Жамбыл облысы әкімдігі білім басқармасы

«Білім» кәсіби гуманитарлық-техникалық колледжі

(білім беру ұйымының атауы)

**Оқу сабағының жоспары**

(теориялық немесе өндірістік оқыту)

**Терезенің қосымша tkinter құрылымы (оқиғаларды өңдеу).**

(сабақ тақырыбы)

**Модуль/Пән атауы** Бағдарламалық кодты қайта өңдеу

**Дайындаған педагог** Нургисаева У.М

**20\_**25**\_ жылғы** «\_\_\_\_» \_\_\_сәуір\_\_\_

**1. Жалпы мәліметтер**

Курс, оқу жылы, топ 3 курс, 3БҚ-1-22, 3БҚ-2-22

Сабақ түрі: Тәжірибелік

**2. Мақсаты, міндеттері:**

**Оқу:** Білім алушыларға python (пайтон) программалау тілінде терезенің қосымша tkinter құрылымы, есеп шығаруды жүзеге асыру, қолдану жолдарын ұйымдастыру, программаны өңдеу жұмыснегіздерін үйрету, шартты операторлардың жұмыс істеу принциптерін түсіндіру.

#### **Дамыту:** Логикалық ойлау қабілеттерін дамыту, есептердің әртүрлі шешімдерін табуға дағдыландыру, программалық кодтың құрылымын дұрыс құру қабілеттерін қалыптастыру.

#### **Тәрбиелік:** Жауапкершілік пен ұқыптылыққа тәрбиелеу, алгоритмдік ойлауды дамыту, шығармашылық қабілеттерін жетілдіру.

**3. Оқу-жаттығу процесінде білім алушылар меңгеретін күтілетін нәтижелер және кәсіби дағдылар тізбесі:** Python программалау тілінде терезенің қосымша tkinter құрылымына есеп шығаруды жүзеге асыру, қолдану жолдарын ұйымдастыру, программаны өңдеу және пайдалану дағдыларын меңгеру

**4. Қажетті ресурстар:** ДК немесе ноутбуктер

[**https://skillbox.ru/media/code/pishem-desktopprilozhenie-na-python-s-pomoshchyu-tkinter/**](https://skillbox.ru/media/code/pishem-desktopprilozhenie-na-python-s-pomoshchyu-tkinter/)

***5. Сабақтың барысы: (90 минут)***

**5.1. Ұйымдастыру кезеңі:** *( 3 мин )*

**5.2. Үй жұмысын жан-жақты тексеру:**

***“Домино ” әдісі*** *(15 минут)*

**Терезенің қосымша tkinter құрылымы (оқиғаларды өңдеу).**

Tkinter – бұл Python тілінде графикалық интерфейсті (GUI) жасауға арналған стандартты кітапхана. Ол Windows, macOS және Linux операциялық жүйелерінде жұмыс істейді және Python бағдарламаларына ыңғайлы қолданушы интерфейсін (UI) құруға мүмкіндік береді. Tkinter кітапханасының көмегімен жасалған интерфейстер интерактивті болады, яғни пайдаланушы әрекеттеріне жауап беретін оқиғаларды өңдеу механизмдерін қамтиды.

GUI-дің маңызды аспектілерінің бірі – **оқиғаларды өңдеу механизмі**. Tkinter-дің негізгі ерекшелігі – пайдаланушы әрекеттерін (мысалы, батырманы басу, мәтін енгізу, тышқанды жылжыту, пернетақта пернелерін басу) өңдеу мүмкіндігі. Бұл процестер оқиғаларды өңдеудің қосымша құрылымдарымен жүзеге асады.

**1. Tkinter терезесінің негізгі құрылымы**

Tkinter интерфейстерін жасау үшін негізгі терезе (main window) құрылуы керек. Терезе – барлық интерфейс элементтерін қамтитын негізгі контейнер. Tkinter-де терезе құру үшін Tk() класы қолданылады:

import tkinter as tk

# Негізгі терезені құру

root = tk.Tk()

# Терезені көрсету

root.mainloop()

Бұл код root атты негізгі терезені жасайды және оны іске қосу үшін mainloop() әдісін шақырады.

**2. Оқиғаларды өңдеу (Event Handling) механизмдері**

Оқиғаларды өңдеу – GUI қосымшаларының негізгі функцияларының бірі. Tkinter-де оқиғаларды өңдеу механизмдері **bind() әдісі**, **command параметрі**, **after() және after\_cancel() әдістері** арқылы жүзеге асады.

**2.1. bind() әдісі арқылы оқиғаны өңдеу**

bind() әдісі қолданушы әрекеттеріне жауап беру үшін қолданылады. Бұл әдіс белгілі бір оқиға орын алған кезде функция шақыруға мүмкіндік береді.

**Мысал: Пернетақта батырмасын басқанда оқиға өңдеу**

import tkinter as tk

def key\_pressed(event):

print(f"Басылған перне: {event.char}")

root = tk.Tk()

root.bind("<Key>", key\_pressed) # Пернетақта оқиғасын өңдеу

root.mainloop()

Бұл бағдарламада кез келген пернетақта батырмасы басылған кезде, оқиға өңделіп, терминалда қандай батырма басылғаны көрсетіледі.

**Кейбір кең таралған оқиғалар:**

* "<Button-1>" – тышқанның сол жақ батырмасын басу
* "<Button-3>" – тышқанның оң жақ батырмасын басу
* "<KeyPress>" – кез келген пернетақта батырмасын басу
* "<Motion>" – тышқанды жылжыту

**2.2. command параметрі арқылы оқиғаны өңдеу**

command параметрі көбінесе батырма сияқты виджеттермен қолданылады.

**Мысал: Батырманы басу арқылы оқиға өңдеу**

import tkinter as tk

def button\_click():

print("Батырма басылды!")

root = tk.Tk()

button = tk.Button(root, text="Басыңыз", command=button\_click)

button.pack()

root.mainloop()

Бұл бағдарламада батырма басылғанда button\_click() функциясы орындалып, терминалда "Батырма басылды!" деген хабарлама шығады.

**2.3. after() және after\_cancel() әдістері**

after() әдісі белгілі бір уақыт өткеннен кейін белгілі бір функцияны шақыру үшін қолданылады.

**Мысал: Белгілі бір уақыттан кейін хабарлама шығару**

import tkinter as tk

def show\_message():

label.config(text="5 секунд өтті!")

root = tk.Tk()

label = tk.Label(root, text="Күтіңіз...")

label.pack()

root.after(5000, show\_message) # 5000 миллисекунд (5 секунд) кейін функция орындалады

root.mainloop()

Бұл бағдарлама іске қосылған кезде, алдымен "Күтіңіз..." деген мәтін шығады, ал 5 секундтан кейін ол "5 секунд өтті!" деп өзгереді.

**3. Тышқан оқиғаларын өңдеу**

Тышқанмен орындалатын әрекеттерді өңдеу үшін bind() әдісі жиі қолданылады.

**Мысал: Тышқанның сол жақ батырмасын басқанда хабарлама шығару**

import tkinter as tk

def mouse\_click(event):

print(f"Тышқан басылды: {event.x}, {event.y}")

root = tk.Tk()

root.bind("<Button-1>", mouse\_click) # Сол жақ батырманы басу

root.mainloop()

Бұл бағдарламада тышқанның сол жақ батырмасын басқан кезде координаттар терминалда көрсетіледі.

**4. Көп деңгейлі оқиғаларды өңдеу**

Кейде бір оқиға бірнеше элементтерде қатар өңделуі мүмкін. Tkinter-де **пропагандация (event propagation)** арқылы оқиғалар жоғары деңгейлі элементтерге берілуі мүмкін.

**Мысал: Батырмадағы оқиғаны негізгі терезеге беру**

import tkinter as tk

def button\_click(event):

print("Батырма басылды!")

event.widget.master.event\_generate("<KeyPress-a>") # Басқа оқиғаны шақыру

def key\_event(event):

print("A пернесі басылды!")

root = tk.Tk()

root.bind("<KeyPress-a>", key\_event)

button = tk.Button(root, text="Басыңыз")

button.pack()

button.bind("<Button-1>", button\_click)

root.mainloop()

Бұл кодта батырманы басқан кезде "Батырма басылды!" хабарламасы шығады және сонымен бірге A батырмасын басқандай эффект пайда болады.

**Тапсырма**

**Дене салмағының индексін есептеу үшін калькуляторды жасаңыз**

Біз дене салмағының индексінің калькуляторын жазамыз. BMI — адамның артық салмағы немесе семіздігін бағалауға мүмкіндік беретін маңызды медициналық көрсеткіш. Ол келесі формула бойынша есептеледі: ​​

**BMI = салмақ (кг-мен) / биіктік2 (метрмен)**

Есептеу нәтижелері арнайы кесте арқылы бағаланады. Дәрігерлерде оның көптеген градациялары бар, бірақ біз жеңілдетілген нұсқаны қолданамыз:



**1-қадам**

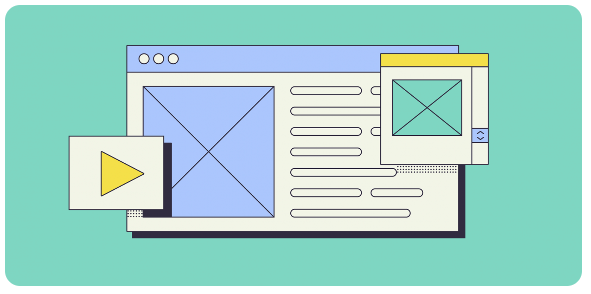
**Python бағдарламасын іске қосыңыз және Tkinter импорттаңыз**

Python-да кодты арнайы IDE-де жазған дұрыс, мысалы, PyCharm немесе [Visual Studio Code](https://code.visualstudio.com/). Олар синтаксисті бөлектейді және кодтың жалғасын ұсынады — бұл бағдарламашының жұмысын айтарлықтай жеңілдетеді. Біз осы мақаладағы барлық кодты Visual Studio Code бағдарламасында жаздық.

Tkinter кітапханасы Python тілінде алдын ала орнатылған. Сондықтан оны тек импорттау керек:

import tkinter as tk

Енді біз осы кітапханадан кез келген модульдерді пайдалана аламыз.



**2-қадам**

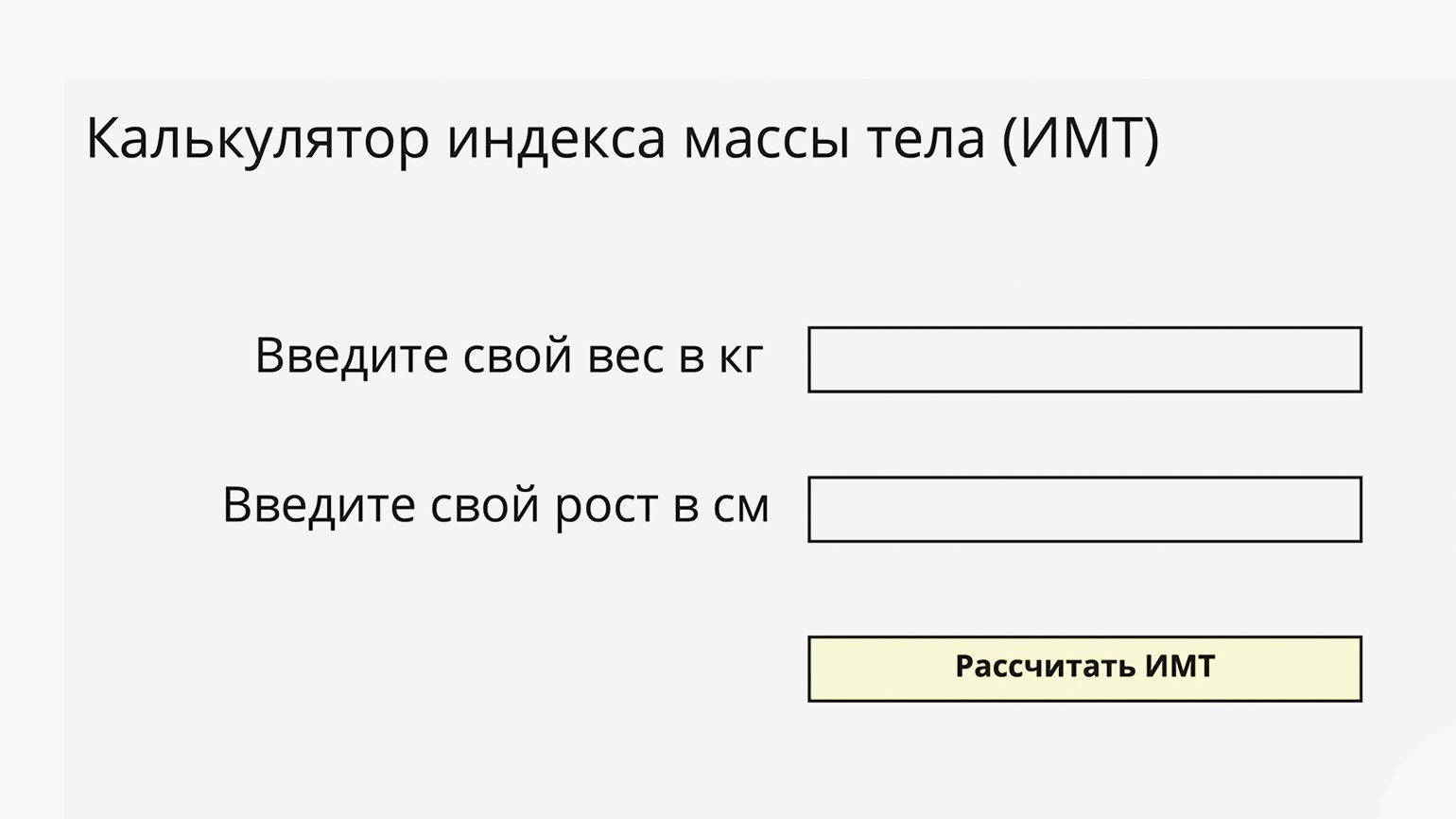
**Интерфейсті жасаймыз және қолданбаның логикасын ойластырамыз**

Кодты жазбас бұрын бірнеше сұрақтарға жауап беру керек:

* Пайдаланушыдан қандай деректерді және қандай пішінде алғымыз келеді?
* BMI есептеуін қандай оқиға іске қосады: түймені басу, қолданбаның барлық қажетті деректерді алуы немесе басқа нәрсе?
* Нәтижені қалай көрсетеміз?

Біздің жағдайда пайдаланушыдан салмақ пен бойды бүтін сандар түрінде алу керек. Бұл жағдайда салмақты килограмммен, ал биіктігін сантиметрмен енгізу керек. BMI түймені басу арқылы есептеледі және нәтиже қалқымалы терезеде BMI мәні және ол тиесілі санат ретінде көрсетіледі.

Схемалық түрде калькуляторымыздың графикалық интерфейсі келесідей болады:



**3-қадам**

**Негізгі терезені жасаймыз және қолданбаның атын көрсетеміз**

Кітапхананы Python-ға импорттағаннан кейін оның әдістерін жүктеп аламыз:

from tkinter import \*

from tkinter import messagebox

Бірінші жол барлық Tkinter әдістерін жүктеп алуға және олардың атына сілтеме жасамай-ақ кодта пайдалануға мүмкіндік береді. Екінші жолмен біз әдісті анық импорттаймыз messagebox, оны нәтижесі бар қалқымалы терезені көрсету үшін пайдаланамыз. Бұл ыңғайлы, өйткені әдіс бізге бірнеше рет қажет болады.

Енді қолданбамыздың терезесін жасайық. Ол үшін модульді қолданамыз Tk. Қолданбаны "Дене салмағының индексі (BMI) калькуляторы" деп атаймыз.:

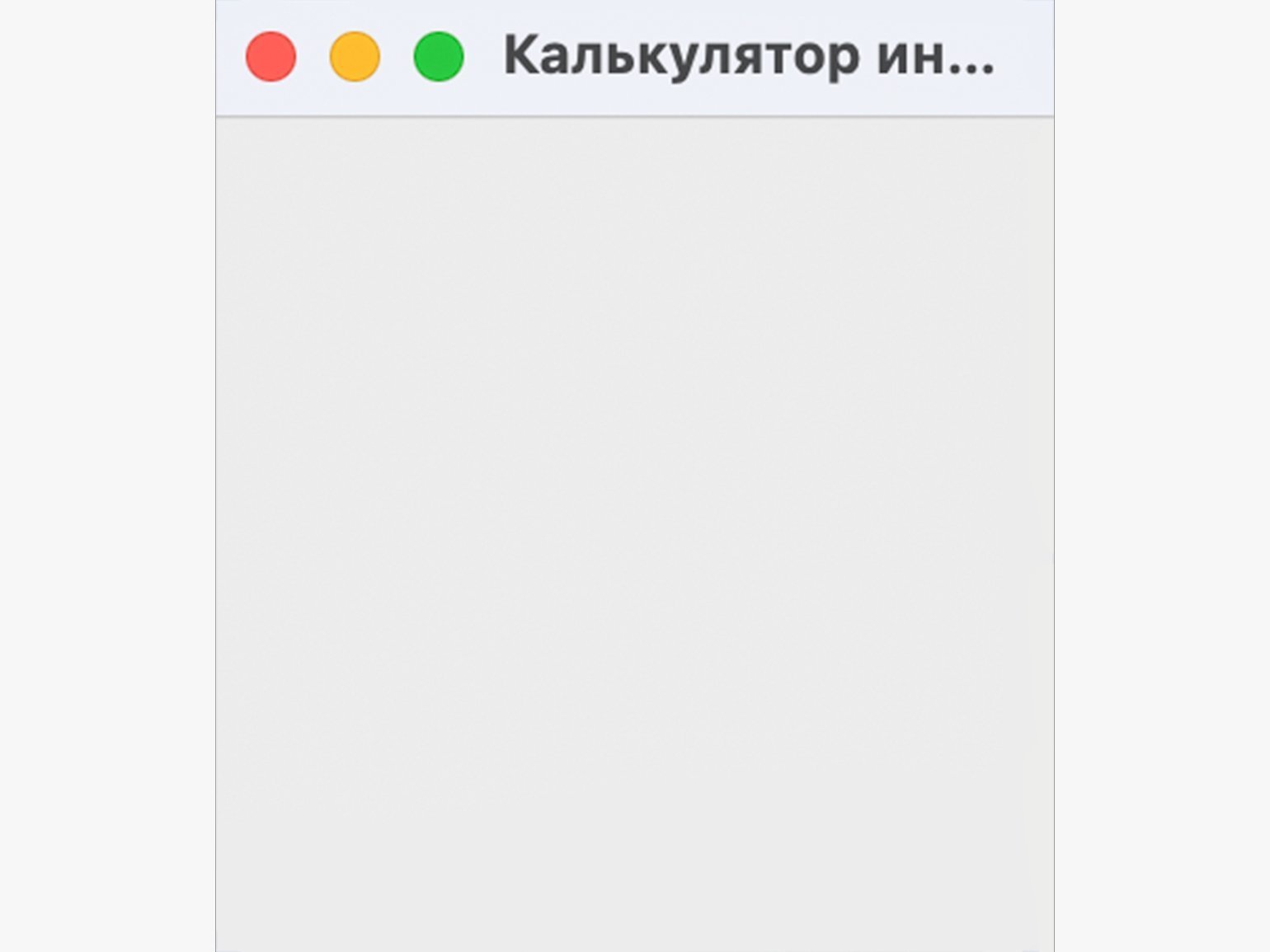
window = Tk() #Создаём окно приложения.

window.title("Калькулятор индекса массы тела (ИМТ)") #Добавляем название приложения.

Кодты іске қосқаннан кейін ештеңе болмайды. Бұл қате емес. Шындығында, код орындалды және терезе жабылды. Пайдаланушы мұны өзі жасамайынша, қолданба терезесі жабылмауы керек екенін нақты көрсету керек. Ол үшін кодқа функция қосамыз window.mainloop () оқиғалар циклінің басталуын көрсетеді:

window.mainloop()

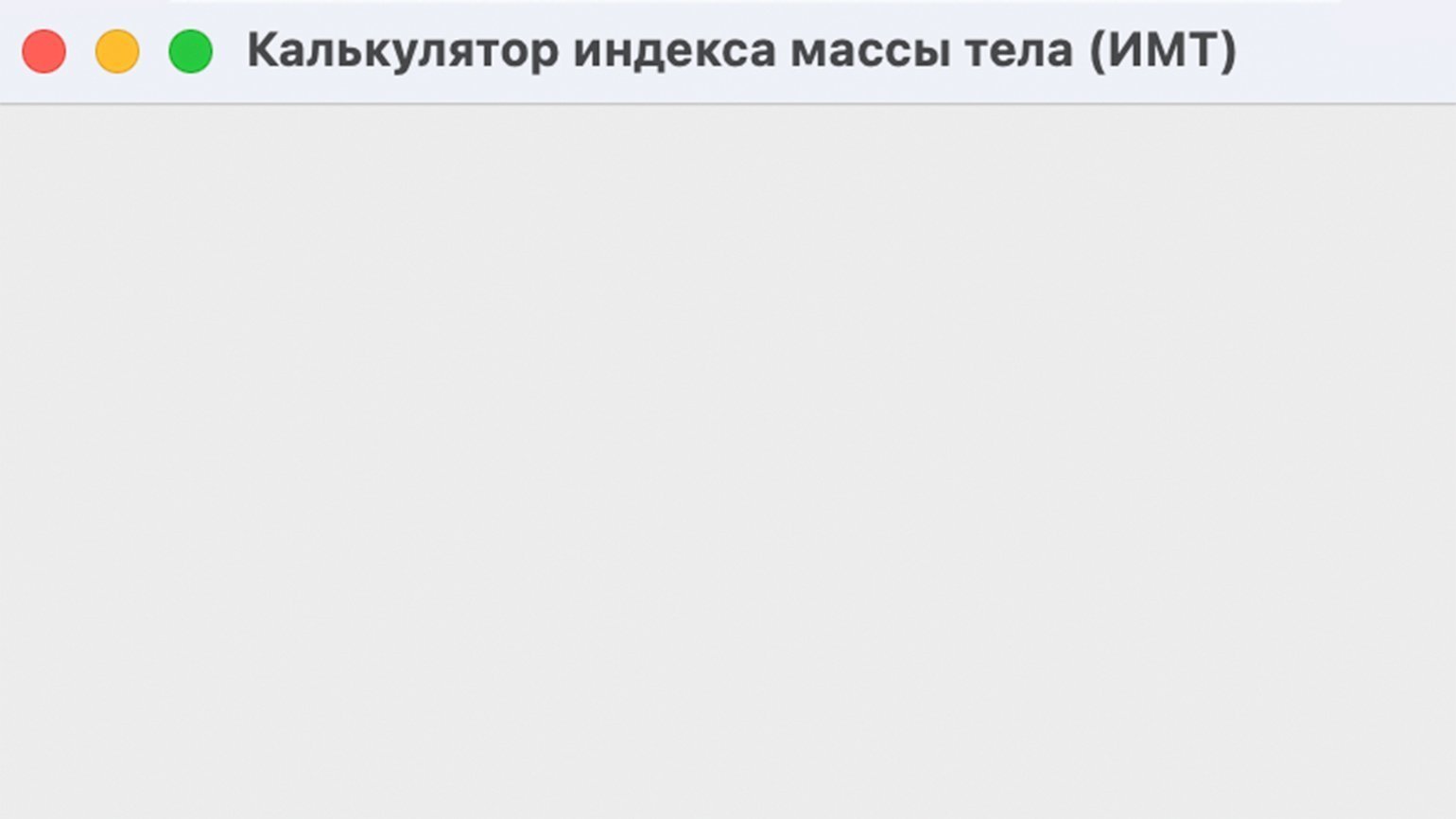
Кодты іске қосқаннан кейін қолданба экранын көреміз:



Біз терезенің өлшемін көрсетпедік, сондықтан қолданбаның атауы оған толық сәйкес келмейді. Мұны әдіс арқылы түзетейік geometry:

window.geometry('400x300')

Енді қолданбаның атауы толығымен көрінеді:



**4-қадам**

**Элементтердің орналасуын басқару үшін жақтау виджетін жасаңыз**

Қолданба терезесінде нобайымыздан бірнеше элементтерді орналастыру керек: субтитрлері бар екі ақпаратты енгізу өрісі және бір түйме. Шеттердің бір-біріне сәйкес келмеуі және терезенің сыртына шығып кетпеуі маңызды. Tkinter-де бұл үшін бірнеше әдістер бар:

* pack — элементтерге арналған контейнерлермен жұмыс істегенде пайдаланылады. Түймелерді, белгілерді немесе басқа элементтерді контейнерлер ішінде орналастыруға мүмкіндік береді.
* place — нақты координаттарды көрсете отырып, элементтерді орналастыруға мүмкіндік береді.
* grid — элементтерді қолданба терезесін бөлетін шартты тордың ұяшықтары бойынша орналастырады.

Біз әдістердің комбинациясын қолданамыз pack және grid. Бастау үшін жазуларды, енгізу өрістерін және түймелерді орналастыру үшін жақтау виджетін жасайық. Виджет жұмысының толық сипаттамасы берілген [құжаттама](https://www.tutorialspoint.com/python/tk_frame.htm" \t "_blank). Біз тек екі сипатты пайдаланамыз: padx және pady.

Frame ішінде орналасатын элементтер үшін тігінен және көлденеңінен 10 пиксельдік шегіністерді белгілейік:

frame = Frame(

window, #Обязательный параметр, который указывает окно для размещения Frame.

padx = 10, #Задаём отступ по горизонтали.

pady = 10 #Задаём отступ по вертикали.

)

frame.pack(expand=True) #Не забываем позиционировать виджет в окне. Здесь используется метод pack. С помощью свойства expand=True указываем, что Frame заполняет весь контейнер, созданный для него.

**5-қадам**

**Ақпаратты енгізу өрістерін және дене салмағының индексін есептеуді бастау түймешігін қосыңыз**

Қолданба терезесінде виджеттердің үш түрін қосуымыз керек: ақпаратты енгізуге арналған өріс (Entry), мәтіндік белгілер (Label) және түйме (Button).

Жазбалардан бастайық. Жапсырма виджетін пайдаланайық:

height\_lb = Label(

frame,

text="Введите свой рост (в см) "

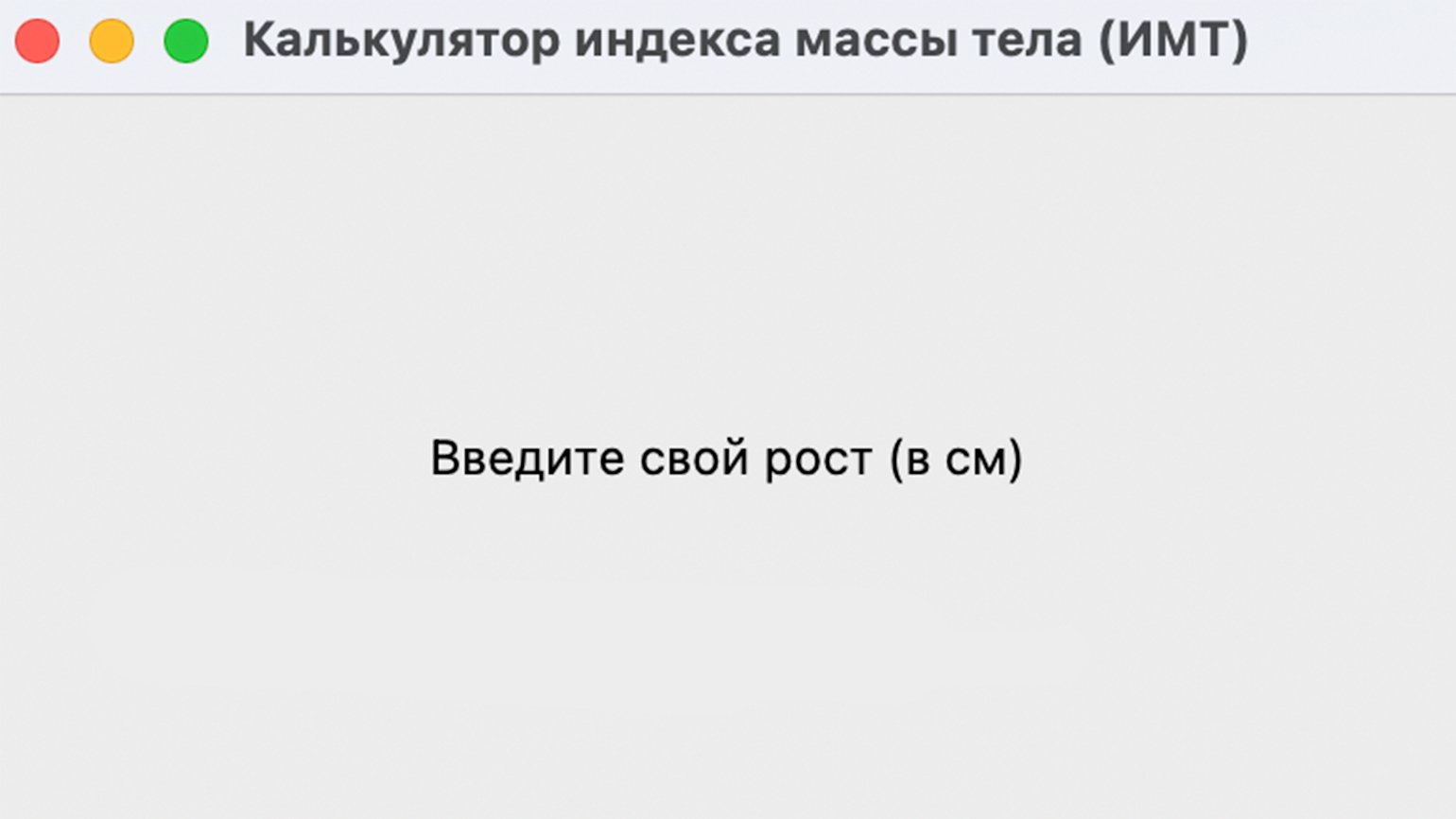
)

height\_lb.grid(row=3, column=1)

Label виджетіне екі параметрді береміз:

* frame — тік және көлденең толтыру әлдеқашан конфигурацияланған жақтау виджетінің дайындамасын пайдаланамыз.
* text — экранда көрсетілетін мәтін.

Виджетті орналастыру үшін әдісті қолданамыз grid. Мәтін координаттары бар ұяшықта орналасуы керек екенін көрсетеміз "3-ші жол, 1-ші баған". Егер кодты іске қоссақ, онда бір элементті көреміз:



Элемент қазір терезенің ортасында орналасқан, бірақ басқа элементтерді жазған кезде ол дұрыс орынға орналасады.

Салмақ туралы екінші жазбаны дәл осылай қосамыз, бірақ орналасу кезінде grid келесі, төртінші жолды көрсетейік:

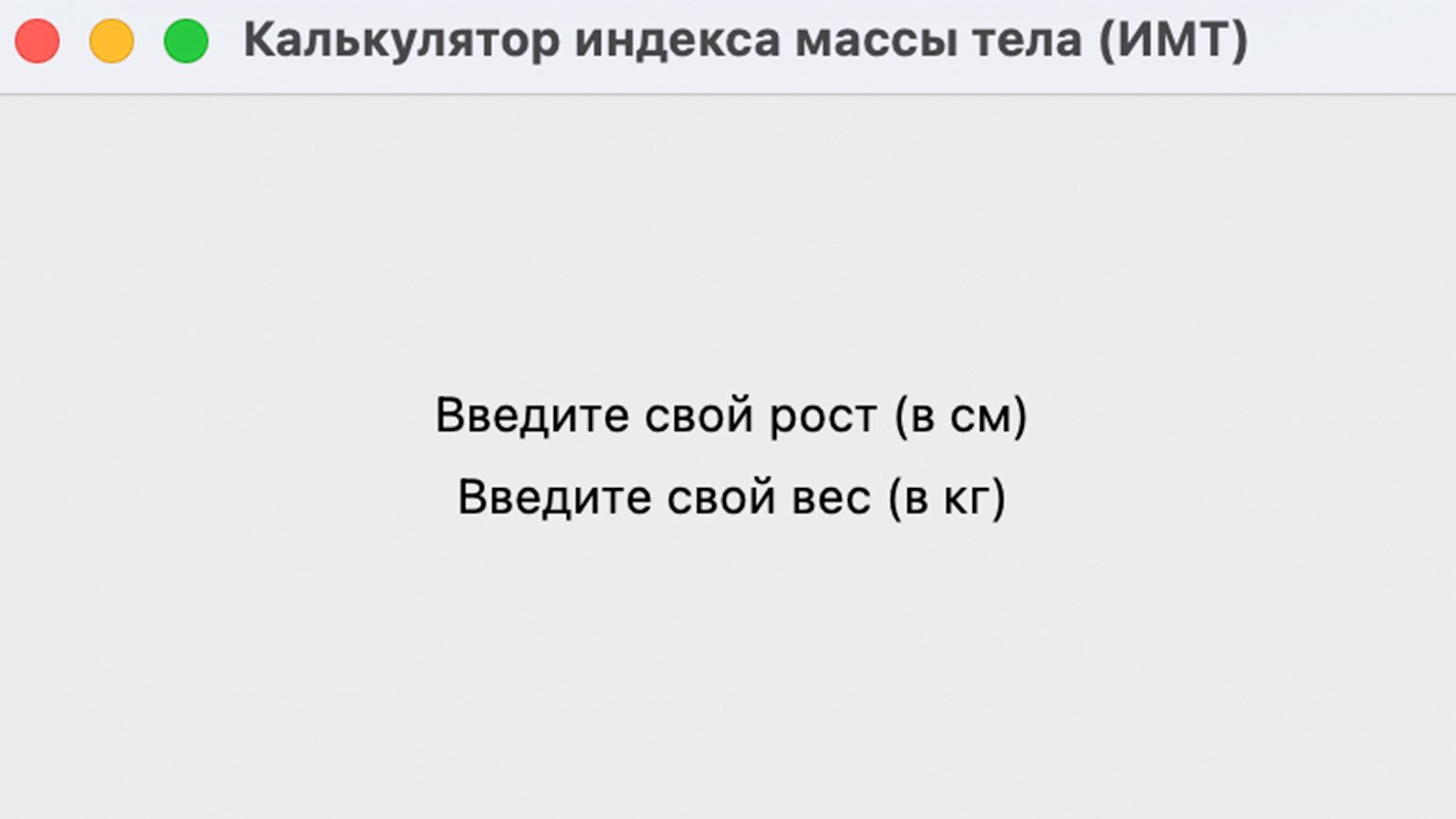
weight\_lb = Label(

frame,

text="Введите свой вес (в кг) ", )

weight\_lb.grid(row=4, column=1)

Кодты іске қосамыз және нәтижеге қараймыз:



Енді Entry виджетін пайдаланып пайдаланушы ақпаратын енгізу үшін өрістерді қосайық:

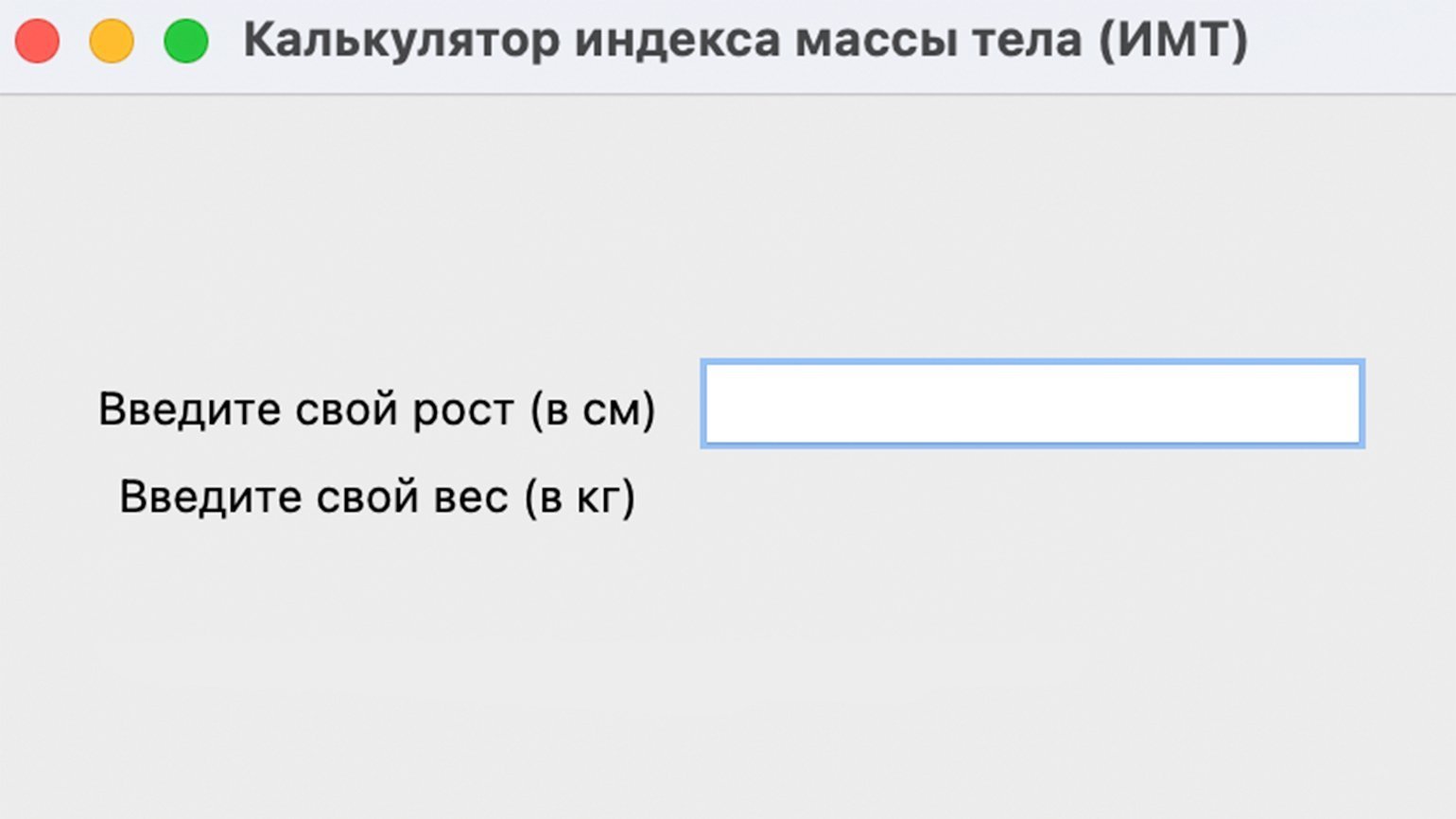
height\_tf = Entry(

frame, #Используем нашу заготовку с настроенными отступами.

)

height\_tf.grid(row=3, column=2)

Орналастыру үшін біз әдісті де қолдандық grid. Біздің элемент "Бойыңызды (см-мен) енгізіңіз" дегенге қарама-қарсы орналасуы керек екенін ескеріңіз. Сондықтан біз ұяшықты бір жолда, бірақ екінші бағанда пайдаланамыз. Кодты іске қосайық және нәтижені көрейік:



Барлығы ойдағыдай болды. Аналогия бойынша салмақ енгізу өрісін қосу қалады:

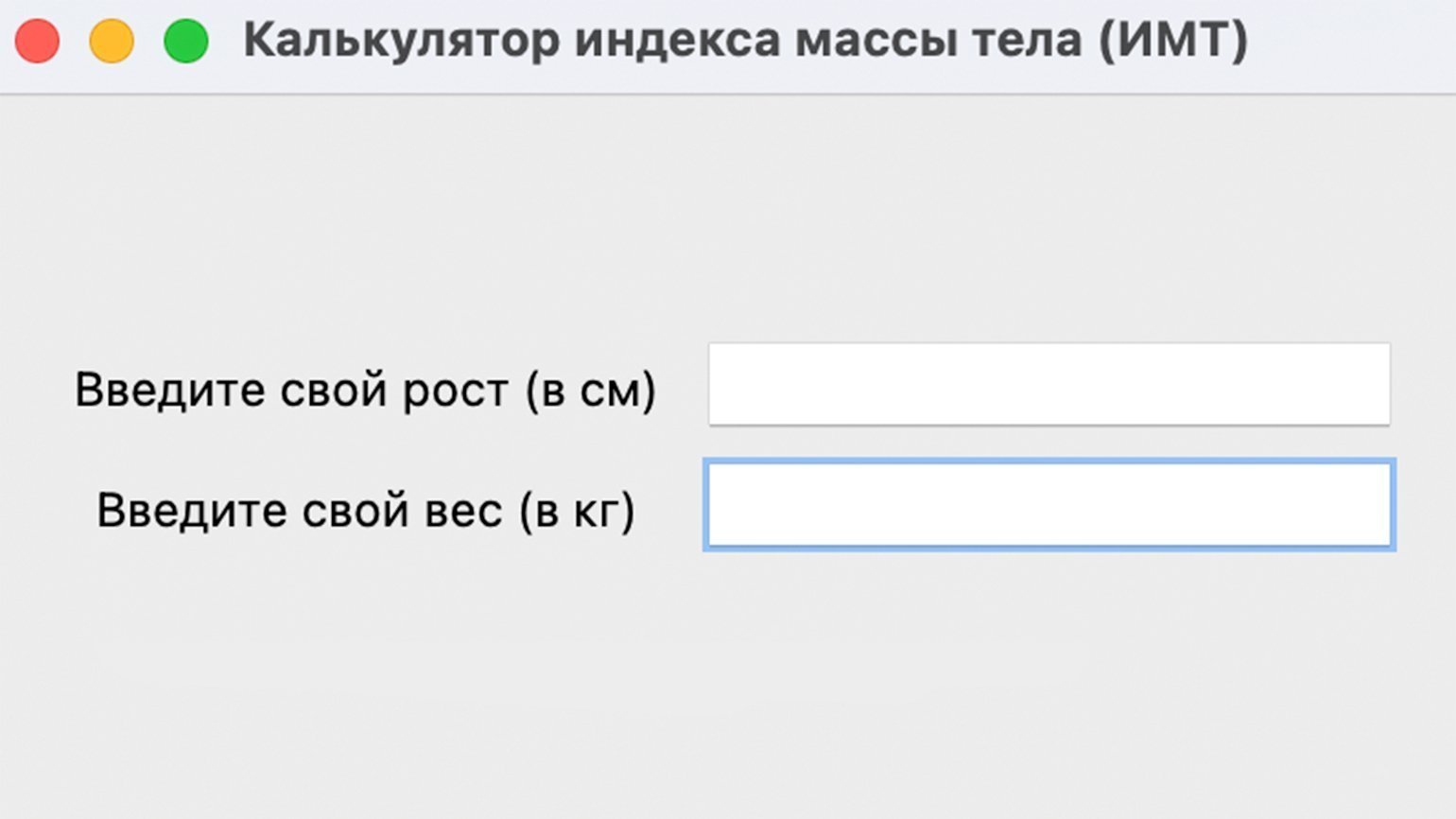
weight\_tf = Entry(

frame,

)

weight\_tf.grid(row=4, column=2, pady=5)

Нәтижесін көрейік:



Енді BMI есептеуді бастайтын түймені қосайық. Мұны Button виджетінің көмегімен орындаймыз:

cal\_btn = Button(

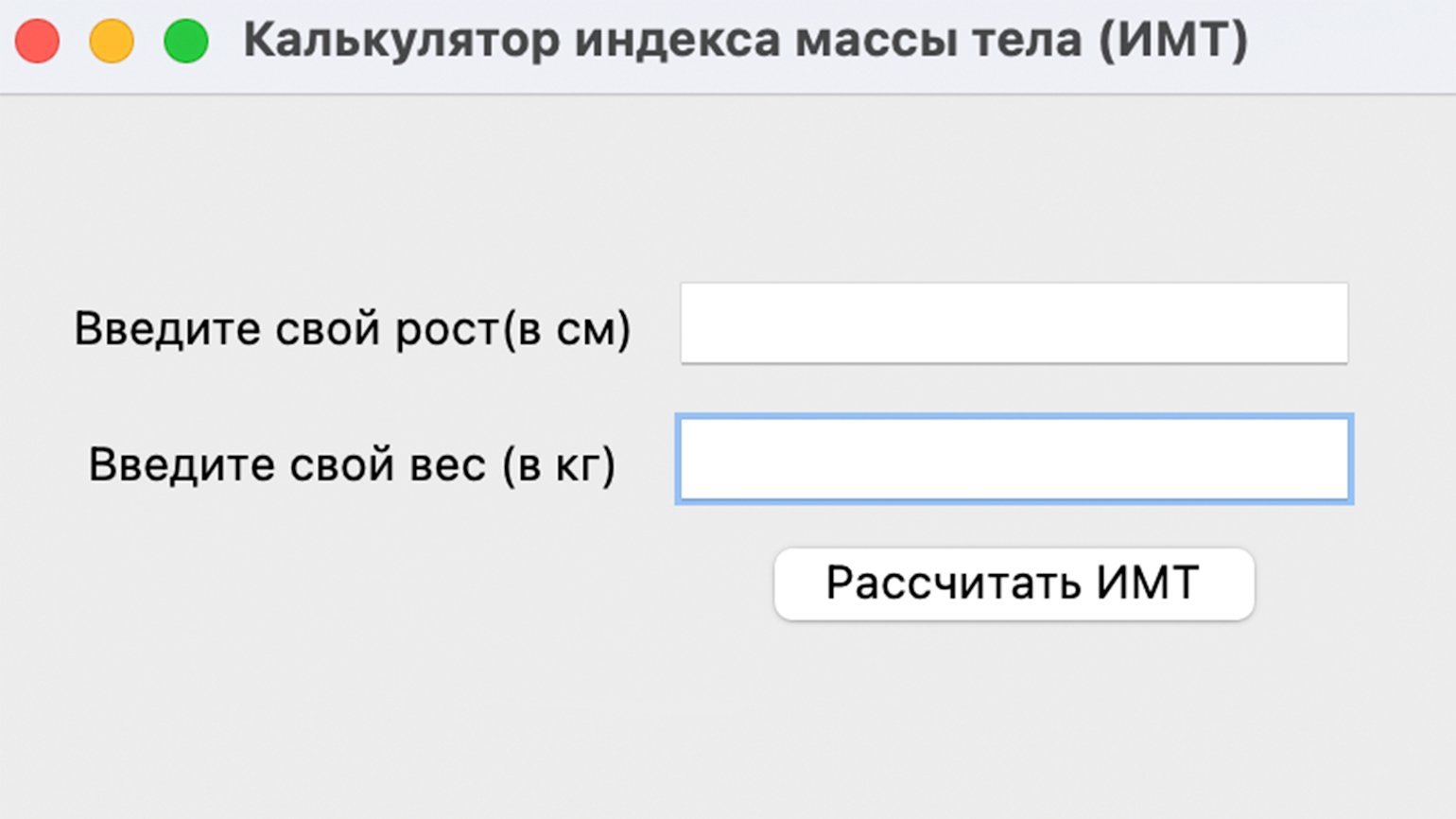
frame, #Заготовка с настроенными отступами.

text='Рассчитать ИМТ', #Надпись на кнопке.

)

cal\_btn.grid(row=5, column=2) #Размещаем кнопку в ячейке, расположенной ниже, чем наши надписи, но во втором столбце, то есть под ячейками для ввода информации.

Нәтижесін көрейік:



Енді қолданбада барлық графикалық элементтер бар. Entry виджеттерінен ақпаратты алатын және дене салмағының индексін есептейтін кодты жазу ғана қалады.

**6-қадам**

**Entry виджеттерінен ақпарат аламыз және дене салмағының индексін есептейміз**

Қарапайым функцияны жазып, оны жол бойынша талдап көрейік:

def calculate\_bmi(): #Объявляем функцию.

kg = int(weight\_tf.get()) #С помощью метода .get получаем из поля ввода с именем weight\_tf значение веса, которое ввёл пользователь и конвертируем в целое число с помощью int().

m = int(height\_tf.get())/100 #С помощью метода .get получаем из поля ввода с именем height\_tf значение роста и конвертируем в целое число с помощью int(). Обязательно делим его на 100, так как пользователь вводит рост в сантиметрах, а в формуле для расчёта ИМТ используются метры.

bmi = kg/(m\*m)#Рассчитываем значение индекса массы тела.

bmi = round(bmi, 1) #Округляем результат до одного знака после запятой.

Функция дайын. Бірақ енді есептеу нәтижесін бағалап, пайдаланушыға хабарлама көрсетуіміз керек.

**7-қадам**

**Нәтижені бағалаймыз және пайдаланушыға көрсетеміз**

Функциямызды толықтырайық calculate\_bmi. Шартты операторды қолданайық if алынған BMI мәндерін ескеру үшін және Tkinter әдісін қолданыңыз messagebox қалқымалы терезеде хабарды көрсету үшін:

if bmi < 18.5:

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует недостаточному весу')

elif (bmi > 18.5) and (bmi < 24.9):

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует нормальному весу')

elif (bmi > 24.9) and (bmi < 29.9):

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует избыточному весу')

else:

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует ожирению')

Соңғы қадам қалады — біздің функция "BMI есептеу" түймесін басқан кезде іске қосылуы керек. Ол үшін сипатты қосамыз command виджет түймесінде:

cal\_btn = Button(

frame,

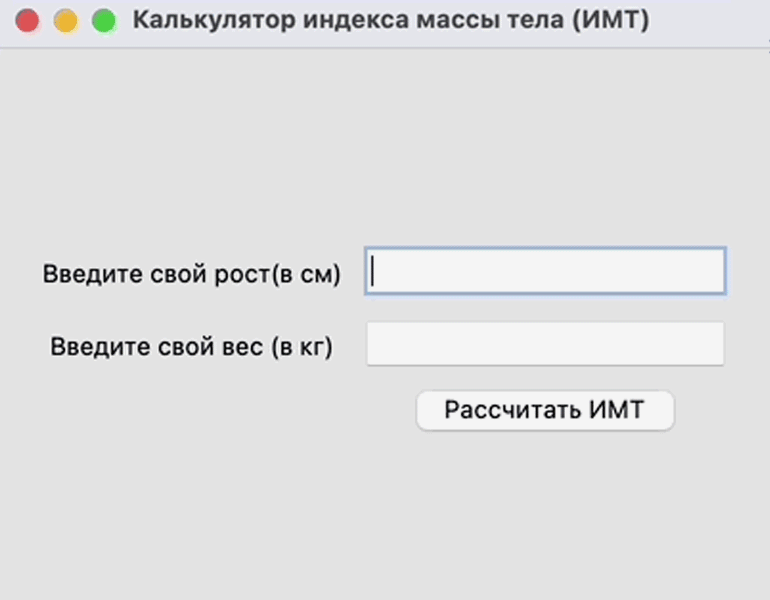
text='Рассчитать ИМТ',

command=calculate\_bmi #Позволяет запустить событие с функцией при нажатии на кнопку.

)

cal\_btn.grid(row=5, column=2)

Кодты іске қосайық және нәтижені көрейік:



Барлығы жұмыс істейді. Функция деректерді енгізу өрістерінен алады және нәтижені экранда көрсете отырып, дене салмағының индексін есептейді.

**Кодты толығымен келтірейік пікірлер жоқ**

from tkinter import \*

from tkinter import messagebox

def calculate\_bmi():

kg = int(weight\_tf.get())

m = int(height\_tf.get())/100

bmi = kg/(m\*m)

bmi = round(bmi, 1)

if bmi < 18.5:

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует недостаточному весу')

elif (bmi > 18.5) and (bmi < 24.9):

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует нормальному весу')

elif (bmi > 24.9) and (bmi < 29.9):

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует избыточному весу')

else:

messagebox.showinfo('bmi-pythonguides', f'ИМТ = {bmi} соответствует ожирению')

window = Tk()

window.title('Калькулятор индекса массы тела (ИМТ)')

window.geometry('400x300')

frame = Frame(

window,

padx=10,

pady=10 )

frame.pack(expand=True)

height\_lb = Label(

frame,

text="Введите свой рост (в см) " )

height\_lb.grid(row=3, column=1)

weight\_lb = Label(

frame,

text="Введите свой вес (в кг) ", )

weight\_lb.grid(row=4, column=1)

height\_tf = Entry(

frame, )

height\_tf.grid(row=3, column=2, pady=5)

weight\_tf = Entry(

frame, )

weight\_tf.grid(row=4, column=2, pady=5)

cal\_btn = Button(

frame,

text='Рассчитать ИМТ',

command=calculate\_bmi)

cal\_btn.grid(row=5, column=2)

window.mainloop()

|  |  |  |
| --- | --- | --- |
| **Білдім** | **Білемін** | **Білгім келеді** |
|  |  |  |

**Рефлексия**

**Үй тапсырмасы –** Тапсырманы орындау және қорғау