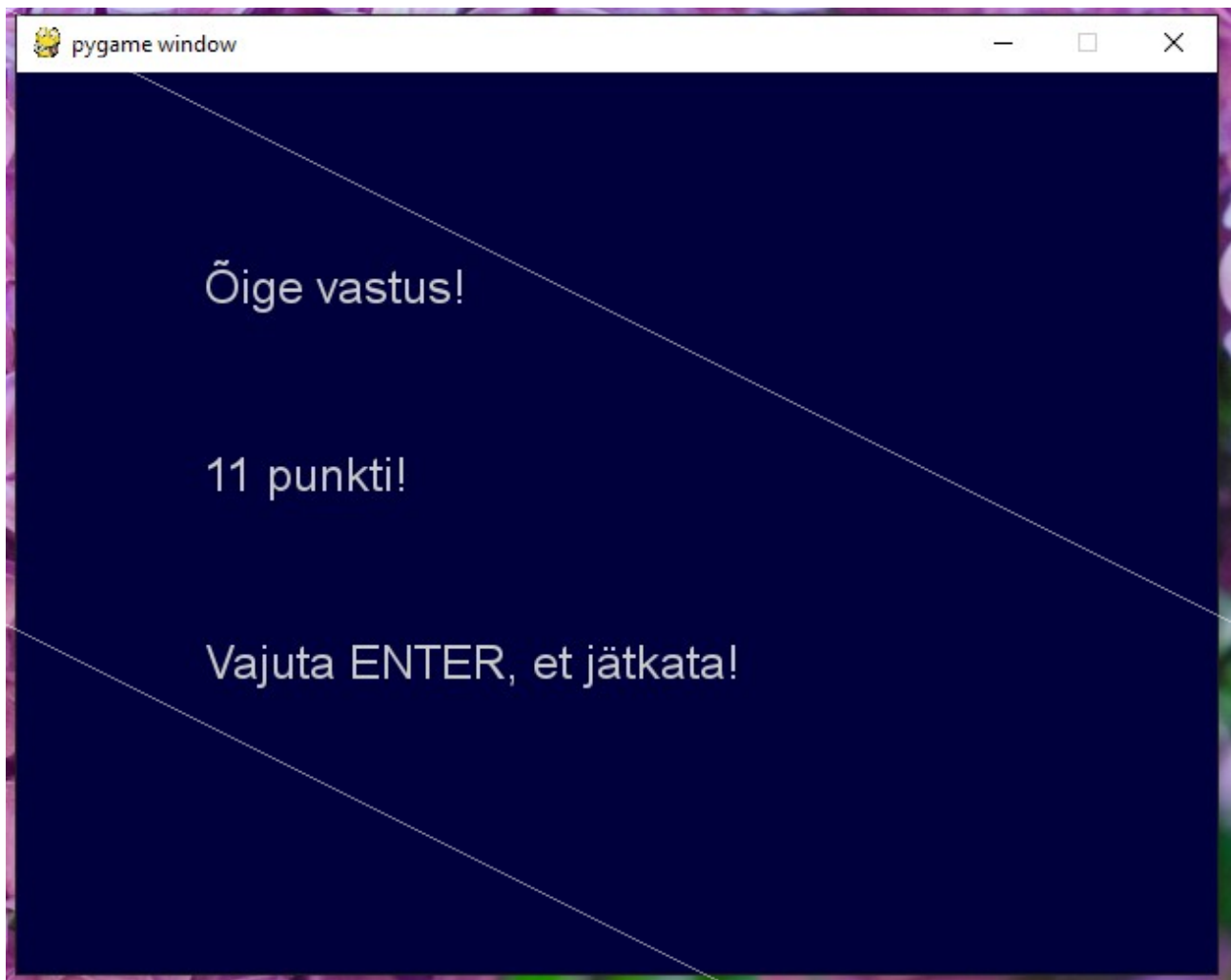
See kood käitub nii, et esiteks kuvatakse ekraanile tekstid ning seejärel jääb *while* tsüklil ootama kas sulgemist või enteri vajutust. Sulegmise loomulikult väljab programmist. Enteri vajutamine aga lõpetab tsükli ning mäng jätkub väljaspool *õnnitle()* funktsiooni.

Proovime mis tulemus on järgmisel koodujupil lõppu lisatuna.

```
õnnitle(11)
pygame.quit()
```

Tulemus on järgneval lehel.





Kunstipärasuse auhinda ilmselt ei võida, kuid ajab praegu näidisena asja ära. :)

### **Teade valest vastusest ning teade lõpus.**

Järgmisena võib tekkida kange tahtmine kirjutada selle peatüki pealkirjas olevate asjade tegemiseks uued funktsioonid. Aga see on ju üsna tüütu? Ainus mis meil on tarvis muuta, on teate tekst.

Miks mitte hoopis muuta õnnitlemise funktsiooni selliselt, et see tekst tuleb hoopis argumendina!? Nii polegi tarvis täitsa uusi funktsioone defineerida. Tuletame meelde *õnnitle()* funktsiooni algusosa, kus on näha tekst "Õige vastus!":

```
def õnnitle(punkte):
    #Tumesinine taust.
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Tekstid, hallika värviga.
    txt1 = font.render("Õige vastus!", 1, (200,200,200))
```

Muudame selle teksti argumendina sisestatavaks. Muudame ka funktsiooni nime nüüd üldisemaks. Tulemus on järgneval lehel.

```
def teata(ekraan, punkte, tekst):
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

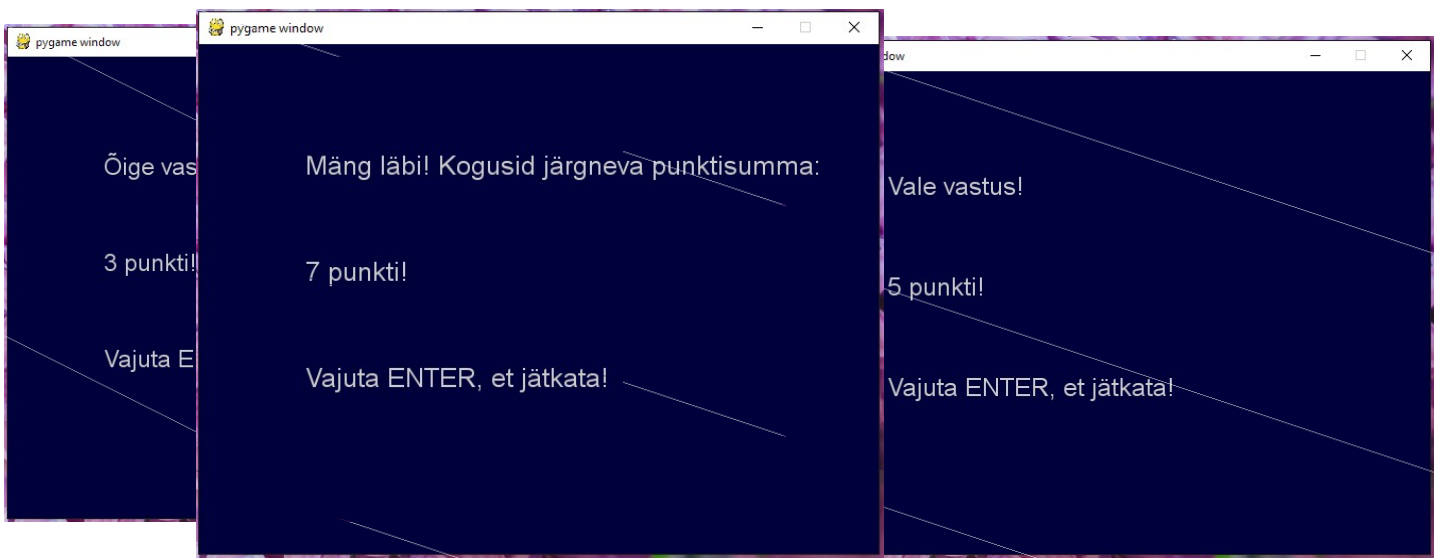
    #Teade muutujast tekst, hallika värviga.
    txt1 = font.render(tekst, 1, (200,200,200))
    txt2 = font.render(str(punkte)+' punkti!', 1, (200,200,200))
    txt3 = font.render("Vajuta ENTER, et jätkata!", 1, (200,200,200))
    #Joonistame numbritega määratud koordinaatidele teksti.
    lava.blit(txt1, (100, 100))
    lava.blit(txt2, (100, 200))
    lava.blit(txt3, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                if i.key == pygame.K_RETURN:
                    ootame = False
```

Seda saame nüüd juba kõigi kolme esimese graafilise nõude täitmiseks kasutada! Proovime seda järgneva koodijupiga.

```
teata(3, "Õige vastus!")
teata(5, "Vale vastus!")
teata(7, "Mäng läbi! Kogusid järgneva punktisumma:")
pygame.quit()
```

Kõik teated on toimivad.



## Küsimuse graafika

On jäänud teha vaid viimane osa - graafika küsimusele.

Lahendame selle nii, et kirjutame funktsiooni, mille argumentideks on küsimused, mille vahel kasutaja saab valida. Funktsioon tagastab valitud küsimuse. Proovime ka selle algul eraldi valmis kirjutada.

Järgnev funktsioon teeb täpselt seda.

```
#lause on küsimus ise
#a,b,c,d on neile tähtedele vastavad vastused.
def küsi_graafiliselt(lause, a, b, c, d):
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Argumendist küsimuse tekst, hallika värviga.
    txt_kys = font.render(lause, 1, (200,200,200))
    #Argumentidest vastuste tekstid, hallika värviga.
    txt_a = font.render('a' '+a, 1, (200,200,200))
    txt_b = font.render('b' '+b, 1, (200,200,200))
    txt_c = font.render('c' '+c, 1, (200,200,200))
    txt_d = font.render('d' '+d, 1, (200,200,200))

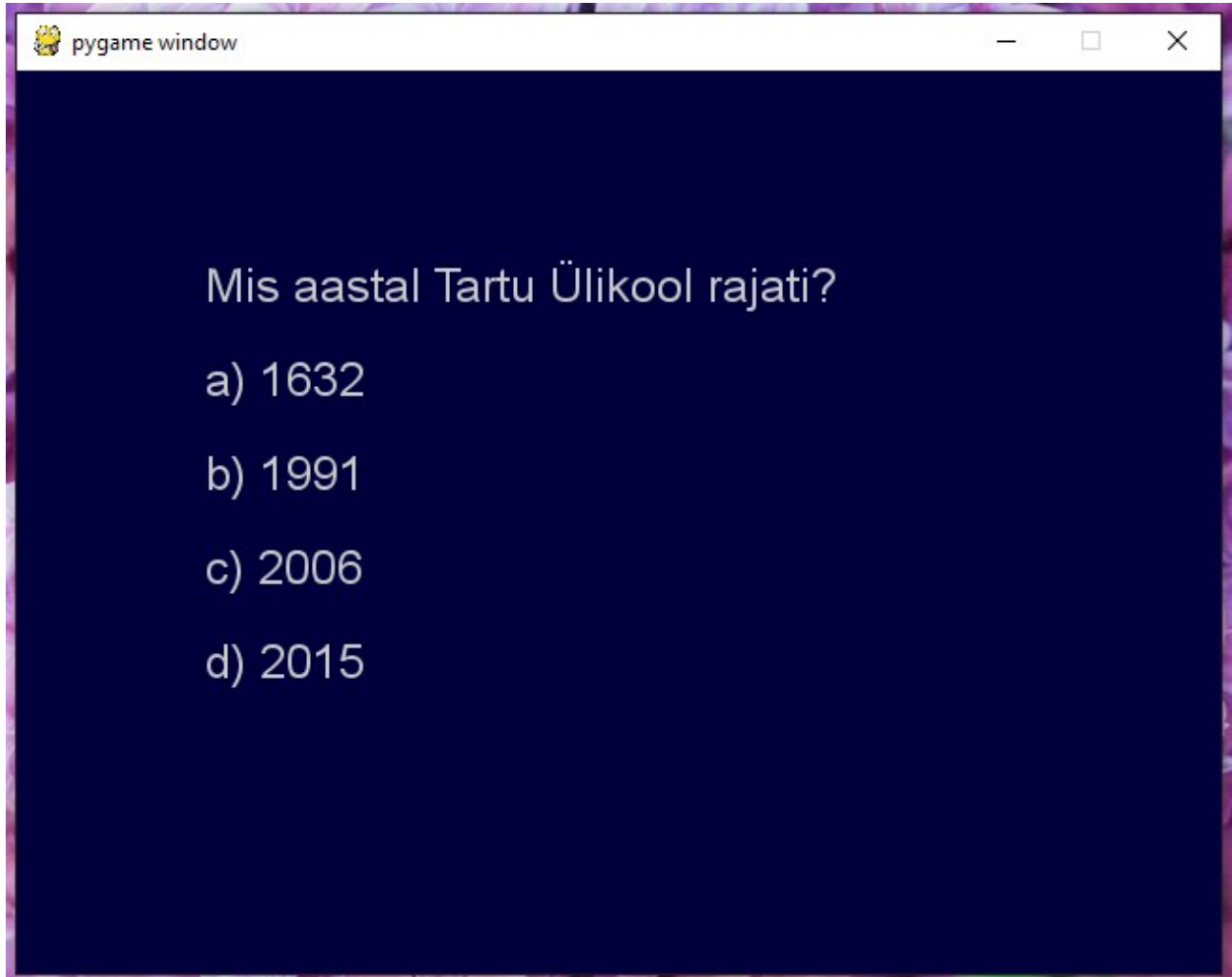
    #Joonistame numbritega määratud koordinaatidele teksti.
    lava.blit(txt_kys, (100, 100))
    lava.blit(txt_a, (100, 150))
    lava.blit(txt_b, (100, 200))
    lava.blit(txt_c, (100, 250))
    lava.blit(txt_d, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                #Vaatame, millist kasutaja vajutas.
                #Seejärel tagastame vastava tähe vastuse.
                if i.key == pygame.K_a:
                    return a
                if i.key == pygame.K_b:
                    return b
                if i.key == pygame.K_c:
                    return c
                if i.key == pygame.K_d:
                    return d
```

Katsetame näiteks järgneva koodijupiga funktsiooni toimimist.

```
küsimus = 'Mis aastal Tartu Ülikool rajati?'  
v1 = '1632'  
v2 = '1991'  
v3 = '2006'  
v4 = '2015'  
  
küsi_graafiliselt(küsimus, v1, v2, v3, v4)  
pygame.quit()
```

Saame järgneva tulemuse.



Pole paha! Nüüd jääb vaid üle neid defineeritud funktsioone mängu sisse lisada. Sellele on pühendatud järgmine peatükk.

# Graafikafunktsioonide integreerimine

Eelnevatest peatükkidest on meil nüüd olemas valmis tekstipõhine mäng ning funktsioonid, millega sellele graafiline liides anda.

Nüüd on vaja liita see mäng ning graafika funktsioonid. Proovime seda teha samm haaval, iga samm eraldi lehel. Jätan halliks kõik selle, mis jäi nagu varem. Värvilises tekstis on lisatud kood.

Tuletame kõigepealt meelde, milline oli meie tekstipõhine mäng.

```
from random import shuffle

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandid = [õige, vale1, vale2, vale3]
    shuffle(variandid)

    print(küsimus)
    print('Võimalikud vastused on:')
    for variant in variandid:
        print(variant)

    vastus = input('Sisesta vastus: ')
    if vastus == õige:
        return True
    else:
        return False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

punktide_arv = 0
#Käib üle kõik küsimused.
for element in küsimused:
    küsimus, v1, v2, v3, v4 = element

    vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
    if vastas_õigesti:
        print('Õige vastus!')
        punktide_arv += 1
    else:
        print('Vale vastus!')
    print('Praegune punktide arv on', punktide_arv)

print('Mäng läbi! Kogusid', punktide_arv, 'punkti.')
```

Esimene samm on lisada lihtsalt impordid, Pygame akna tekitamine ning lõpus sulgemine.

```
from random import shuffle
import pygame, sys

# Initsialiseerime Pygame ja tekitame akna.
pygame.init()
lava = pygame.display.set_mode([640,480])

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandid = [õige,vale1,vale2,vale3]
    shuffle(variandid)

    print(küsimus)
    print('Võimalikud vastused on:')
    for variant in variandid:
        print(variant)

    vastus = input('Sisesta vastus: ')
    if vastus == õige:
        return True
    else:
        return False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

punktide_arv = 0
#Käib üle kõik küsimused.
for element in küsimused:
    küsimus,v1,v2,v3,v4 = element

    vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
    if vastas_õigesti:
        print('Õige vastus!')
        punktide_arv += 1
    else:
        print('Vale vastus!')
    print('Praegune punktide arv on',punktide_arv)

print('Mäng läbi! Kogusid',punktide_arv,'punkti.')
pygame.quit()
```

Lisame õigete ja valede vastuste andmistele teadete näitamise.

```
from random import shuffle
import pygame, sys

# Initsialiseerime Pygame ja tekitame akna.
pygame.init()
lava = pygame.display.set_mode([640,480])

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandidid = [õige,vale1,vale2,vale3]
    shuffle(variandidid)

    print(küsimus)
    print('Võimalikud vastused on:')
    for variant in variandidid:
        print(variant)

    vastus = input('Sisesta vastus: ')
    if vastus == õige:
        return True
    else:
        return False

#Ekraan on siin sama tüüpi, mis muutuja lava.
def teata(punkte, tekst):
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Teadete muutujast teksts, kollase värviga.
    txt1 = font.render(tekst, 1, (200,200,200))
    txt2 = font.render(str(punkte)+' punkti!', 1, (200,200,200))
    txt3 = font.render("Vajuta ENTER, et jätkata!", 1, (200,200,200))
    #Joonistame numbritega määratud koordinaatidele teksti.
    lava.blit(txt1, (100, 100))
    lava.blit(txt2, (100, 200))
    lava.blit(txt3, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                if i.key == pygame.K_RETURN:
                    ootame = False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

punktide_arv = 0
#Käib üle kõik küsimused.
for element in küsimused:
```

```
küsimus,v1,v2,v3,v4 = element

vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
if vastas_õigesti:
    punktide_arv += 1
    teata(punktide_arv, 'Õige vastus!')
else:
    teata(punktide_arv, 'Vale vastus!')

teata(punktide_arv, 'Mäng läbi! Kogusid järgneva punktisumma:')
pygame.quit()
```



Viimasena lisame küsimuste näitamise:

Toon välja selle, et funktsioonis *küsi()* kasutatakse nüüd *input()* asemel *küsi\_graafiliselt()* funktsiooni, et saada kasutajalt valitud vastus.

Kokkuvõttes, *küsi\_graafiliselt()* saab argumentidena need 4 sõne, mis on võimalikud vastused, ning tagastab neist selle, mille kasutaja oma mupuvajutusega valib.

```
from random import shuffle
import pygame, sys

# Initsialiseerime Pygame ja tekitame akna.
pygame.init()
lava = pygame.display.set_mode([640,480])

#Ekraan on siin sama tüüpi, mis muutuja lava.
#lause on küsimus ise
#a,b,c,d on neile tähtedele vastavad vastused.
def küsi_graafiliselt(lause, a, b, c, d):
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Argumendist küsimuse tekst, hallika värviga.
    txt_kys = font.render(lause, 1, (200,200,200))
    #Argumentidest vastuste tekstid, hallika värviga.
    txt_a = font.render('a) '+a, 1, (200,200,200))
    txt_b = font.render('b) '+b, 1, (200,200,200))
    txt_c = font.render('c) '+c, 1, (200,200,200))
    txt_d = font.render('d) '+d, 1, (200,200,200))

    #Joonistame numbritega määratud koordinaatidele teksti.
    lava.blit(txt_kys, (100, 100))
    lava.blit(txt_a, (100, 150))
    lava.blit(txt_b, (100, 200))
    lava.blit(txt_c, (100, 250))
    lava.blit(txt_d, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                #Vaatame, millist kasutaja vajutas.
                #Seejärel tagastame vastava tähe vastuse.
                if i.key == pygame.K_a:
                    return a
                if i.key == pygame.K_b:
                    return b
                if i.key == pygame.K_c:
                    return c
                if i.key == pygame.K_d:
                    return d
```

```

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandid = [õige,vale1,vale2,vale3]
    shuffle(variandid)

    a,b,c,d = variandid
    vastus = küsi_graafiliselt(küsimus, a, b, c, d)
    if vastus == õige:
        return True
    else:
        return False

#Ekraan on siin sama tüüpi, mis muutuja lava.
def teata(punkte, tekst):
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Teade muutujast teksts, kollase värviga.
    txt1 = font.render(tekst, 1, (200,200,200))
    txt2 = font.render(str(punkte)+' punkti!', 1, (200,200,200))
    txt3 = font.render("Vajuta ENTER, et jätkata!", 1, (200,200,200))
    #Joonistame numbritega määratud koordinaatidele teksti.
    lava.blit(txt1, (100, 100))
    lava.blit(txt2, (100, 200))
    lava.blit(txt3, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                if i.key == pygame.K_RETURN:
                    ootame = False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

punktide_arv = 0
#Käib üle kõik küsimused.
for element in küsimused:
    küsimus,v1,v2,v3,v4 = element

    vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
    if vastas_õigesti:
        punktide_arv += 1
        teata(punktide_arv, 'Õige vastus!')
    else:
        teata(punktide_arv, 'Vale vastus!')

teata(punktide_arv, 'Mäng läbi! Kogusid järgneva punktisumma:')
pygame.quit()

```

## Faili „küsimused.txt” sisu näide

Mis on elevant?;imetaja;kala;lind;kahepaikne

Mis aastal Tartu Ülikool rajati?;1632;1991;2006;2015

Mitu maakonda on Eestis?;15;3;99;2

## Lõplik kood

```
from random import shuffle
import pygame, sys

# Initsialiseerime Pygame ja tekitame akna.
pygame.init()
lava = pygame.display.set_mode([640,480])

#Ekraan on siin sama tüüpi, mis muutuja lava.
#lause on küsimus ise
#a,b,c,d on neile tähtedele vastavad vastused.
def küsi_graafiliselt(lause, a, b, c, d):
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Argumendist küsimuse tekst, hallika värviga.
    txt_kys = font.render(lause, 1, (200,200,200))
    #Argumentidest vastuste tekstid, hallika värviga.
    txt_a = font.render('a' +a, 1, (200,200,200))
    txt_b = font.render('b' +b, 1, (200,200,200))
    txt_c = font.render('c' +c, 1, (200,200,200))
    txt_d = font.render('d' +d, 1, (200,200,200))

    #Joonistame numbritega määratud koordinaatidele teksti.
    lava.blit(txt_kys, (100, 100))
    lava.blit(txt_a, (100, 150))
    lava.blit(txt_b, (100, 200))
    lava.blit(txt_c, (100, 250))
    lava.blit(txt_d, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                #Vaatame, millist kasutaja vajutas.
                #Seejärel tagastame vastava tähe vastuse.
                if i.key == pygame.K_a:
                    return a
                if i.key == pygame.K_b:
                    return b
                if i.key == pygame.K_c:
                    return c
                if i.key == pygame.K_d:
                    return d

def küsi(küsimus, õige, vale1, vale2, vale3):
    variandid = [õige,vale1,vale2,vale3]
```

```

shuffle(variandid)

a,b,c,d = variandid
vastus = küsi_graafiliselt(küsimus, a, b, c, d)
if vastus == õige:
    return True
else:
    return False

#Ekraan on siin sama tüüpi, mis muutuja lava.
def teata(punkte, tekst):
    lava.fill([0,0,55])

    #Font arial suurusega 15
    font = pygame.font.SysFont("arial", 25)

    #Teade muutujast teks, kollase värviga.
    txt1 = font.render(tekst, 1, (200,200,200))
    txt2 = font.render(str(punkte)+' punkti!', 1, (200,200,200))
    txt3 = font.render("Vajuta ENTER, et jätkata!", 1, (200,200,200))
    #Joonistame numbritega määratud koordinaatidele teksti.
    lava.blit(txt1, (100, 100))
    lava.blit(txt2, (100, 200))
    lava.blit(txt3, (100, 300))

    pygame.display.flip()
    ootame = True
    while ootame:
        for i in pygame.event.get():
            if i.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if i.type == pygame.KEYDOWN:
                if i.key == pygame.K_RETURN:
                    ootame = False

# Loeme sisse kõik küsimused.
küsimused = []
for failiRida in open("küsimused.txt"):
    failiRida = failiRida.strip('\n')
    küsimus = failiRida.split(";")
    küsimused.append(küsimus)
shuffle(küsimused)

punktide_arv = 0
#Käib üle kõik küsimused.
for element in küsimused:
    küsimus,v1,v2,v3,v4 = element

    vastas_õigesti = küsi(küsimus, v1, v2, v3, v4)
    if vastas_õigesti:
        punktide_arv += 1
        teata(punktide_arv, 'Õige vastus!')
    else:
        teata(punktide_arv, 'Vale vastus!')

teata(punktide_arv, 'Mäng läbi! Kogusid järgneva punktisumma:')
pygame.quit()

```