Қазақстан Республикасының білім және ғылым министірлігі

Ш. Уәлиханов атындағы Көкшетау мемлекеттік университеті

Министерство образования и науки Республики Казахстан

Кокшетауский государственный университет им. Ш. Уалиханова

**ДИПЛОМДЫҚ ЖҰМЫС**

**ДИПЛОМНАЯ РАБОТА**

Тақырыбы:

На тему:

Мамандығы

Специальность

Орындады:

Выполнил**: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(қолы/подпись) (аты-жөні / ФИО)

#### Жетекші

Руководитель **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(қолы/подпись) (аты-жөні / ФИО)

#### «Қорғауға жіберілді»

#### «Допускается к защите»

Кафедра меңгерушісі

Зав. Кафедрой  **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

(қолы/подпись) (аты-жөні / ФИО)

Көкшетау 20\_\_\_

Мазмұны

[БЕЛГІЛЕР МЕН ҚЫСҚАРТУЛАР ТІЗІМІ 2](#_Toc198246532)

[КІРІСПЕ 3](#_Toc198246533)

[1. НЕГІЗГІ БӨЛІМ 4](#_Toc198246534)

[1.1 Тапсырма қою. 4](#_Toc198246535)

[1.2 Құралдарды таңдау 5](#_Toc198246536)

[1.3 Оқулық құрылымының сипаттамасы 5](#_Toc198246537)

[1.4 Оқулықты кезең-кезеңмен құру 6](#_Toc198246538)

[1.5 Файл құрылымын құру 8](#_Toc198246539)

[1.5.1 Дәрістерді әзірлеу 9](#_Toc198246540)

[1.5.2 Тесттер құру 39](#_Toc198246541)

[1.5.3 Практика 44](#_Toc198246542)

[1.6 JavaScript көмегімен интерактивтілікті қосу 49](#_Toc198246543)

[1.7 Тестілеу және жөндеу 57](#_Toc198246544)

[2 ЭКСПЕРИМЕНТТІК БӨЛІМ 59](#_Toc198246545)

[2.1 Эксперименттің мақсаты мен міндеттері 59](#_Toc198246546)

[2.2 Эксперименттік топтар 59](#_Toc198246547)

[2.3 Эксперимент өткізу барысы 59](#_Toc198246548)

[2.4 Сауалнама және кері байланыс 60](#_Toc198246549)

[2.5 Нәтижелерді өңдеу және есептеу формуласы 60](#_Toc198246550)

[2.6 Нәтижелерді талдау 61](#_Toc198246551)

[ҚОРЫТЫНДЫ 62](#_Toc198246552)

[ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ 63](#_Toc198246553)

# БЕЛГІЛЕР МЕН ҚЫСҚАРТУЛАР ТІЗІМІ

Дипломдық жобаға тиісті белгілері мен қысқартулары бар бірқатар терминдер қолданылады:

|  |  |  |
| --- | --- | --- |
| HTML | - | Hypertext Markup Language (гипермәтінді белгілеу тілі). |
| CSS | - | Cascading Style Sheets (каскадты стиль кестелері). |
| API | - | Application Programming Interface (қолданбалы бағдарламалау интерфейсі). |
| IDE | - | интеграцияланған даму ортасы (интеграцияланған даму ортасы). |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# КІРІСПЕ

Қазіргі уақытта бағдарламалау және дамыту салаларының негізгі бағыттарының бірі – қолжетімді және тиімді оқу ресурстарын әзірлеу. Python бағдарламалау тілі өзінің қарапайымдылығымен, әмбебаптығымен және кең экожүйесімен ең танымал және сұранысқа ие тілдердің бірі ретінде танылды. Ол білім беру мекемелерінде, ғылыми зерттеулерде және күрделі бағдарламалық шешімдер әзірлеуде белсенді түрде қолданылады.

Дипломдық жоба аясында жоғары оқу орындарының студенттеріне Python бағдарламалау тілін нөлден үйренуге көмектесетін электрондық оқулық әзірлеу жоспарланды. Оқулық дәріс материалдарын, практикалық тапсырмаларды және өзін-өзі тексеруге арналған тестілерді қамтиды, осылайша оқытудың толық циклін қамтамасыз етеді.

Жобаның мақсаты – теория мен практиканы біріктіретін, оқушыларға өз бетінше оқуға және аудиторияда жұмыс істеуге қолайлы, құрылымдалған оқу ресурсын құру.

Жобаның міндеттері:

* Электрондық оқу материалдарын жасау тәсілдерін зерттеу.
* Дәрістер, тесттер және практикалық тапсырмаларды қамтитын оқулық құрылымын әзірлеу.
* Әзірлеу құралдарын таңдау (HTML, CSS, JavaScript, Visual Studio Code).
* Таңдалған құралдарды пайдаланып дәрістер, тесттер және тапсырмаларды әзірлеу.
* Оқулықтың қарапайымдылығы үшін оның интерактивтілігін қамтамасыз ету.
* Оқулықтың функционалдығын тексеру үшін оны тестілеу және жөндеу.

Жобаның өзектілігі. Білім беру процесінің цифрлық трансформациясы жағдайында электрондық оқу материалдары маңызды рөл атқарады. Python тілінің деректер ғылымы, жасанды интеллект және веб-әзірлеу сияқты салаларда қолданылуы оны техникалық мамандықтар студенттері арасында ерекше сұранысқа ие етеді. Осы орайда Python тілі бойынша сапалы оқу ресурстарын әзірлеу оқу процесін жақсартуға және студенттердің бағдарламалауға деген қызығушылығын арттыруға мүмкіндік береді.

# НЕГІЗГІ БӨЛІМ

# 1.1 Тапсырма қою.

Қазіргі уақытта білім беру процесі цифрландырылуда, бұл өз кезегінде оқу материалдарын тиімді әрі интерактивті түрде ұсынуға мүмкіндік береді. Python бағдарламалау тілі кеңінен қолданысқа ие болуына байланысты, оның негізгі түсініктерін студенттерге жеңіл әрі қызықты түрде жеткізу өте маңызды. Python тілінің қарапайымдылығы мен көпфункционалдығы оны бағдарламалау тілін жаңа бастап жатқан студенттер үшін ыңғайлы етеді. Бұл жоба осы мақсатты көздеп, студенттерге Python тілін үйретуге арналған толыққанды электронды оқулықты құруды мақсат етеді.

Жобаның негізгі мақсаты - Python тілін үйренушілер үшін толыққанды электронды оқулық құру. Бұл оқулық студенттерге тек теориялық білім ғана емес, сонымен қатар практикалық тапсырмаларды орындау арқылы код жазу дағдыларын дамытуға мүмкіндік береді. Оқулық 10 дәріс материалын, тест бөлімдерін және студенттердің білімін тексеруге арналған практикалық тапсырмаларды қамтиды. Сонымен қатар, оқулықта интерактивті элементтер мен оңай пайдалануға болатын навигация жүйесі бар.

Жобаның міндеттері:

* 1. Электрондық оқулық құрылымын жасау және бөлімдерді орналастыру.
  2. Дәрістер мен практикалық тапсырмаларды интерактивті форматта дайындау.
  3. Тесттер мен сұрақтар әзірлеу, студенттердің білімін тексеру үшін тиімді жүйе құру.
  4. Python тілінде оқытудың тиімді әдістемелерін зерттеу және оларды оқулыққа енгізу.
  5. Құралдар мен технологияларды таңдау және оқулықтың жұмысының барлық кезеңдерінде тиімді пайдалану.

Жобаның өзектілігі. Білім беру процесінің цифрлық трансформациясы жағдайында электронды оқу материалдары маңызды рөл атқарады. Python бағдарламалау тілі өзінің қарапайымдылығымен, әмбебаптығымен және кең экожүйесімен танымал. Бұл тілді оқу студенттерге тек бағдарламалауды ғана емес, сонымен қатар деректер ғылымы, жасанды интеллект және веб-әзірлеу сияқты салаларда да негіз қалауға мүмкіндік береді. Сонымен қатар, Python-ның қарапайым синтаксисі мен жеңіл үйренілуі студенттерге бағдарламалауға деген қызығушылықтарын арттыруға көмектеседі.

Күтілетін нәтижелер: Жобаның нәтижесінде Python бағдарламалау тілін үйренушілерге арналған толыққанды, тиімді оқулық дайын болады. Бұл оқулықта теориялық бөлімдер, практикалық тапсырмалар, интерактивті тесттер және өзін-өзі тексеру мүмкіндіктері қамтылған. Оқулық студенттердің Python-ды өз бетінше үйренуіне немесе оқу процесінде пайдалануға арналған.

# 1.2 Құралдарды таңдау

Жобаны іске асыру үшін пайдаланылған негізгі құралдар мен технологиялар:

* 1. Visual Studio Code - бұл танымал, көпфункционалды код редакторы. Ол HTML, CSS, JavaScript, Python сияқты бағдарламалау тілдеріне қолдау көрсетеді және көптеген кеңейтімдер мен плагиндер арқылы жұмыс процесін оңтайландыруға мүмкіндік береді. Жоба үшін VS Code HTML, CSS және JavaScript кодтарын жазу мен өңдеуге ыңғайлы болды[1].
  2. HTML мен CSS - веб-беттердің құрылымы мен көрінісін жасауға арналған негізгі технологиялар. HTML арқылы оқулықтың құрылымы құрастырылып, CSS арқылы оның стилі мен визуалды көрінісі жасалды. HTML және CSS оқулықтың парақтарын орналастыру мен безендіру үшін қолданылды[2].
  3. JavaScript - веб-беттерде интерактивтілік пен динамикалық функционалдылықты қамтамасыз ететін бағдарламалау тілі. Жоба барысында JavaScript интерактивті тесттер мен тапсырмаларды орындау үшін қолданылды. Студенттердің жауаптары тексеріліп, нәтижелер көрсетілді. Сонымен қатар, динамикалық мазмұнды жүктеу үшін де JavaScript пайдаланылды[3].
  4. Fetch API - веб-сервермен асинхронды түрде байланысуға арналған JavaScript интерфейсі. Бұл құрал парақты қайта жүктемей деректерді жүктеуге мүмкіндік берді. Жоба барысында Fetch API дәрістерді, тесттер мен тапсырмаларды жүктеу үшін қолданылды[4].
  5. Chrome браузері - веб-беттерді тестілеу және күйін келтіру үшін қолданылатын танымал браузер. Оның әзірлеуші құралдары кодтың қателіктерін табуға, браузердің жүктелу уақытын бақылауға мүмкіндік береді. Жоба барысында Chrome браузері тестілеу мен күйін келтіру үшін қолданылды[5].

Бұл құралдар мен технологиялар оқулықты құру үшін ыңғайлы әрі тиімді болды. Әрбір құрал өз функциясын орындап, жобаның функционалдығын арттыруға мүмкіндік берді.

# 1.3 Оқулық құрылымының сипаттамасы

Оқулық үш негізгі бөлімге бөлінген:

1. Оқулықтың негізгі бөлімдерінің бірі - дәрістер. Дәрістер Python бағдарламалау тілінің негізгі тақырыптарын қамтиды, олардың саны 10 болып белгіленген. Әрбір дәріс белгілі бір тақырыпқа арналған және ол толыққанды теориялық материалды, код үлгілерін және мысалдарды қамтиды.

Дәрістерді құру кезінде келесі мақсаттар көзделді:

* Python тілінің негізгі принциптерін және синтаксисін түсіндіру.
* Әрбір тақырып бойынша код жазу үлгілері мен практикалық мысалдар келтіру.
* Теория мен практика арасындағы байланысты қамтамасыз ету.
* Әрбір дәріс бойынша түсіндірмелер мен қолжетімді тілде жазылған нұсқаулар студенттердің түсінігін жеңілдетеді.

1. Тесттер. Бұл бөлім оқулықта берілген дәрістер бойынша студенттердің білімін тексеруге арналған тесттерден тұрады. Әрбір дәріске қатысты тесттер дайындалады, олар студенттің тақырыпты қаншалықты меңгергенін анықтайды. Тесттерде бірнеше жауап нұсқасы болады, оның ішінде дұрыс жауапты таңдау қажет. Тест сұрақтары жеңілден күрделіге дейін өзгеріп отырады, сондықтан әр студент өз деңгейіне сәйкес тесттен өту мүмкіндігіне ие болады. Бұл тесттер оқушылардың дәрістерді толық түсінгенін және теориялық білімдерін жақсы меңгергенін тексеруге көмектеседі. Тесттердің нәтижесі студенттің білімін бағалауға, оның қай жерлерде қосымша көмек қажет екенін көрсетуге мүмкіндік береді.
2. Практикалық міндеттер. Студенттер дәрістерде алған теориялық білімдерін практикада қолдануға арналған тапсырмаларды орындайды. Тапсырмалар Python тілінде бағдарламалау дағдыларын дамытуға бағытталған. Оқулықтағы практикалық тапсырмалар үш деңгейге бөлінеді: жеңіл, орташа және күрделі. Жеңіл тапсырмалар студенттерге негізгі синтаксис пен құрылымдарды түсінуге көмектеседі, орташа тапсырмалар Python-ның күрделі конструкцияларымен жұмыс істеуге мүмкіндік береді, ал күрделі тапсырмалар алгоритмдер құру және деректерді өңдеу сияқты күрделі мәселелерді шешуге арналған. Тапсырмаларды орындау барысында студенттер өздерінің дағдыларын жақсартып, код жазу тәжірибесін жинақтайды.

# 1.4 Оқулықты кезең-кезеңмен құру

Бұл оқу құралы студенттерге Python тілінің негізгі түсініктері мен принциптерін түсінуге көмектесетін үш негізгі бөлімнен тұрады: лекциялар, тесттер және практикалық тапсырмалар. Осы бөлімдер арқылы студенттер теориялық білімдерін бекітіп, алған дағдыларын практикада қолдануға мүмкіндік алады.

Лекциялар. Бұл бөлімде Python бағдарламалау тіліне қатысты 15 негізгі лекция ұсынылады. Әрбір лекция теориялық түсіндірмелер, код үлгілері және түсініктемелерден тұрады, бұл студенттерге тақырыпты толыққанды түсінуге мүмкіндік береді.

1. Кіріспе. Кіріспе лекциясында Python бағдарламалау тіліне жалпы шолу жасалады. Python тілінің тарихы, оның ерекшеліктері, қолданылу салалары мен артықшылықтары туралы ақпарат беріледі. Python тілінің әлемдегі ең танымал тілдердің бірі екендігі, оның қолдану аясы мен өндірістегі маңызы талқыланады. Бұл студенттерге тілдің негізгі артықшылықтарын түсінуге және оны үйренуге деген мотивация береді.
2. Айнымалылар. Бұл лекцияда айнымалылардың маңызы мен оларды қолдану принциптері қарастырылады. Айнымалыларды дұрыс пайдалану, олардың мәндерін сақтау мен өзгерту әдістері түсіндіріледі. Python-да айнымалыларды жариялау мен олардың типтеріне қатысты түсініктер беріледі. Оқыту барысында айнымалылардың нақты мысалдары мен олардың қолданылу аясын көрсету арқылы теорияны практикамен байланыстырамыз.
3. Дерек түрлері. Python тілінде қолданылатын деректер түрлері, олардың ерекшеліктері мен қолданылу аясын түсіндіру мақсаты көзделеді. Студенттерге деректер түрлерінің, мысалы, бүтін сандар (int), ондық сандар (float), жолдар (string), және логикалық мәндер (bool) туралы толық ақпарат беріледі. Python-ның деректер типтерінің еркіндігі мен қарапайымдылығы студенттердің бағдарламалау тілінде жұмыс істеу дағдыларын дамытуға ықпал етеді.
4. Сандар. Python тілінде сандарды қолдану және олардың ерекшеліктері туралы түсінік беріледі. Бүтін сандар (int), ондық сандар (float) және күрделі сандар (complex) туралы ақпарат беріледі. Сонымен қатар, студенттер сандарды арифметикалық операциялармен өңдеу әдістерін үйренеді. Сандармен жұмыс істеу барысында кездесетін қателіктер мен оларды болдырмау жолдары қарастырылады.
5. Жолдар (strings) - Python бағдарламалау тілінде кеңінен қолданылатын деректер типі. Бұл лекцияда жолдармен жұмыс жасау әдістері, олардың операциялары мен манипуляциялары түсіндіріледі. Студенттерге жолдарды біріктіру, бөлу, іздеу және өзгерту секілді негізгі операциялар көрсетіледі.
6. Логикалық мәндер. Бұл лекцияда булев типі мен оның мәндері, логикалық операторлар (AND, OR, NOT) түсіндіріледі. Логикалық мәндерді пайдалану бағдарламаның негізгі жұмыс істеу логикасын құрастыруда маңызды рөл атқарады. Булев типінің негізгі ерекшеліктері мен қолдану аясын талқылай отырып, студенттерге шартты операторларды қолдану жолдары көрсетіледі.
7. Операторлар - бағдарламалау тілінде мәліметтерді өңдеуге арналған құралдар. Бұл лекцияда арифметикалық, салыстырмалы және логикалық операторлар түсіндіріледі. Студенттерге операторлардың әртүрлі түрлерінің қандай жағдайда қолданылатыны көрсетіледі.
8. Тізімдер (list): Python тілінде тізімдер (list - мәліметтерді сақтау және оларды ұйымдастырудың тиімді әдісі. Бұл лекцияда тізімдермен жұмыс жасау тәсілдері қарастырылады. Студенттерге тізімдерге элементтер қосу, жою, іздеу және сұрыптау әдістері көрсетіледі.
9. If және else шарттық операторлары. Шартты операторлар бағдарламаның орындалу жолын өзгерту үшін қолданылады. Бұл лекцияда if, elif, және else операторларының маңызы мен қолданылуы көрсетіледі. Шартты операторлар көмегімен бағдарламаға нақты шарттарды орындауға және қажет жағдайда әртүрлі әрекеттер жасауға мүмкіндік беріледі.
10. WHILE циклі. Циклдер мәліметтерді қайталап өңдеуге мүмкіндік береді. Бұл лекцияда while циклін қолдану принциптері түсіндіріледі. Студенттерге циклдардың орындалуын басқару, циклдерде кездесетін қателіктер мен оларды болдырмау жолдары көрсетіледі.
11. FOR циклі. Бұл лекцияда for циклі мен оның Python-дағы ерекшеліктері түсіндіріледі. Студенттерге for циклімен тізімдер, сөздіктер, жолдар және басқа да коллекциялармен жұмыс істеу тәсілдері көрсетіледі.
12. Функциялар. Функциялар - Python тілінің маңызды құрылымдарының бірі. Бұл лекцияда функцияларды қалай анықтап, оларды қалай қолдануға болатыны түсіндіріледі. Студенттерге функциялардың аргументтерін және қайтару мәндерін дұрыс пайдалану әдістері көрсетіледі.
13. Lambda функциялары. Lambda функциялары - қысқа және анонимді функциялар. Бұл лекцияда lambda функцияларының құрылымы, қолдану мүмкіндіктері мен практикалық мысалдары қарастырылады.
14. Массивтер. Лекцияда массивтер мен олардың Python-дағы қолдану жолдары қарастырылады. Студенттерге массивтерді тиімді пайдалану, олардың индекстері мен әртүрлі операциялары туралы түсінік беріледі.
15. Мұрагерлік. Мұрагерлік - объектіге бағытталған бағдарламалаудың негізгі принциптерінің бірі. Бұл лекцияда мұрагерлік механизмдері түсіндіріледі, яғни бір кластың қасиеттері мен әдістерін екінші классқа беру мүмкіндігі. Студенттерге мұрагерлік арқылы кодты қайта пайдалану мен оның тиімділігін арттыру жолдары көрсетіледі.

# 1.5 Файл құрылымын құру

Файл құрылымын дұрыс құру - жобаның тиімділігі мен қолайлылығын қамтамасыз ету үшін маңызды. Әрбір жоба үшін файл құрылымы жүйелі және түсінікті болуы керек. Бұл әсіресе үлкен жобаларда өте маңызды, өйткені дұрыс ұйымдастырылған құрылым жобаны дамыту мен қолдауды жеңілдетеді.

Оқулықты әзірлеу үшін келесі файл құрылымын ұсынуға болады:

index.html - Бұл негізгі HTML беті, барлық басқа беттерге сілтемелерді қамтитын басты бет болып табылады. Оқулықтың бастапқы парағы осы беттен басталады. Мұнда жалпы навигация мен құрылым анықталады [6].

styles.css - Оқулықтың стилін анықтайтын CSS файлдары. Бұл беттердің көрінісін, макетін және түстерін реттейді. Барлық сыртқы стильдер осы файлда жинақталады [7].

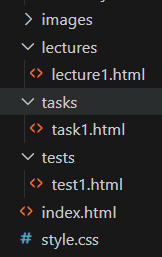
script.js - Бұл файлда барлық JavaScript кодтары орналасады. Мұнда интерактивтілік, тесттерді тексеру және динамикалық мазмұнды жүктеу кодтары жазылады. Барлық оқулықтың функционалдылығы осы файлда жүзеге асырылады [8].

lectures/ - Бұл каталог ішінде дәрістердің жеке HTML беттері сақталады. Әрбір дәріс өзінің жеке HTML файлына ие болады. Мысалы:

* lecture1.html
* lecture2.html
* tests/ - Тесттердің жеке беттері мен олардың нәтижелерін сақтау үшін бұл каталог қолданылады. Әрбір тест жеке файлға бөлінеді, мысалы:
* test1.html
* test2.html
* tasks/ - Практикалық тапсырмаларды қамтитын каталог. Бұл каталог ішінде студенттерге арналған тапсырмалар болады. Мысалы:
* task1.html
* task2.html
* images/ - Суреттер, иконкалар, логотиптер және басқа да медиа файлдарын сақтау үшін қолданылатын каталог. Бұл файлдар оқулықтың визуалды элементтерін құрамдастырады. Мысалы:
* image1.jpg
* icon.png

Бұл құрылым оқулықты логикалық түрде ұйымдастыруға және оның әрбір компонентін оңай табуға мүмкіндік береді. Әрбір файл мен каталогтың өзіндік функциясы болуы керек, бұл жұмысты жеңілдетіп, болашақта кодты дамыту мен жөндеуді оңайырақ етеді.

Файл құрылымын құру Сурет 1-де көрсетілген.



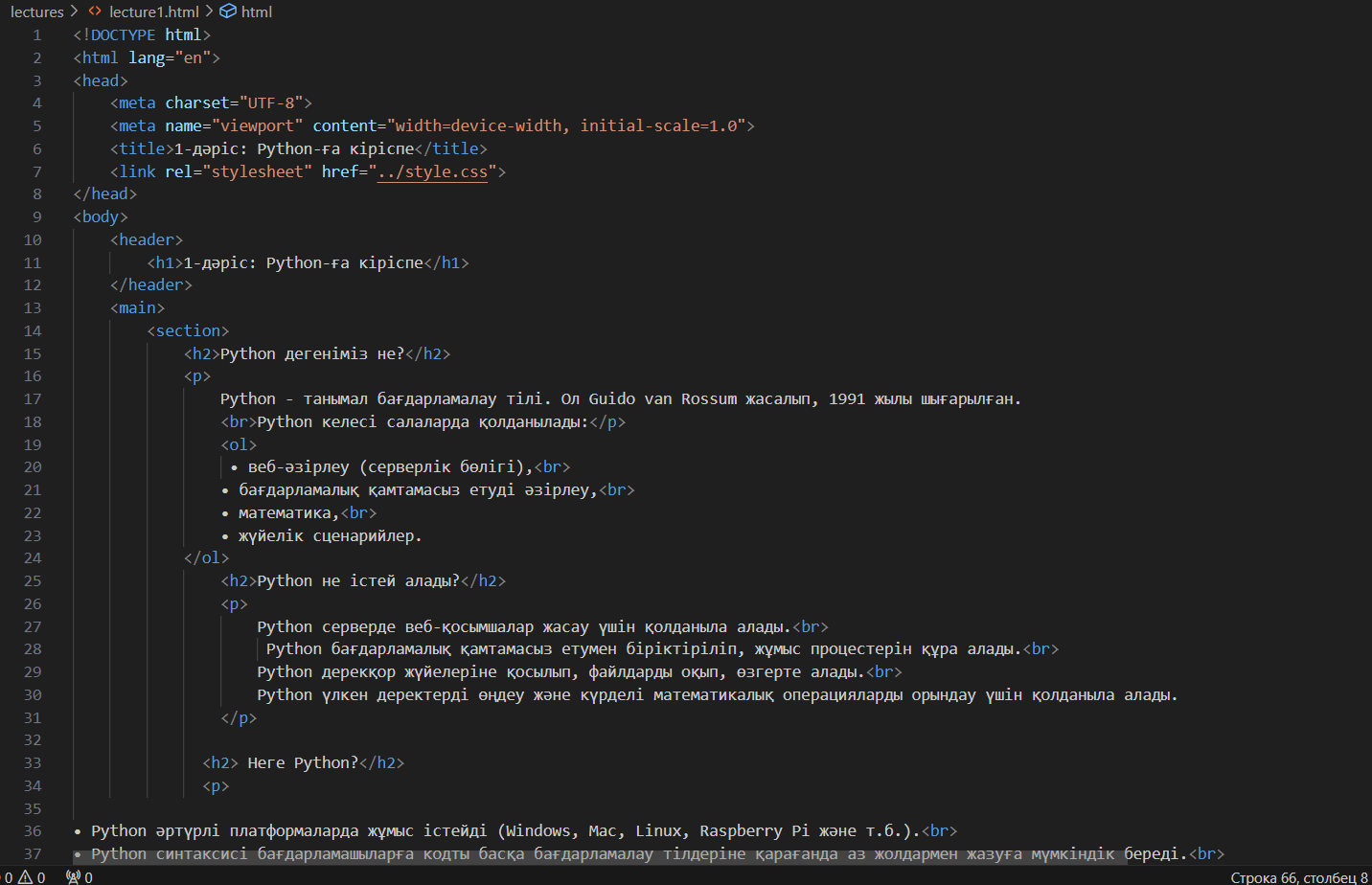
Сурет 1 файлдық құрылым

Бірінші кезеңде жобаның файлдық құрылымы ұйымдастырылды:

* images/ - Суреттер, иконкалар, логотиптер бар қалта.
* lectures/ - HTML дәріс файлдары бар қалта;
* tasks/ - практикалық тапсырмалар қалтасы;
* tests/ - тест қалтасы;
* index.html -оқулықтың басты беті;
* style.css – CSS стильдері бар файл;

# 1.5.1 Дәрістерді әзірлеу

Әр дәріс үшін жеке HTML файлы жасалды. 1-ші дәріс Python бағдарламалау тілімен таныстырады. «Python дегеніміз не?», «Python не істей алады?», «неге Python?», «Python синтаксисі басқа бағдарламалау тілдерімен салыстырғанда қандай?», оның құрылу тарихы, қолдану салалары және негізгі мүмкіндіктері қарастырылады. Python веб-әзірлеу (сервер бөлігі), бағдарламалық жасақтама жасау, математика және жүйелік сценарийлермен жұмыс істеу үшін қолданылады. Оның артықшылықтары да талқыланады: әмбебаптық, әртүрлі платформаларда жұмыс істеу мүмкіндігі (Windows, Mac, Linux), ағылшын тіліне ұқсас қарапайым синтаксис және жазғаннан кейін бірден кодты орындау мүмкіндігі. Python жылдам прототиптеу үшін де, дайын бағдарламалық жасақтаманы жасау үшін де ыңғайлы екендігі баса айтылған. Python синтаксисін басқа тілдермен салыстыруға ерекше назар аударылады: бұл оңайырақ, өйткені бұйра жақшалардың орнына шегіністерді қолданады және командаларды жаңа жолдармен аяқтайды. Мысал ретінде 1-ші дәрістің құрылымы 2 суретте көрсетілген.



Сурет 2 1-ші дәрістің құрылымдық коды

1-ші дәрістің құрылымдық кодын жазу нәтижесінде келесі қадамдар жасалды:

* 1. <!DOCTYPE html> - HTML5 құжатының түрін анықтайды. [9]
  2. <head> - Мета-ақпаратты, бет атауын, сыртқы стильдер файлын қосады [9].
  3. <body> - Негізгі ақпаратты орналастыратын бөлім [9] .
  4. <header> - Әр лекцияның тақырыбы, мысалы: «1-дәріс: Python-ға кіріспе», «2-дәріс: Python-дағы айнымалылар» [9].
  5. <main> - Негізгі бөлімде әр лекцияның негізгі тақырыбы мен оның мазмұны көрсетіледі [9].

Дәріс <body> бөлімінде жазылатын болғандықтан, бұл бөлімді талқылайық.

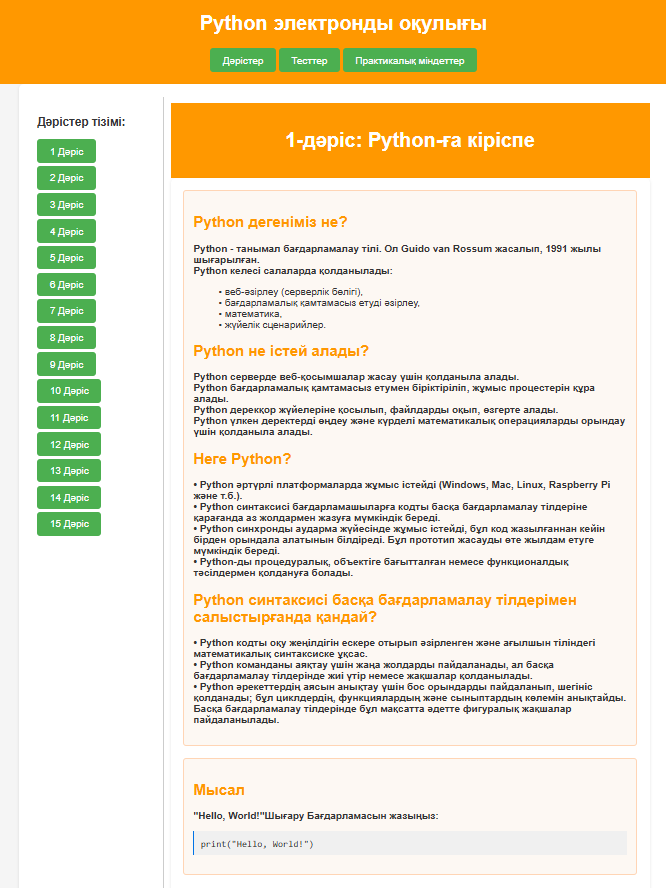
<body> бөлімі - бұл HTML құжатының негізгі мазмұнын қамтитын бөлім. Бұл жерде веб-беттегі барлық көрінетін элементтер орналасады.

Әрбір лекцияның құрылымында <body> бөлімі екі негізгі элементтен тұрады [9]:

1. <header> - беттегі тақырып, яғни барлық дәрістердің басталуы осы бөлімде көрсетіледі. Мысалы, бірінші дәрістің тақырыбы «1-дәріс: Python-ға кіріспе», екінші дәрістің тақырыбы «2-дәріс: Python-дағы айнымалылар» және т.б [9].
2. <main> - негізгі бөлім, бұл жерде дәрістің негізгі мазмұны мен ақпараттары көрсетіледі. Бұл бөлім бірнеше бөлімнен тұруы мүмкін, мысалы [9]:

* <section> - әр түрлі тақырыптар мен бөлімдерді қамтитын тег. Әр дәрісте әртүрлі тақырыптар мен бөліктер орналасады [9]. Мысалы:
* <h2> - бөлімдердің тақырыптары. Мысалы, «Python дегеніміз не?», «Python синтаксисі», «Python функциялары» және т.б [9].
* <p> - бөлімдегі мәтіндер мен түсініктемелер [9].
* <ol> - нөмірленген тізімдер. Бұл тізімдер Python тілінің негізгі мүмкіндіктерін немесе қолданылатын салаларын көрсетеді [9].

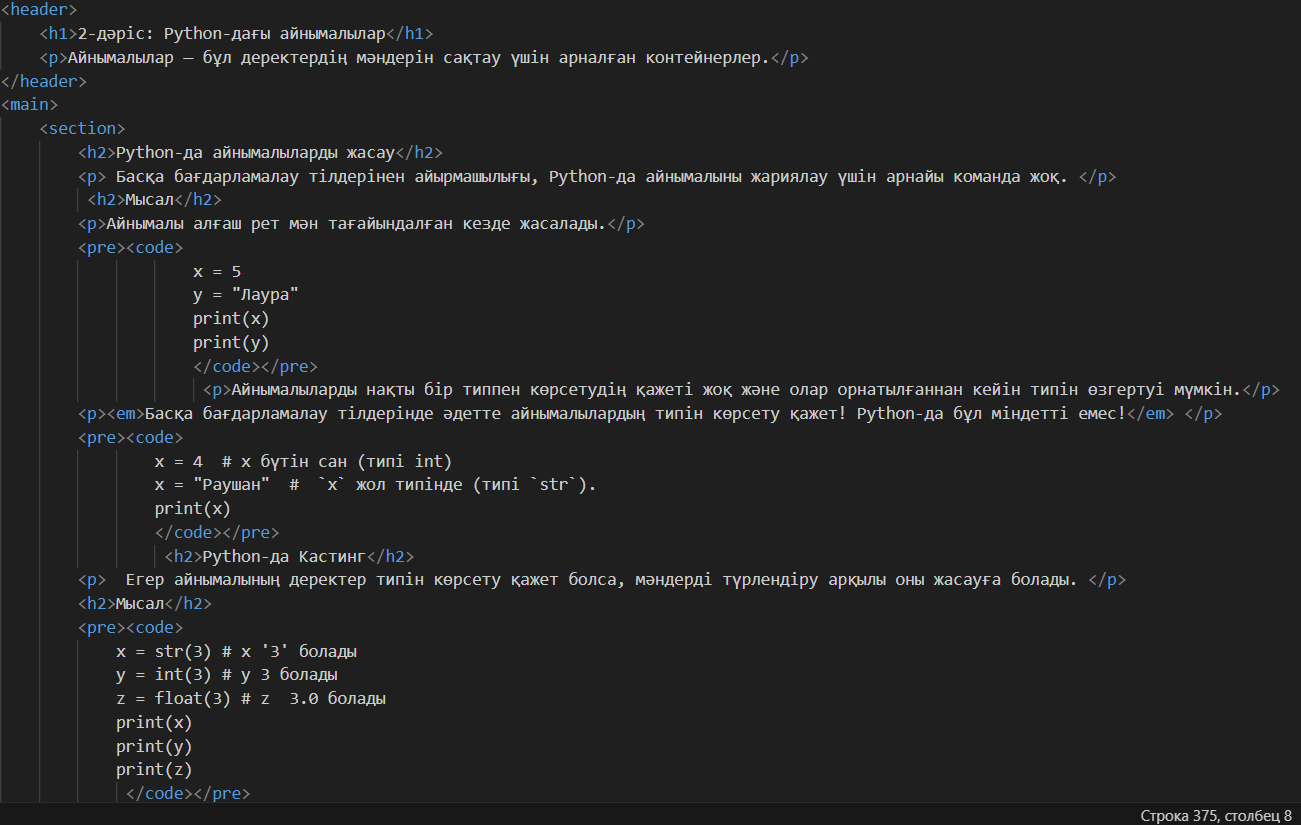
Жоғарыда келтірілген кодты браузерде көргенде қалай көрінетінін қарастырайық. HTML код веб-беттің құрылымын, контентін және функционалдығын анықтайды. Қорытындылай келе 1-дәріс мета деректерді қоса 59 жолдан тұрады. Веб-бет нақты уақыт режимінде көрсетілу үшін кодты Live Server арқылы браузерде ашу керек. 1-ші дәрістің веб-беті 3-ші суретте көрсетілген.



Сурет 3 1-ші дәрістің веб-беті

Live Server - Visual Studio Code редакторында жұмыс істейтін кеңейтім болып табылады. Ол HTML, CSS және JavaScript файлдарындағы өзгерістерді бірден браузерде көрсетуге мүмкіндік береді. Бұл құрал веб-дамыту процессін айтарлықтай жеңілдетеді, себебі әрбір өзгеріс жасаған сайын бетті қайта жүктемей-ақ өзгерістерді бірден көруге болады [10].

2-ші дәріс: «Python-дағы айнымалылар» деп аталады. Дәріс Python-дағы айнымалыларға арналған. Айнымалылар деректерді сақтауға арналған контейнерлер болып саналады және мәнді алғаш тағайындаған кезде автоматты түрде жасалады. Көптеген басқа бағдарламалау тілдерінен айырмашылығы, Python-да айнымалы түрін алдын-ала көрсетудің қажеті жоқ. Бағдарламаны орындау барысында айнымалы оның түрін өзгерте алады. Мысалы, айнымалыға алдымен санды, содан кейін жолды тағайындауға болады және бұл қателерді тудырмайды. Кастинг тұжырымдамасы да түсіндіріледі-бағдарламашы қажет болған жағдайда айнымалының деректер түрін нақты көрсете алады [11]. 2-ші дәрістің құрылымдық коды 4-ші суретте көруге болады.



Сурет 4 2-ші дәрістің құрылымдық коды

Python-да айнымалыларды жасау. Басқа бағдарламалау тілдерінен айырмашылығы, Python-да айнымалыны жариялау үшін арнайы команда жоқ [11].

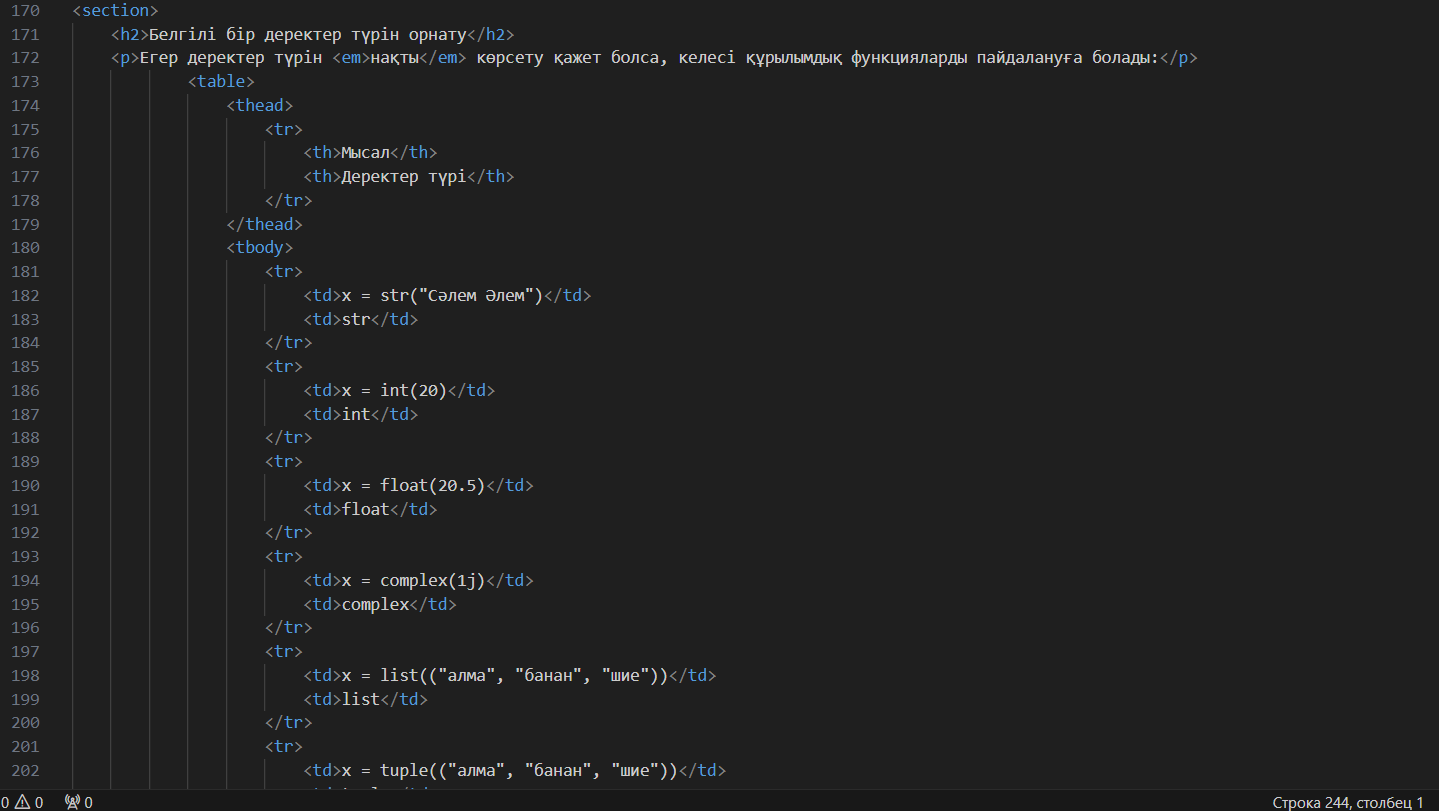
Басқа бағдарламалау тілдерінде әдетте айнымалылардың типін көрсету қажет! Python-да бұл міндетті емес!

Айнымалы алғаш рет мән тағайындалған кезде жасалады.2-ші дәрісте мысалдар бөлімін (<pre><code>) жазылған. <pre><code> тегтері Python кодтарын көрсету үшін қолданылады. Код көрсетілген кезде оның форматталған күйі сақталады.Тақырып <h2></h2> тегтерінде жазылып, ақпарат <p></p> тегтерінде орнатылады. 1-ші дәріске қарағанда 2-ші дәріс 376 жолды қамтиды. Дәрістің веб-бетін 5-ші суретте баяндалған.



Сурет 5 2-ші дәрістің веб-беті

3-ші дәріс: «Python-дағы дерек түрлері» тақырыбын қамтиды. Бұл дәріс Python-дағы кіріктірілген деректер түрлерін және олардың бағдарламалаудағы маңыздылығын түсіндіреді. Деректер түрі айнымалының қандай мәндерді қабылдай алатынын және оларға қандай операциялар рұқсат етілетінін анықтайды [12]. 3-ші дәрістің құрылымдық коды 6-шы суретте көруге болады.

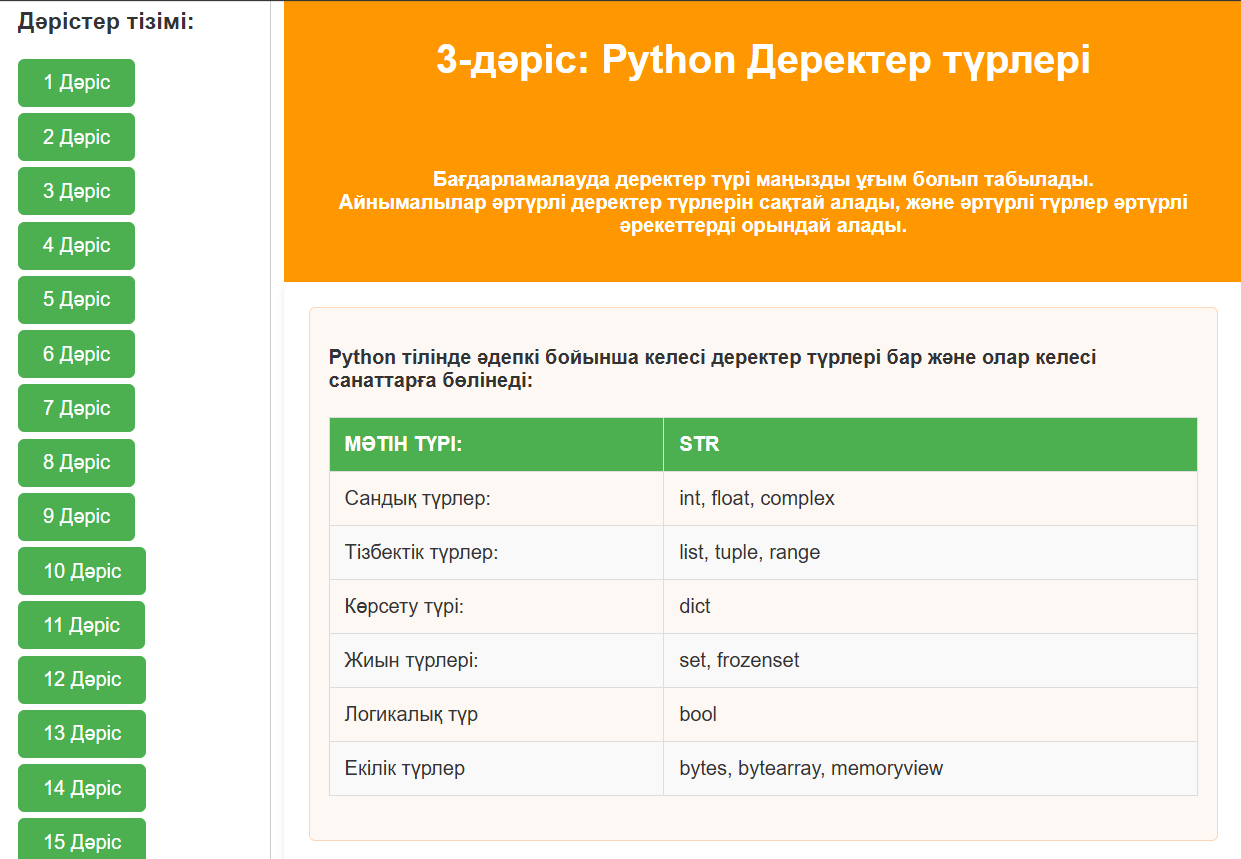


Сурет 6 3-ші дәрістің құрылымдық коды

Python-да мәтіндік, сандық, реттілік, дисплей, жиындар, логикалық және екілік сияқты деректер түрлерінің көптеген санаттары бар. Python мән берілген кезде айнымалының деректер түрін автоматты түрде анықтайды, бұл кодпен жұмыс істеуді жеңілдетеді .

Алайда, қажет болған жағдайда, әзірлеуші арнайы конструкторлық функцияларды қолдана отырып, белгілі бір деректер түрін нақты орната алады. Деректер түрлері ақпаратты тиімді басқаруға және өңдеуге мүмкіндік беретін бағдарламаларды құруда шешуші рөл атқарады .

HTML-де кестелер мәліметтерді жолдар мен бағандарға (строки и столбцы) бөлу үшін қолданылады. Кесте <table> тегінің көмегімен жасалады. Кестенің әрбір жолы <tr> (table row) арқылы, ал әрбір бағаны немесе ұяшығы <td> (table data) арқылы анықталады. Егер тақырыптық ұяшықтар (заголовки) керек болса, <th> (table header) тегін қолданамыз [12].

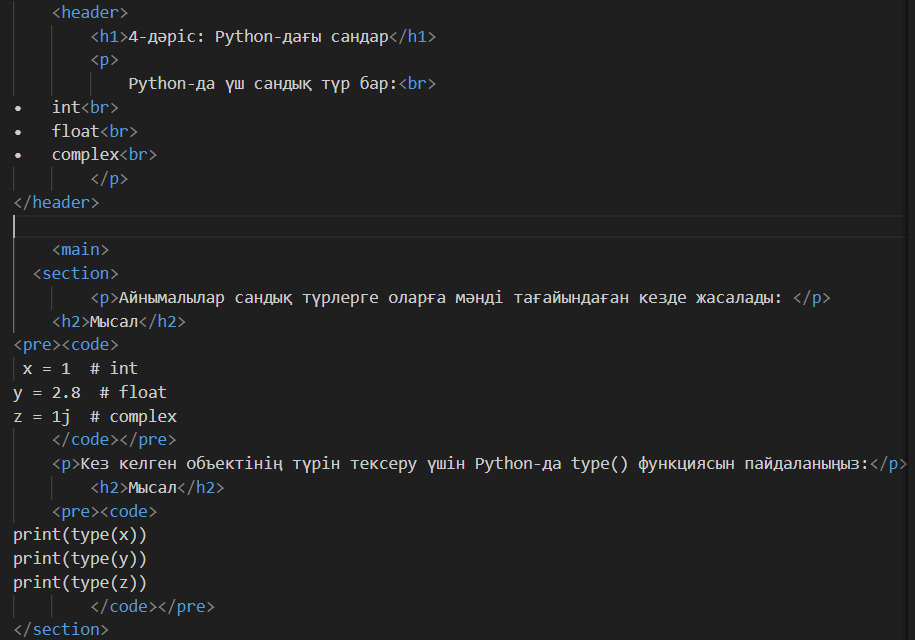


Сурет 7 3-ші дәрістің веб-беті

4-ші дәріс: «Python-дағы сандар» деп аталады. Бұл дәріс әртүрлі математикалық және есептеу есептерімен жұмыс істеуде маңызды рөл атқаратын Python-дағы сандық деректер түрлері туралы айтады [13].

Python үш негізгі сандық түрді қолдайды: бүтін сандар (int), өзгермелі нүкте сандары (float) және күрделі сандар (complex). Бүтін сандар Бөлшек бөлігі жоқ оң және теріс мәндерді білдіреді және олардың ұзындығы шектелмейді. Өзгермелі нүкте сандары бөлшек мәндерді, соның ішінде үлкен және кіші сандармен жұмыс істеуге мүмкіндік беретін жазудың экспоненциалды формаларын көрсету үшін қолданылады. Күрделі сандар, өз кезегінде, нақты және ойдан шығарылған бөліктерді қамтиды, бұл оларды ғылыми және инженерлік есептеулерге ыңғайлы етеді [13].

Python мәнді тағайындау кезінде сандық айнымалының түрін автоматты түрде анықтайды, бұл әзірлеушілердің жұмысын жеңілдетеді. Алайда, егер қажеттілік туындаса, Int(), float() және complex () сияқты арнайы әдістер арқылы бір сандық түрді екіншісіне түрлендіруге болады. Күрделі сандарды басқа түрлерге түрлендіру мүмкін емес екенін ескеру маңызды. Сондай-ақ, Python-да кездейсоқ сандармен жұмыс істеу мүмкіндігі бар. Ол үшін кездейсоқ мәндерді құруға мүмкіндік беретін кездейсоқ модуль қолданылады, ол көбінесе ойындарда, статистикада және басқа салаларда қолданылады [13]. 4-ші дәрістің құрылымдық коды 8-ші суретте көруге болады.



Сурет 8 4-ші дәрістің құрылымдық коды

4-ші дәрістің құрылымдық кодын жазу нәтижесінде келесі қадамдар жасалды:

1. x = 1 - Бұл айнымалыға бүтін сан (int) мәні беріледі. Мұнда 1 деген бүтін санды x айнымалысына сақтаймыз [13].

2. y = 2.8 - Бұл айнымалыға бөлшек сан (float) мәні беріледі. Мұнда 2.8 деп жазылған санды y айнымалысына сақтаймыз [13].

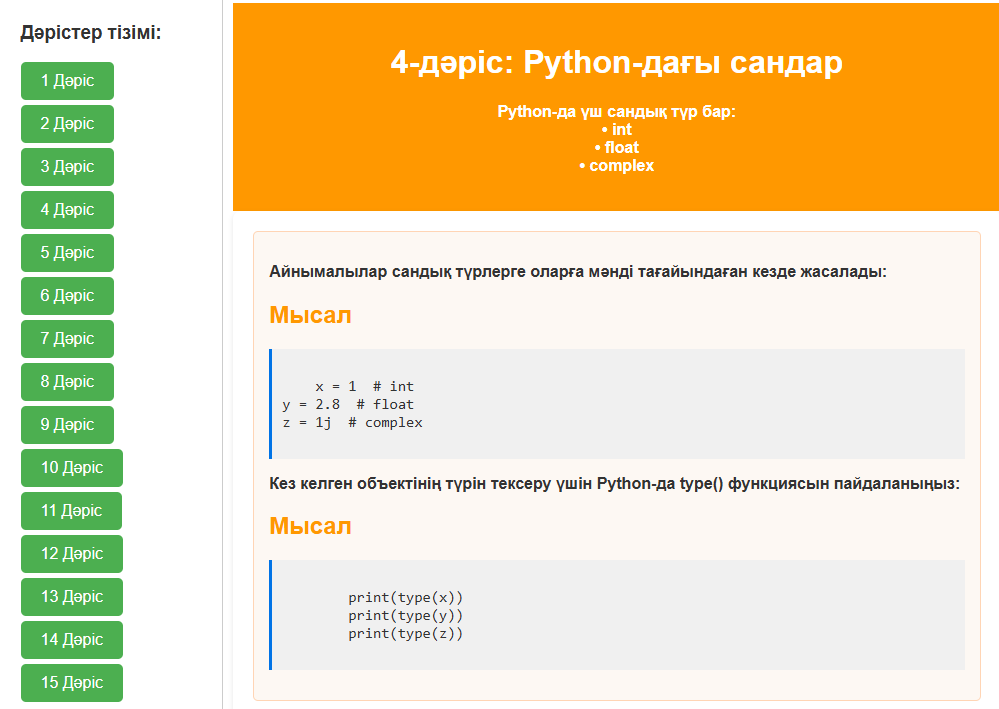
3. z = 1j - Бұл айнымалыға күрделі сан (complex) мәні беріледі. 1j мнималық бөлшекті білдіреді, және оны z айнымалысына сақтаймыз [13].

4. print(type(x)) - Бұл жолда x айнымалысының типін анықтау үшін type() функциясы қолданылады. Нәтижесінде x айнымалысының типі экранға шығарылады [13].

5. print(type(y)) - Бұл жолда y айнымалысының типін анықтаймыз. Нәтижесінде, y айнымалысының типі экранға шығады [13].

6. print(type(z)) - Бұл жолда z айнымалысының типін анықтаймыз. z айнымалысының типі де экранға шығады [13].

Python айнымалының деректер түрін автоматты түрде анықтайды, оны құру кезінде нақты түр көрсетілмейді. Айнымалы түрін тексеру үшін type () ' функциясы қолданылады. Бұл айнымалылармен жұмыс істеуді жеңілдетеді және деректер түрлерін түрлендіруді жеңілдетеді. Дәрістің веб-бетін 9-шы суретте баяндалған [13].



Сурет 9 4-ші дәрістің веб-беті

5-ші дәріс: «Python-дағы жолдар» деп аталады. Бұл дәрісте Python тіліндегі жолдар мен олардың қолданылуы туралы кеңірек түсініктер беріледі. Жолдар, әдетте, бір немесе қос тырнақшаға алынады, және бұл таңбалардың кез келгені жолды белгілеу үшін қолданылуы мүмкін [14].

Мысалы, 'Сәлем' және "Сәлем" бірдей мағынаны білдіреді. Жолды айнымалыға тағайындау кезінде сол айнымалының атына теңдік белгісі арқылы мән беріледі. Бұл оңай әрі қарапайым процесс, әрі кодтың оқылуын жеңілдетеді [14].

Кейде көпжолды жолдарды жазу қажет болуы мүмкін. Бұл үшін үш тырнақшалар қолданылады, бұл арқылы мәтін бірнеше жолдан тұрып, әрбір жол өз орнында сақталады. Python-да жолдар тексеруге, өңдеуге және өзгертуге арналған көптеген әдістер бар. Мысалы, жолдың ұзындығын білу үшін len() функциясы қолданылады, ол жолдағы таңбалардың санын қайтарады [14].

Сондай-ақ, жолдардың ішіндегі элементтерге қол жеткізу үшін төртбұрышты жақшалар пайдаланылады, яғни жолдар массивтер ретінде жұмыс істейді. Python тілінде жеке таңба да жол ретінде қарастырылады, себебі ол ұзындығы 1 болатын жол болып табылады [14].

Сонымен қатар, in және not in операторлары жолдағы белгілі бір сөздің бар-жоғын тексеру үшін қолданылады, бұл Python тіліндегі мәтіндерді талдау мен өңдеуде жиі қолданылатын құралдар болып табылады. Дәрістің веб-бетін 10-шы суретте көруге болады [14].



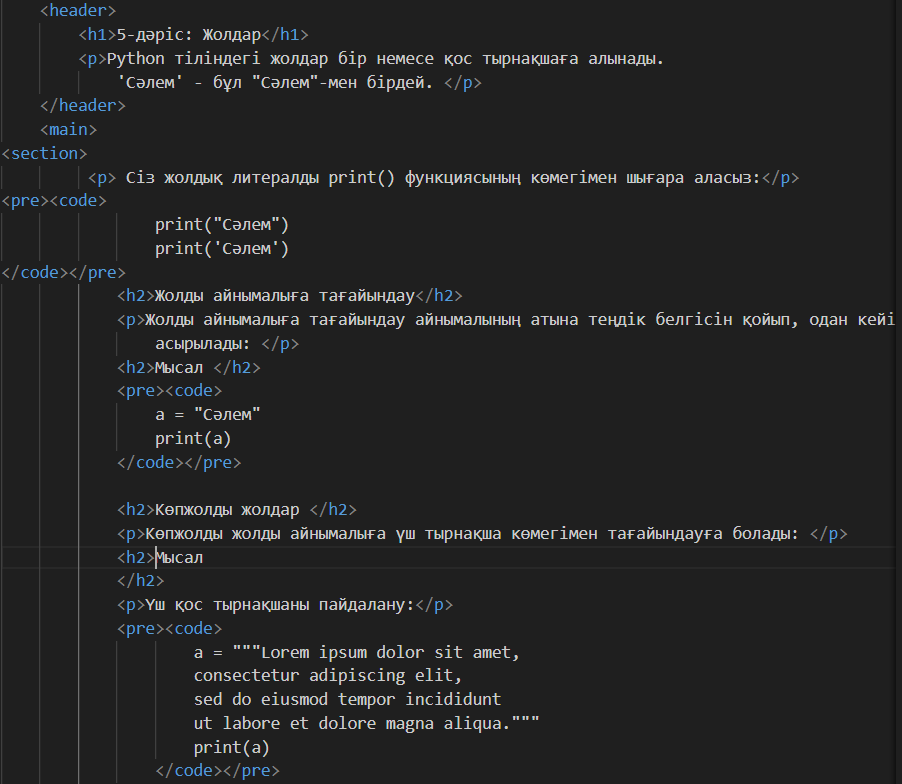
Сурет 10 5-ші дәрістің веб-беті

Python тілінде жолдарды қалай шығарып, айнымалыларға тағайындау және көпжолды жолдарды қалай қолдануға болатыны код құрылымында көрсетіледі [14].

1. Жолды шығару: Кодтың алғашқы бөлігінде жолды print() функциясы арқылы шығару көрсетілген. print() функциясы ішінде жазылған мәтінді экранға шығарады. Мысалы, "Сәлем" және 'Сәлем' деген екі түрлі жолды шығару кезінде екі нәтижені де экранда көре аласыз, себебі бір немесе қос тырнақшалар бірдей мағына береді [14].

2. Жолды айнымалыға тағайындау: Кодтың келесі бөлігінде жолды айнымалыға тағайындау көрсетілген. Мұнда айнымалының атына теңдік белгісі арқылы жолды тағайындап, кейін оны print() арқылы шығару керек. Мысалы, айнымалы a арқылы "Сәлем" жолын сақтау және оны экранға шығару жүзеге асырылады [14].

3. Көпжолды жолдар: Python-да көпжолды жолдарды үш тырнақшалар арқылы жазуға болады. Мұнда кодта үш қос немесе үш бір тырнақшамен көпжолды мәтіндер жасалып, оларды айнымалыға тағайындап, экранға шығару көрсетілген. Бұл әдіс жолдар арасындағы үзілістерді сақтай отырып, мәтіннің форматын өзгеріссіз береді [14]. 5-ші дәрістің құрылымдық коды 11-ші суретте көруге болады .



Сурет 11 5-ші дәрістің құрылымдық коды

6-шы дәріс: «Python-дағы Булев типі» деп аталады. Бұл дәріс Python - да жолдарды кесу процесін түсіндіреді. Жолдарды кесу индекстер ауқымын көрсете отырып, жолдан ішкі жолдарды шығаруға мүмкіндік береді. Кесуді орындау үшін қос нүкте синтаксисі қолданылады, мұнда бірінші индекс кесудің басталуын, ал екіншісі оның соңын көрсетеді. Екінші индекс нәтижеге қосылмайтынын есте ұстаған жөн [15].

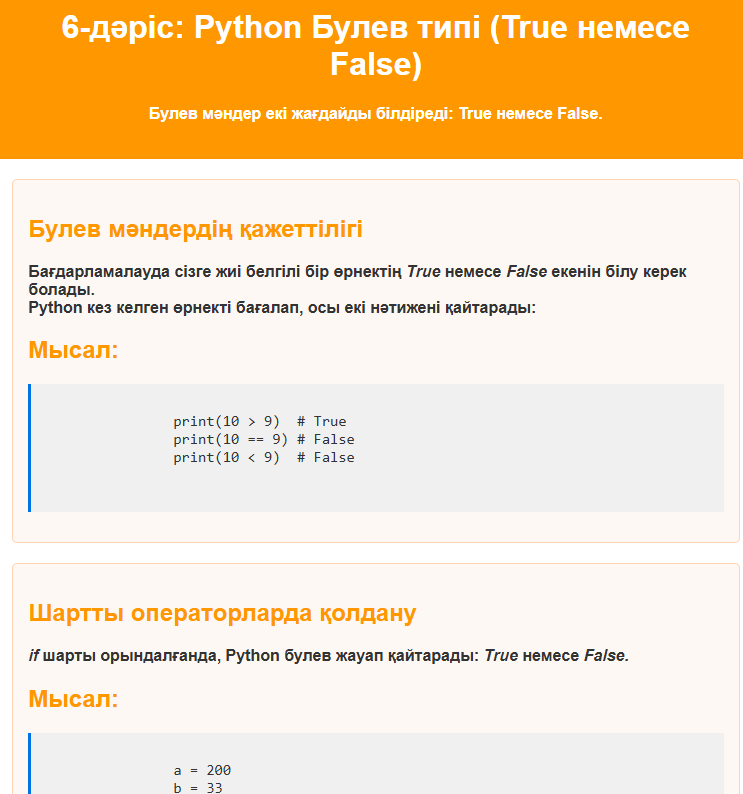
Бастапқыда екі индексті көрсете отырып, жолдың бір бөлігін қалай алуға болатындығы түсіндіріледі. Мысалы, сіз 2-ден 4-ке дейінгі позициядан таңбаларды ала аласыз, ал 5-позициядағы таңба кесуге түспейді. Сондай-ақ, индекстеу 0-ден басталады және жолдың бірінші таңбасында 0 индексі болады.

Егер бастапқы индекс көрсетілмесе, кесу жолдың басынан басталады және мысалда жолдың басынан бастап 6-орынға дейінгі барлық таңбаларды қалай алуға болатындығы көрсетілген (6-шы таңбаны қоспағанда) [15].

Егер соңғы индекс көрсетілмесе, кесу жолдың соңына дейін жалғасады. 2-позициядан жолдың соңына дейін кесу мысалы да келтірілген [15].

Сонымен қатар, жолды соңынан кесу үшін теріс индекстерді қолдану қарастырылады. Теріс индекстер жолдың соңынан бастап таңбаларға қол жеткізуге мүмкіндік береді, мұнда - 1 индексі соңғы таңбаны, - 2 - соңғы таңбаны және т.б. білдіреді. Бұл жолдың соңында икемді жұмыс істеуге мүмкіндік беретін пайдалы мүмкіндік [15].

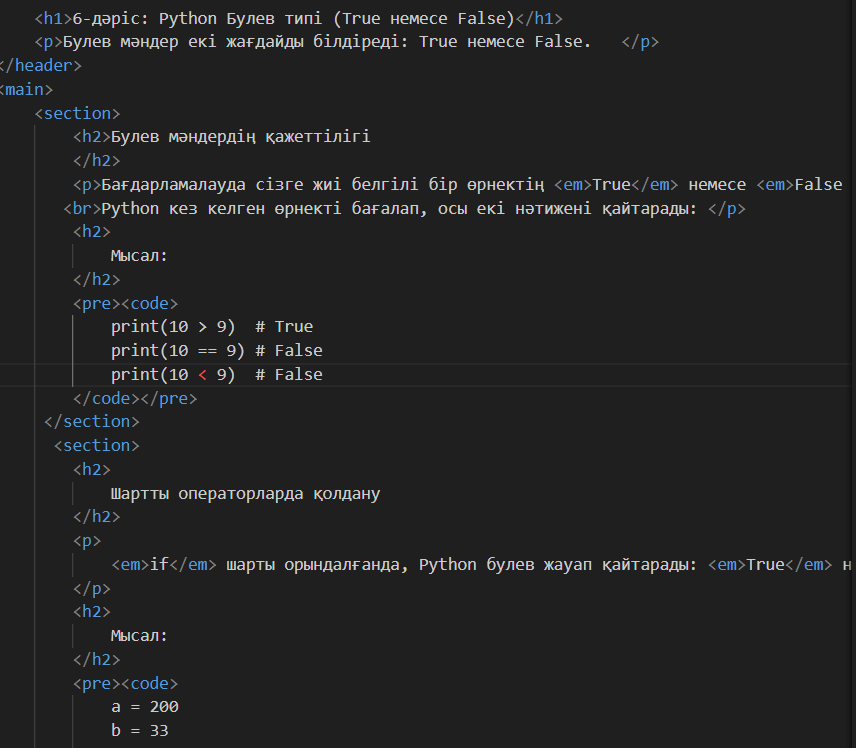
Дәріс Python-да ішкі жолдармен қалай жұмыс істеу керектігін түсінуге көмектеседі, бұл бағдарламалаудағы мәтіндік деректерді басқарудың маңызды құралы [15]. Дәрістің веб-бетін 12-ші суретте көруге болады.



Сурет 12 6-шы дәрістің веб-беті

Булев мәндердің қажеттілігі: Булев мәндер бағдарламалауда өте маңызды, себебі олар тек екі мәнді ғана қабылдайды: True (Шын) және False (Жалған). Бұл мәндер әртүрлі өрнектер мен шарттарды тексеру үшін қолданылады. Python тілінде өрнектерді бағалаған кезде автоматты түрде осы екі мәннің біреуі қайтарылады. Мысалы, егер сіз 10 > 9 деп сұрасаңыз, Python True деген жауап қайтарады, ал 10 == 9 дегенде False болады [15].

Шартты операторларда қолдану: Булев мәндер if шартты операторы арқылы тексеріледі. Егер шарт шын болса, онда Python True мәнін қайтарады, ал шарт жалған болса, False мәні қайтарады. Мысалы, егер b айнымалысы a айнымалысынан үлкен болса, онда «b a-дан үлкен» деген хабарлама шығарылады, әйтпесе «b a-дан кіші» деген хабарлама көрсетіледі [15]. 6-шы дәрістің құрылымдық коды 13-ші суретте көруге болады.

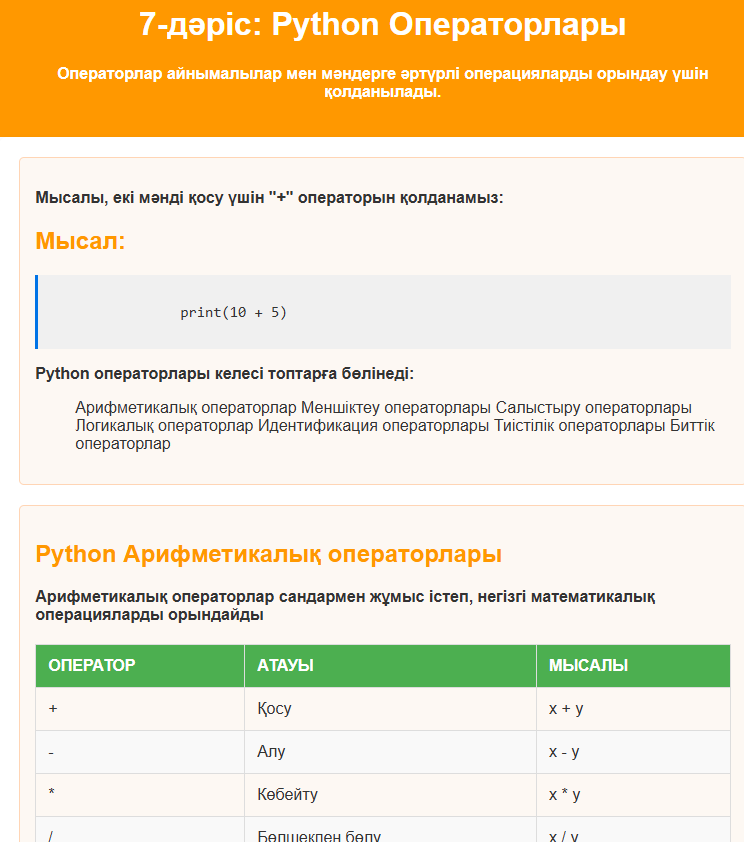


Сурет 13 6-ші дәрістің құрылымдық коды

7-ші дәріс: «Python-дағы оператор» деп аталады .Операторлар Python тілінде айнымалылар мен мәндерге әртүрлі операцияларды орындау үшін қолданылады. Олар бағдарламаның негізгі құрамдас бөлігі болып табылады, өйткені олар деректерді өңдеу, салыстыру және түрлендіру үшін қажетті барлық функцияларды атқарады. Операторлардың көптеген түрлері бар, олар әртүрлі мақсаттарға қызмет етеді, мысалы, арифметикалық, меншіктеу, салыстыру, логикалық және басқа да түрлері. Арифметикалық операторлар сандармен жұмыс істеуге арналған және негізгі математикалық операцияларды орындауға мүмкіндік береді. Бұл операторлар арқылы қосу, алу, көбейту, бөлу, дәрежеге шығару, қалдықты табу секілді амалдарды орындауға болады. Меншіктеу операторлары айнымалыларға мәндер тағайындау үшін қолданылады, сонымен қатар олар айнымалылардың мәндерін өзгертуге мүмкіндік береді. Мысалы, «+=», «-=», «\*=», «/=» сияқты операторлар айнымалының мәнін арттыру, азайту, көбейту немесе бөлу үшін қолданылады [16].

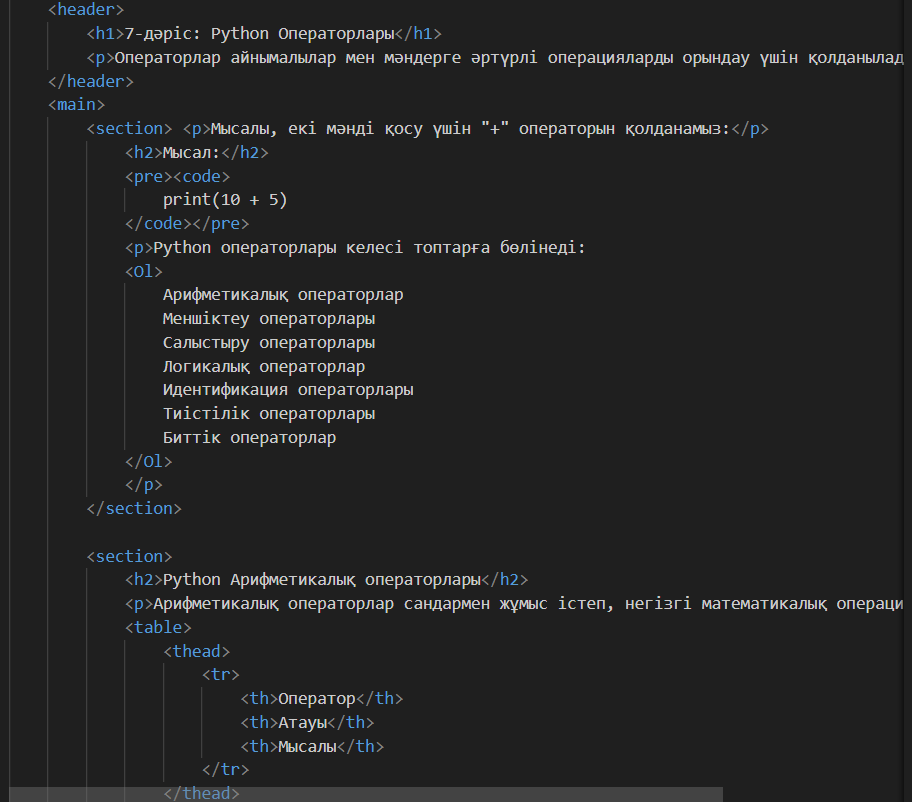
Салыстыру операторлары екі мәнді салыстыру үшін пайдаланылады. Олар арқылы айнымалылардың теңдігін немесе олардың арасындағы қатынасты тексеруге болады. Логикалық операторлар шарттарды біріктіру үшін қолданылады, олар «and», «or», «not» арқылы әртүрлі шарттар арасындағы логикалық байланыстарды орнатуға мүмкіндік береді. Идентификация операторлары объектілердің бірдей жад орнында орналасқанын тексеру үшін қолданылады, ал тиістілік операторлары тізбектің объект ішінде бар-жоғын анықтайды [16].

Биттік операторлар екілік сандарды бит бойынша салыстыру үшін қолданылатын арнайы операторлар болып табылады. Олар санның биттерін өңдеу кезінде пайдалы болады, мысалы, солға немесе оңға жылжыту, биттерді инверттеу, және екілік сандарды логикалық түрде біріктіру үшін қолданылады [16]. Дәрістің веб-бетін 14-ші суретте көруге болады.



Сурет 14 7-ші дәрістің веб-беті

Бұл код Python арифметикалық операторларын көрсетеді. «+» операторының мысалы екі санның қосылуын көрсетеді. Арифметиканы қоса алғанда, операторлардың негізгі түрлері келтірілген (мысалы, +, -, \*, /), тағайындау, салыстыру, логика, сәйкестендіру және тиесілілік операторлары. Кестеде қосу (+ ), азайту ( - ), көбейту ( \* ), бөлу ( / ), қалдықты табу ( % ), дәрежеге көтеру ( \* \* ) және бүтін бөлу ( / / ) [16]. 7-ші дәрістің құрылымдық коды 15-шы суретте көруге болады.



Сурет 15 7-ші дәрістің құрылымдық коды

8-ші дәріс: «Python-дағы Тізімдер» тақырыбын қамтиды. Тізімдер - бұл Python бағдарламалау тілінде бірнеше элементтерді бір айнымалыда сақтау үшін қолданылатын деректер түрі. Олар мәліметтерді сақтауға арналған негізгі төрт кірістірілген деректер түрінің бірі болып табылады, ал қалғандары: Кортеж, Жиын және Сөздік. Тізімдерді жасау үшін квадрат жақшалар пайдаланылады, және олардың ішіне әртүрлі элементтерді қосуға болады [17].

Тізімдердің басты ерекшеліктерінің бірі - олардың реттелген, өзгермелі және қайталанатын мәндерді қолдай алатындығы. Элементтердің реттілігі тізімде маңызды: бірінші элементтің индексі әрдайым 0-ге тең, екінші элементтікі - 1, және т.с.с. Бұл реттілік өзгермейді, бірақ тізімнің өзін өзгертуге, яғни жаңа элементтер қосуға немесе бар элементтерді жоюға болады. Бұл тізімдердің өзгермелілік қасиеті деп аталады [17].

Сонымен қатар, тізімдерде бірдей мәндер бірнеше рет қайталануы мүмкін, бұл олардың қайталанатын элементтерді қолдайтынын білдіреді. Бұл ерекшелігі тізімдерді мәліметтерді реттеуде немесе қайталанатын мәндер қажет болған жағдайларда ыңғайлы етеді [17].

Мысалы, бір тізімде «алма», «банан», «шие» сияқты элементтер болуы мүмкін, және олар барлық элементтерді сақтау үшін реттілікті бұзбайды [17]. Дәрістің веб-бетін 16-ші суретте көруге болады.



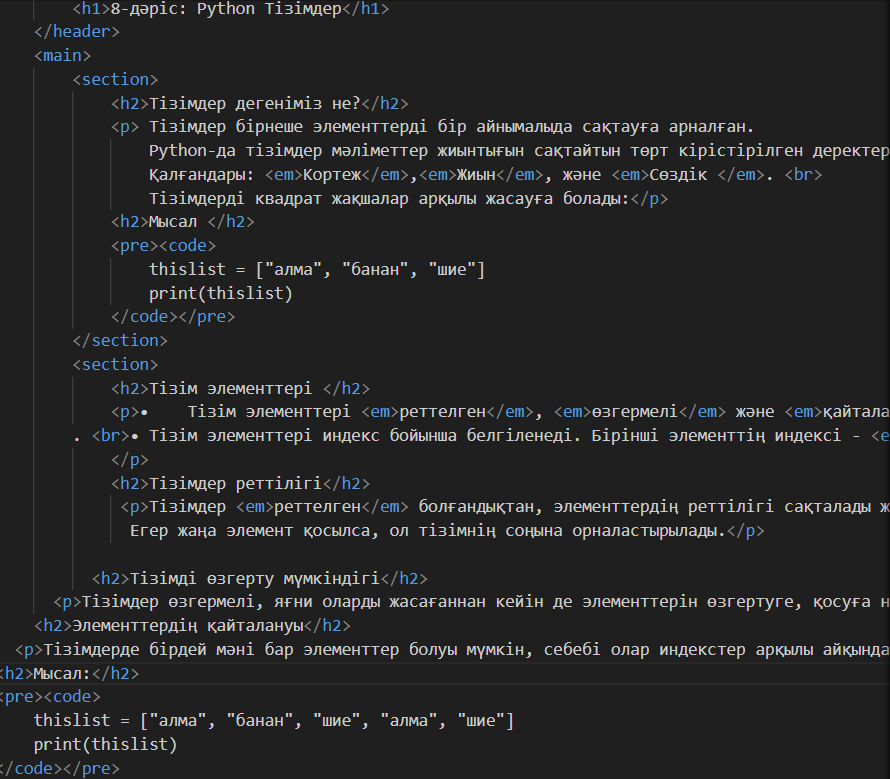
Сурет 16 8-ші дәрістің веб-беті

Python-дағы тізім «[ ]» төртбұрышты жақшалардың көмегімен жасалады, олардың ішінде элементтер үтір арқылы көрсетіледі. Элементтер кез келген деректер түрі болуы мүмкін, мысалы, жолдар, сандар немесе тіпті басқа тізімдер. Тізім жасалғаннан кейін оны болашақта жұмыс істеу үшін айнымалыға тағайындауға болады [17].

Мысалы, сатып алғыңыз келетін сүйікті тағамдарыңыз туралы деректерді сақтау үшін пайдалы болуы мүмкін. Жеке элементтерге олардың индексі бойынша оңай қол жеткізуге болады. Python-да индекстеу нөлден басталады, яғни бірінші элементтің индексі 0, екіншісі-1 және т.б. [17] .

Тізімдер көптеген кіріктірілген функциялар мен әдістердің арқасында деректерді өңдеуді жеңілдетеді және айтарлықтай жылдамдатады. Бұл оларды кез-келген деңгейдегі бағдарламашылар үшін қуатты құрал етеді.

Мысалы, жемістер тізімі үш жолдан тұрады: «алма», «банан» және «шие». Тізімді экранға шығару үшін print () функциясы қолданылады [17]. 8-ші дәрістің құрылымдық кодын 17-ші суретте көруге болады.



Сурет 17 8-ші дәрістің құрылымдық коды

9-шы дәріс: «Python Операторлары : If / Else, While, For». Бағдарламада белгілі бір шарттарға байланысты шешім қабылдау немесе қайталанатын әрекеттерді орындау жиі қажет болады. Python тілінде бұл үшін шартты операторлар (if, elif, else) мен циклдік құрылымдар (while, for) қолданылады. Олар бағдарлама логикасын құруға және процестерді автоматтандыруға мүмкіндік береді.

Шартты операторлар бір немесе бірнеше шартты тексеріп, сәйкесінше белгілі бір блоктағы кодты орындауға мүмкіндік береді. Бұл құрылымдар бағдарламаның әртүрлі сценарийлерде әртүрлі әрекет жасауын қамтамасыз етеді. Python-да шарттар анық әрі оқуға жеңіл синтаксиспен жазылады, ал шегіністердің рөлі – код құрылымының айқын болуын қамтамасыз етеді.

Циклдер - белгілі бір әрекеттерді бірнеше рет қайталау үшін қолданылатын құралдар. Python-да екі негізгі цикл бар: while және for. Біріншісі шарт орындалғанша жұмыс істейді, ал екіншісі итерирленетін объектінің элементтері арқылы өтеді. Екі цикл де қайталанатын тапсырмаларды автоматтандыруда өте тиімді, әрі олармен бірге қолданылатын қосымша операторлар арқылы процесті икемді түрде басқаруға болады. «Python-дағы If/Else операторлары » операторлардың жұмыс принципін түсіндіруге арналған. Бұл операторлар логикалық шарттарды тексеру арқылы белгілі бір әрекетті орындауға немесе орындамауға мүмкіндік береді [18].

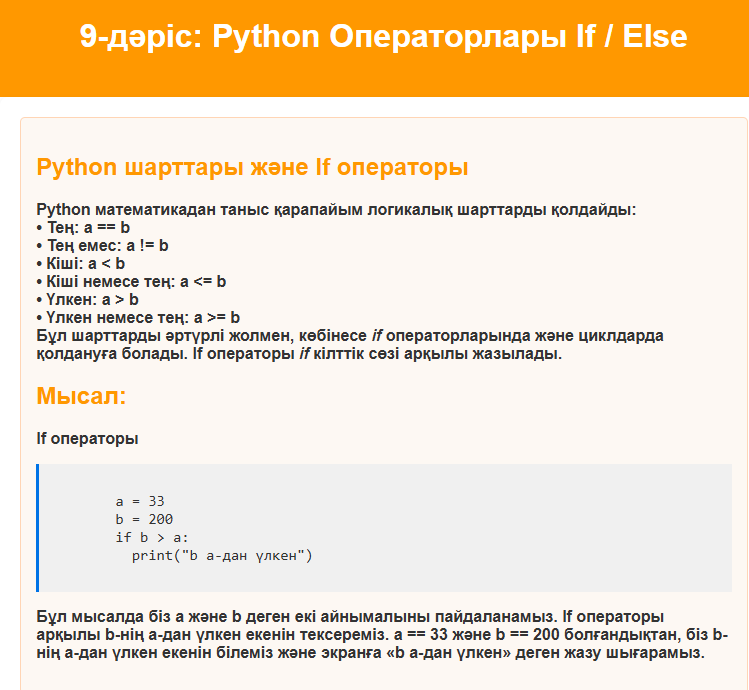
If операторы белгілі бір шарттың орындалуын тексереді. Егер шарт шын (True) болса, онда оператор ішіндегі әрекеттер орындалады. Мысалы, екі айнымалының бір-біріне қатысты мәндерін салыстырып, үлкен не кіші екенін анықтауға болады. Егер тексеру нәтижесінде шарт дұрыс болса, сәйкес нәтиже көрсетіледі [18].

Python-да шегіністер кодтың құрылымын анықтайды. Басқа программалау тілдерінде фигуралық жақшалар қолданылса, Python бұл үшін бос орындар пайдаланады. Егер шегіністер дұрыс болмаса, бағдарлама қате береді [18].

Else операторы If шарты дұрыс болмаған кезде орындалатын әрекеттерді анықтайды. Бұл, мысалы, шарт орындалмаса, басқа әрекетті іске қосу үшін қолданылады [18].

Elif операторы «егер алғашқы шарт дұрыс болмаса, басқа шартты тексер» дегенді білдіреді. Бірнеше шартты тізбектеп тексеру қажет болған жағдайда If, Elif, және Else операторлары бірігіп қолданылады [18].

Егер шарт пен әрекет қысқа болса, оларды бір жолға жазуға болады. Бұл кодты ықшам әрі түсінікті етеді [18]. 9-шы дәрістің веб-бетін 18-ші суретте көруге болады.



Сурет 18 9-шы дәрістің веб-беті

Python-да қарапайым математикалық шарттар пайдаланылады:

* Тең: a == b
* Тең емес: a != b
* Кіші: a < b
* Кіші немесе тең: a <= b
* Үлкен: a > b
* Үлкен немесе тең: a >= b

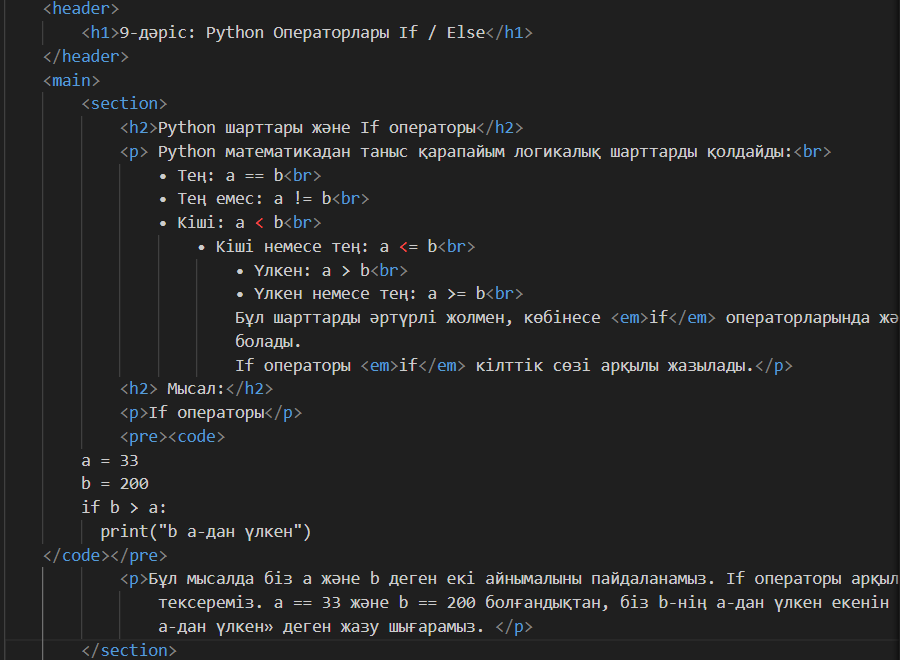
Бұл шарттарды if операторларында және циклдарда қолдануға болады.

Python басқа программалау тілдерінен ерекшеленеді, себебі онда шегініс (жол басындағы бос орындар) өте маңызды. Бұл шегініс кодтың белгілі бір бөлігі қай блокқа жататынын анықтайды [18].

Else операторы if шарты орындалмаған жағдайда орындалатын кодты анықтайды. Python-да elif операторы «егер бастапқы шарт дұрыс болмаса, келесі шартты тексер» дегенді білдіреді[18].

Python-дағы If/Else операторлары бағдарламаларда шешім қабылдау механизмдерін құруға мүмкіндік береді. Олар логикалық шарттарды тексеріп, бағдарламаның орындалу жолын нақтылай алады. Шегіністер мен дұрыс синтаксисті сақтау Python тілінде басты рөл атқарады [18].

1-ші мысалда біз екі айнымалыны (a және b) пайдаланамыз. If операторы арқылы b мәнінің a мәнінен үлкен екенін тексереміз. Егер шарт орындалса (b > a), онда print() функциясы «b a-дан үлкен» деген нәтижені шығарады [18]. 9-шы дәрістің құрылымдық кодын 19-шы суретте көруге болады.



Сурет 19 9-шы дәрістің құрылымдық коды

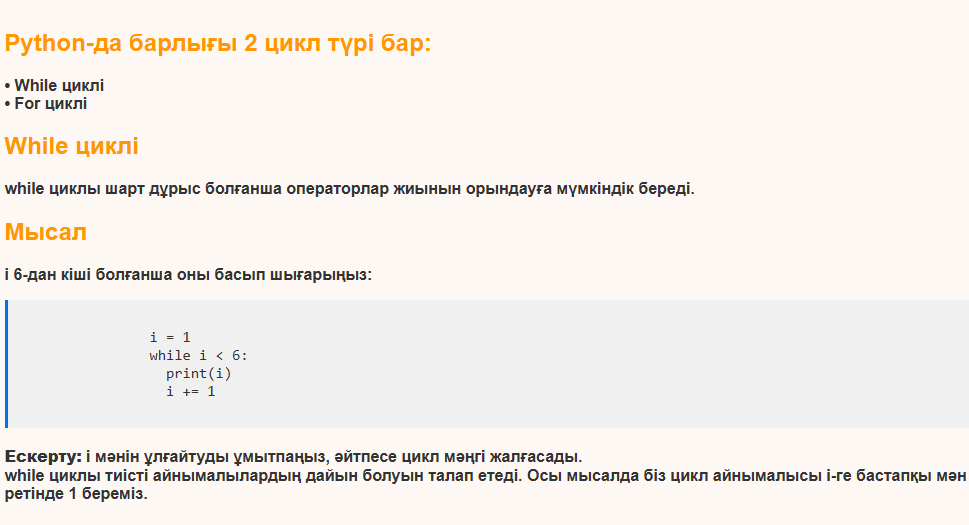
Python бағдарламасында циклдар – бұл белгілі бір шартқа негізделген кодтың қайталануын қамтамасыз ететін негізгі құралдардың бірі [19].

While циклы белгілі бір шарт дұрыс (True) болғанша командаларды қайталайды. Бұл цикл шарт дұрыс болғанда басталып, шарт жалған (False) болғанда тоқтайды. Мысалы, егер айнымалының мәні 6-дан кіші болғанда, осы мәндерді бірінен соң бірін экранға шығару қажет болса, while циклын қолдануға болады. Айнымалының мәнін цикл ішінде өзгерту маңызды, себебі бұл циклдың тоқтауына жауап береді. Егер мән өзгермесе, цикл шексіз қайталанып, бағдарлама дұрыс жұмыс істемейді [19].

Break операторы циклды мерзімінен бұрын тоқтату үшін қолданылады. Бұл шарт дұрыс болғанына қарамастан, циклдан шығып кетуге мүмкіндік береді. Мысалы, егер цикл белгілі бір айнымалының мәні 3-ке тең болған кезде тоқтауы қажет болса, break операторын қолдануға болады [19].

While циклы шарт дұрыс болғанша әрекеттерді қайталауға мүмкіндік береді. Бұл циклдың дұрыс жұмыс істеуі үшін шартқа сәйкес айнымалыны уақтылы өзгертіп отыру керек. Ал break операторы арқылы циклды қалаған кезде тоқтатуға болады, бұл бағдарламаның жұмыс логикасын икемді етеді [19].

While циклы тек шарттарды тексеріп, қайталану процестерін ұйымдастыру үшін ғана емес, сонымен қатар пайдаланушының енгізу деректерімен жұмыс істеу кезінде де жиі қолданылады. Мысалы, белгілі бір шартқа сай келетін мән алынғанша, пайдаланушыдан деректер енгізу талап етілетін бағдарламаларда while циклын пайдалану тиімді. Бұл әдіс шексіз енгізуді немесе нақты шартқа байланысты процестерді орындауға мүмкіндік береді, осылайша бағдарламаларды икемді әрі тиімді етеді [19]. Дәрістің веб-бетін 20-шы суретте көруге болады.



Сурет 20 дәрістің веб-беті

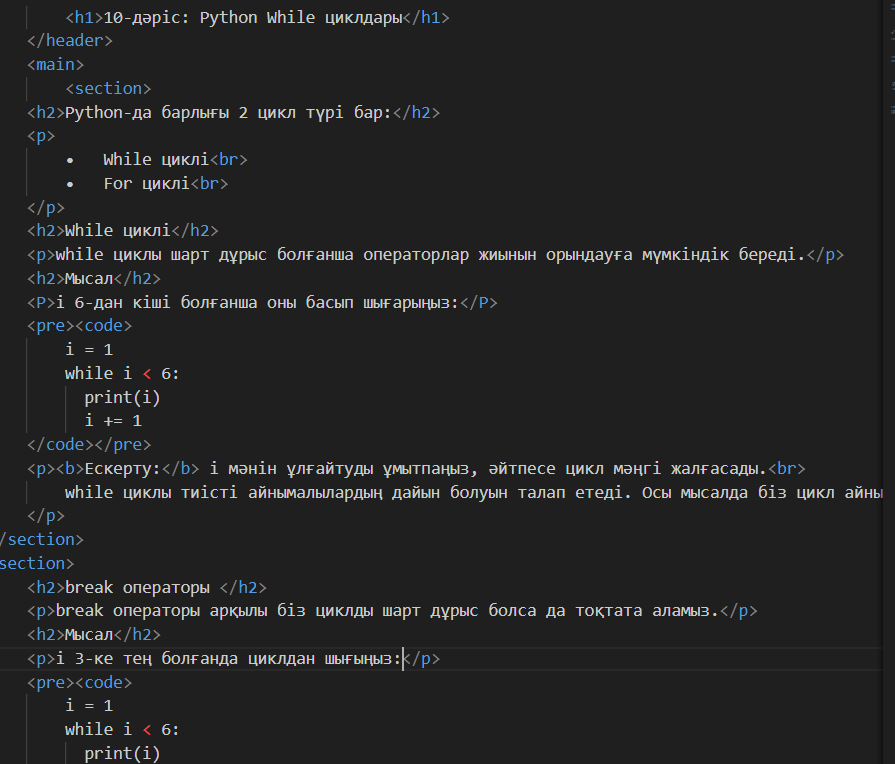
Циклдық процестің негізі – While циклы:

* i = 1: Бұл жерде айнымалы i бастапқы мән ретінде 1 деп беріледі. Бұл циклдың жұмысын бастау үшін қажет бастапқы шартты мән [19].
* while i < 6: Бұл циклдың негізгі шарты. Айнымалы i мәні 6-дан кіші болғанша, циклдың ішіндегі код орындалады. Шарт орындалмаса, цикл аяқталады [19].
* print(i): Әр қайталанған сайын i мәні экранға басылады. Бұл нәтиже циклдың жұмысын көру үшін қолданылады.
* i += 1: Әр итерациядан кейін i мәні 1-ге ұлғайтылады. Бұл қадам циклды шексіз қайталанудан сақтайды, себебі i біртіндеп өсіп, шарттың орындалмауына әкеледі.

Break операторы бар While циклы:

* if i == 3: Бұл шарт i мәні 3-ке тең болған кезде орындалады.
* break: Бұл оператор циклды дереу тоқтатады. Циклдың қалған бөлігі орындалмайды, және басқару циклдан кейінгі кодқа өтеді [19].

Бұл код циклды шарт дұрыс болса да i == 3 болған сәтте тоқтатады. Бұл процесс белгілі бір шектеу қою немесе циклды алдын ала жоспарланған жерде тоқтату үшін қолданылады [19]. Дәрістің құрылымдық кодын 21-ші суретте көруге болады.

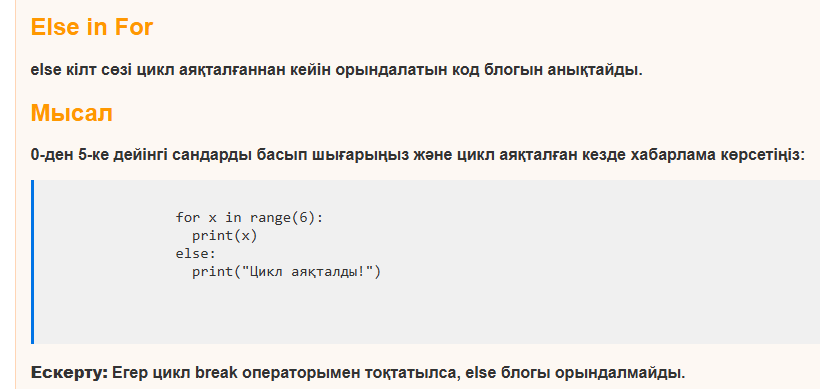


Сурет 21 дәрістің құрылымдық коды

For циклы – бұл Python-да итерирленетін объектілердің әрбір элементіне біртіндеп қол жеткізуге мүмкіндік беретін құрал. Итерирленетін объектілерге тізімдер, жолдар, жиындар, сөздіктер және басқа да коллекциялар жатады. For циклының негізгі артықшылығы – оның жеңіл және қарапайым синтаксисінде, соның арқасында қайталауларды жазу оңай әрі түсінікті болады [20].

Python-дағы for циклы индекстерге немесе басқа айнымалыларға тәуелді болмайды. Оның орнына, ол әрбір элементті ретімен алып, оны цикл ішінде қолдануға мүмкіндік береді. Бұл тәсіл кодты ықшамдайды және индекстерді қолмен басқару қажеттілігін жояды. Мысалы, тізімдегі әрбір элементті немесе сөздегі әрбір символды шығару үшін for циклын оңай қолдануға болады [20].

Сонымен қатар, for циклы тек тізімдермен немесе жолдармен шектелмейді. Ол басқа коллекциялар, мысалы, жиындар немесе сөздіктермен де жұмыс істей алады. Жиындарда ол ретсіз элементтерді бір-бірден алады, ал сөздіктерде әдетте әрбір кілт бойынша итерация жасалады. For циклын қолданудың тағы бір артықшылығы – оны басқа басқару операторларымен, мысалы, break, continue және else операторларымен біріктіру мүмкіндігі. Бұл операторлар циклдың орындалу процесін басқаруға және әртүрлі шарттарға байланысты қайталануларды тоқтатуға немесе өткізіп жіберуге мүмкіндік береді. Мысалы, егер белгілі бір шарт орындалса, break арқылы циклды толық тоқтату, ал continue арқылы тек ағымдағы итерацияны өткізіп жіберу мүмкіндігі бар [20]. Дәрістің веб-бетін 22-ші суретте көруге болады.



Сурет 22 дәрістің веб-беті

Дәрістің бірінші мысалында for циклы арқылы тізімдегі әрбір элемент бірінен кейін бірі алынып, экранға шығарылады [20].

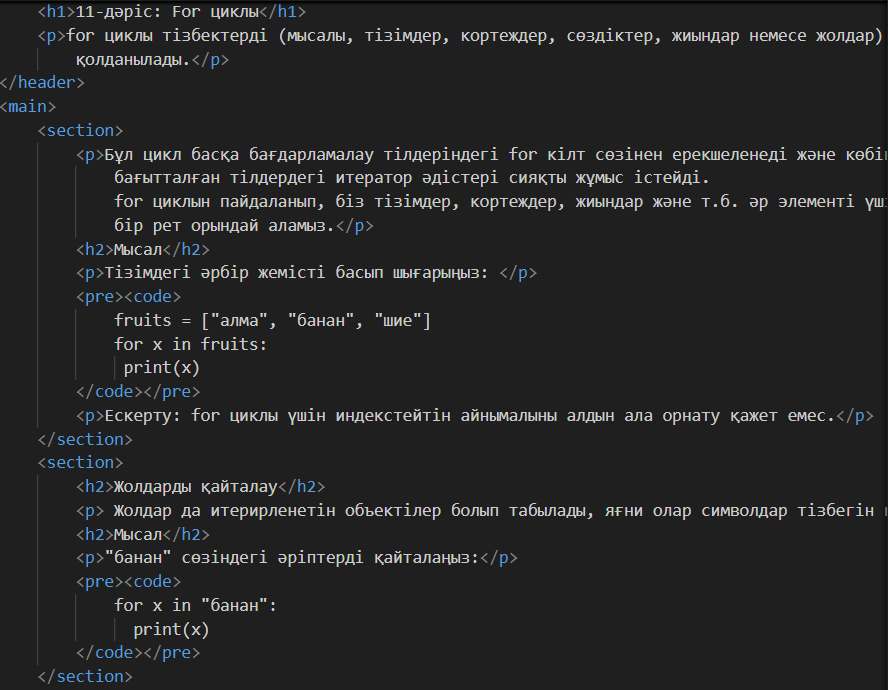
* fruits = [«алма», «банан», «шие»]: Бұл тізім үш элементтен тұрады: «алма», «банан», «шие».
* for x in fruits:: Цикл тізімдегі әрбір элементті бір-бірден x айнымалысына жүктейді.
* print(x): Әр итерацияда x мәні экранға басылады.

Дәрістің екінші мысалында жолдар қайталанатын объектілер табылады, сондықтан for циклы арқылы әрбір символды өңдеуге болады [20].

* for x in «банан»: Цикл «банан» сөзі арқылы өтеді, әрбір итерацияда келесі символды x айнымалысына жүктейді.
* print(x): Әр итерацияда x мәні экранға басылады.

Жолдармен жұмыс: For циклы әрбір символды жеке-жеке қарастыруға мүмкіндік береді. Мысалы, сөздің әрбір әрпін экранға шығару немесе әрбір символ бойынша өңдеу жүргізуге болады [20].

Тізімдермен жұмыс: Тізімдегі барлық элементтерді қайталап шығу оңай және тиімді. Бұл ретте тізім элементтерінің реттілігі автоматты түрде сақталады. Python-дағы for циклы – бұл қайталанатын операцияларды оңай орындауға арналған күшті және икемді құрал. Оның синтаксисі түсінікті, әрі ол әртүрлі мәліметтер құрылымдарымен жақсы жұмыс істейді, бұл бағдарламалаушылар үшін процестерді оңтайландырудың таптырмас құралы [20]. Дәрістің құрылымдық кодын 23-ші суретте көруге болады.



Сурет 23 дәрістің құрылымдық коды

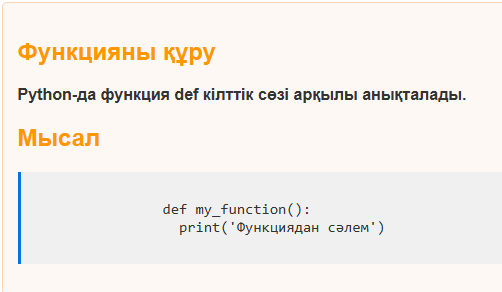
10 - шы дәріс: «Python Функциялары,Лямбда,Массивтер, Мұрагерлік» тақырыбын қамтиды. Бағдарламалау барысында кодты құрылымды, ықшам және қайталанбайтындай етіп жазу өте маңызды. Бұл мақсатта Python тілінде функциялар, лямбда өрнектер, массивтер және мұрагерлік секілді құралдар кеңінен қолданылады. Олар бағдарламаның ауқымын кеңейтіп, күрделі логикаларды тиімді ұйымдастыруға мүмкіндік береді.

Функциялар – бұл кодты ұйымдастыру мен қайта қолдану үшін қолданылатын негізгі құрал. Python-дағы функциялар – бұл белгілі бір тапсырманы орындау үшін қажет болатын кодтың бөлігі. Олар тек шақырылған кезде орындалады және бағдарламаның кез келген жерінде көп мәрте пайдалануға мүмкіндік береді. Бұл тәсіл кодты ықшам, түсінікті және құрылымдалған етеді [21].

Функциялар арқылы біз мәліметтерді өңдей аламыз. Параметрлер (аргументтер) – бұл функцияға қажетті бастапқы мәліметтерді беру үшін қолданылатын құрал. Аргументтердің көмегімен функцияны икемді етуге және әртүрлі мәліметтермен жұмыс істеуге болады. Функция шақырылғанда оған сәйкес аргументтер беріледі, бұл тапсырманы орындау кезінде мәліметтерді өңдеуді жеңілдетеді [21].

Python-да функцияларды анықтау def кілттік сөзімен басталады, бұл бағдарламалаушыға жаңа функцияны сипаттауға мүмкіндік береді. Функция атауынан кейін дөңгелек жақшалар ішінде аргументтерді көрсетуге болады. Ал функцияның ішінде қажетті кодты жазуға болады. Функцияны қолдану үшін оның атауын жазып, дөңгелек жақшаларды қолдану жеткілікті [21].

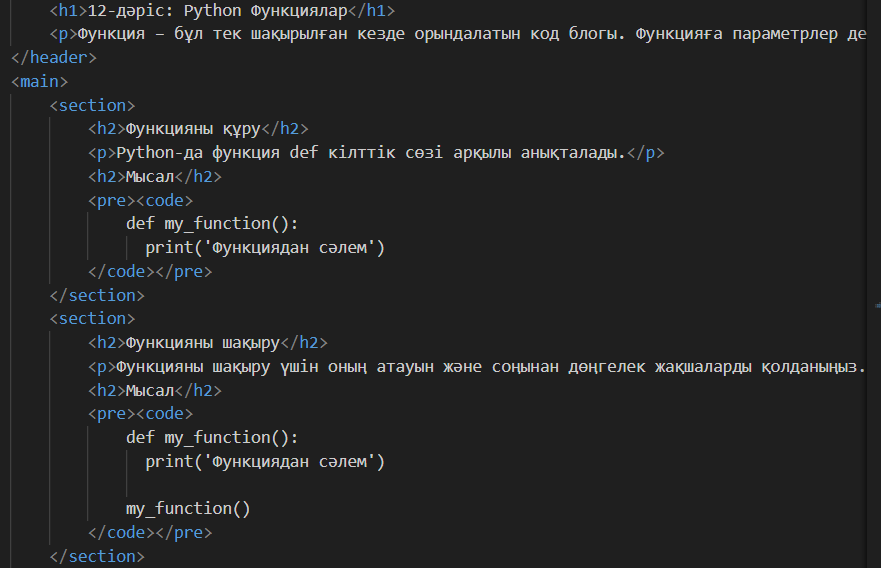
Функциялар кодты құрылымдауға ғана емес, сонымен қатар қайта қолдануға мүмкіндік береді. Бұл әсіресе үлкен бағдарламаларда кодтың қайталануын азайтып, бағдарламаны оңай оқылатын және басқарылатын етеді. Функцияны дұрыс ұйымдастыру бағдарламаны тиімдірек етеді және қателерді азайтады [21]. 10 - шы дәрістің веб-бетін 24-ші суретте көруге болады.



Сурет 24 дәрістің веб-беті

* def my\_function():бұл функцияны анықтау үшін қолданылатын Python-дағы кілт сөз.
* my\_function: өзіңіз орнатқан функцияның атауы. Сіз кез-келген атауды таңдай аласыз, бірақ оның функцияның не істейтінін көрсетуі маңызды [21].
* (): Функция атауынан кейінгі жақшалар. Егер функция параметрлерді (аргументтерді) қабылдаса, олар осы жақшада көрсетіледі. Бұл жағдайда жақшалар бос, сондықтан функция дәлелдерді қабылдамайды.
* :: Қос нүкте келесі функцияға қатысты код блогы бар екенін көрсетеді. Python-да код блогы шегініспен анықталады.
* print ('функциядан сәлем') мәтінді немесе деректерді экранға шығаратын кірістірілген Python мүмкіндігі.
* 'Функциядан сәлем': бұл бір тырнақшаға салынған жол. Функция шақырылған кезде ол экранға шығады.
* my\_function: бұл біз бұрын анықтаған функцияға қоңырау .

Функциялар - бағдарламалаудың негізгі құралдарының бірі, олар кодты құрылымды және қайта пайдалануға ыңғайлы етеді. Python тілінде функциялар белгілі бір әрекеттерді орындайтын код блоктары ретінде құрылады және оларды бірнеше рет, әртүрлі мәліметтермен қолдануға болады. Бұл тәсіл кодты қысқартып, оқуға жеңілдетеді әрі логиканы бөлек-бөлек бөліктерге ұйымдастыруға мүмкіндік береді. Функциялар арқылы күрделі бағдарламаларды модульдерге бөліп, әрқайсысын жеке-жеке тестілеу және дамыту оңайға түседі. Дәрістің құрылымдық кодын 25-ші суретте көруге болады.



Сурет 25 дәрістің құрылымдық коды

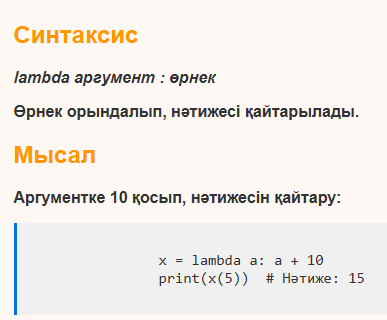
Ламбда функциясы-бұл Lambda кілт сөзімен анықталатын шағын, анонимді функция. Кәдімгі функциялардан айырмашылығы, ламбда функциялары атауды қажет етпейді және қарапайым операцияларды орындауға арналған [22].

Негізгі ерекшелігі-ламбда функциясы кез-келген аргументтерді қабылдайды, бірақ нәтижесі автоматты түрде қайтарылатын бір ғана өрнекті қамтиды. Лямбда функциясының синтаксисі келесідей: lambda аргументтер: өрнек [22].

Аргументтер қос нүктенің сол жағындағы функцияға беріледі, ал оның оң жағындағы өрнек не орындалатынын анықтайды. Мұндай функциялар көбінесеактам кодты жазу үшін қолданылады, әсіресе толыққанды функция жасаудың қажеті жоқ жағдайларда. Лямбда функциялары бірнеше аргументтерді қолдануды қолдайды [22].

Лямбда-функциялар - Python тілінде қолданушының арнайы бір функция жазбай-ақ, қысқаша және бір жолдық функцияларды жасауға мүмкіндік береді. Бұл функциялар көбінесе қарапайым есептер үшін қолданылады, мысалы, элементтерді фильтрациялау немесе өзгертулер енгізу кезінде. Лямбда-функциялар синтаксисі қарапайым және оларды қолдану кодыңызды қысқартуға көмектеседі. Лямбда функциясы lambda кілт сөзінен басталып, параметрлер мен оларды өңдеу нәтижесін бір қатарда анықтайды.

Мысалы, аргумент ретінде берілген екі санды көбейтуге немесе бірден үш санды қосуға болады. Мұндай функциялар деректерді сұрыптау немесе сүзу әдістері сияқты басқа функцияларда қолдануға пайдалы [22]. Дәрістің веб-бетін 26-шы суретте көруге болады.



Сурет 26 дәрістің веб-беті

Қарапайым ламбда функциясын құру:

* + lambda a: a + 10 - бір a аргументін қабылдайтын және A + 10 нәтижесін қайтаратын функция жасайды [22].
  + X айнымалысына ламбда функциясы тағайындалады. X(5) қоңырауы 5 мәнін а аргументіне жіберіп, 15 нәтижесін қайтарады.

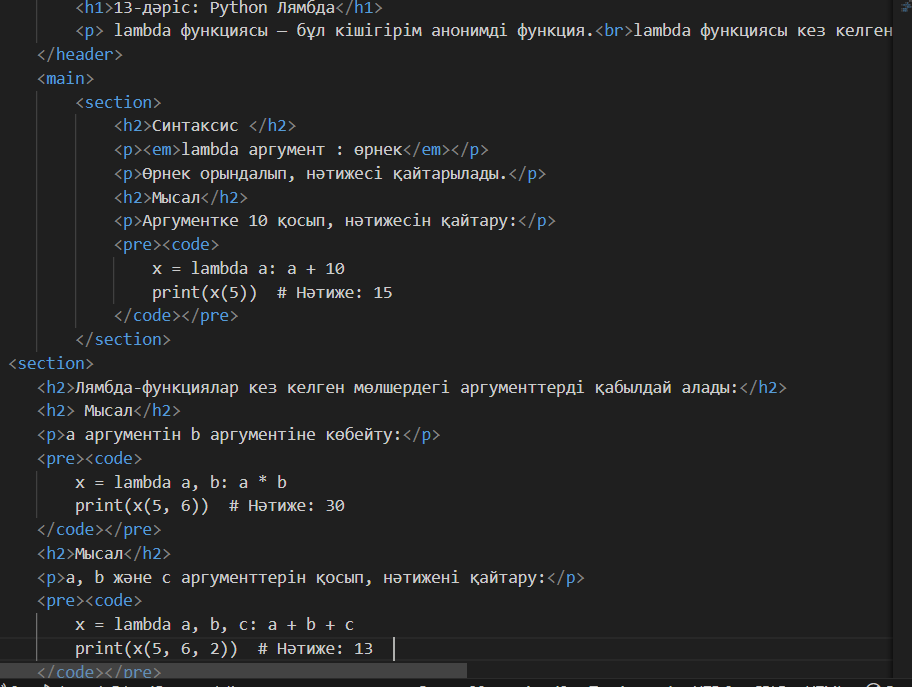
Екі аргументі бар ламбда функциясы:

* lambda a, b: A \* b - екі a және b аргументтерін қабылдайтын және олардың өнімін қайтаратын функция.
* X(5, 6) қоңырауы 5 және 6 мәндерін аргументтерге береді, нәтиже 5 \* 6 = 30 болады.

Үш аргументі бар ламбда функциясы:

* lambda a, b, C: a + b + c - үш a, b, C аргументтерін қабылдайды және олардың қосындысын қайтарады.
* X(5, 6, 2) функциясын шақырған кезде нәтиже сандардың қосындысы болады: 5 + 6 + 2 = 13.

Лямбда функциялары Python тілінде қысқа және ыңғайлы түрде функциялар жазуға мүмкіндік береді. Қарапайым лямбда функциясы бір аргументті қабылдап, оны өңдей алады, мысалы, lambda a: a + 10, онда аргументке 10 қосылады. Екі немесе үш аргументі бар лямбда функциялары да оңай құрылып, сәйкесінше арифметикалық операциялар орындай алады. Мысалы, екі аргументті лямбда функциясы lambda a, b: a \* b екі мәннің көбейтіндісін есептейді, ал үш аргументті функция lambda a, b, c: a + b + c қосындыны қайтарады. Лямбда функцияларының қолданылуы бағдарламаның қысқалығын арттырып, кодты оқуды жеңілдетеді, әсіресе қарапайым есептерді орындағанда. Дәрістің құрылымдық кодын 27-ші суретте көруге болады.



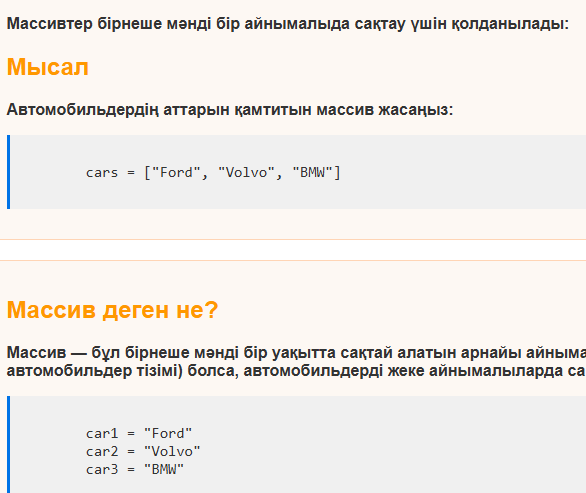
Сурет 27 дәрістің құрылымдық коды

Массивтер-бұл бірнеше мәндерді бір айнымалы атаумен сақтауға мүмкіндік беретін мәліметтер құрылымы. Мысалы, автомобильдердің атаулары сияқты әр элемент үшін жеке айнымалылар құрудың орнына, массивтер оларды бір жерде біріктіруге мүмкіндік береді. Python-да массивтермен жұмыс істеу үшін стандартты тізімдер жиі қолданылады, дегенмен күрделі операциялар үшін NumPy кітапханасы ұсынылады [23].

Массивтер автомобиль тізімдері сияқты көптеген элементтермен жұмыс істеу қажет болған жағдайда ыңғайлы. Жүздеген жеке айнымалыларды құрудың орнына әр элементке индекс тағайындалған массивті қолдануға болады. Бұл индекстер массивтің жеке элементтеріне қол жеткізуге және олармен операцияларды орындауға мүмкіндік береді [23].

Массивтерді пайдалану деректерді басқаруды жеңілдетеді, әсіресе олар көп болса. Мысалы, массивпен белгілі бір элементті табу, барлық мәндерден өту немесе деректерді өзгерту оңай. Бұл массивтерді Python-да деректерді өңдеудің қуатты құралына айналдырады [23].

Массивтер үлкен көлемдегі деректермен жұмыс істеуде шешуші рөл атқарады, өйткені олар индекс бойынша қол жетімді бір құрылымда көптеген мәндерді сақтауға мүмкіндік береді. Бұл элементтерді сұрыптау, сүзу және іздеу процесін жеңілдетеді. Мысалы, егер сізде автомобиль атаулары бар массив болса, сіз олардың кез-келгеніне оңай қол жеткізе аласыз немесе оның мәнін жаңа айнымалылар жасамай өзгерте аласыз. Сонымен қатар, массивтерді күрделі математикалық амалдарды орындау үшін пайдалануға болады, әсіресе NumPy сияқты кітапханалармен біріктіріліп, оларды сандық деректермен және көп өлшемді құрылымдармен жұмыс істеу үшін қажет етеді [23]. Дәрістің веб-бетін 28-ші суретте көруге болады.



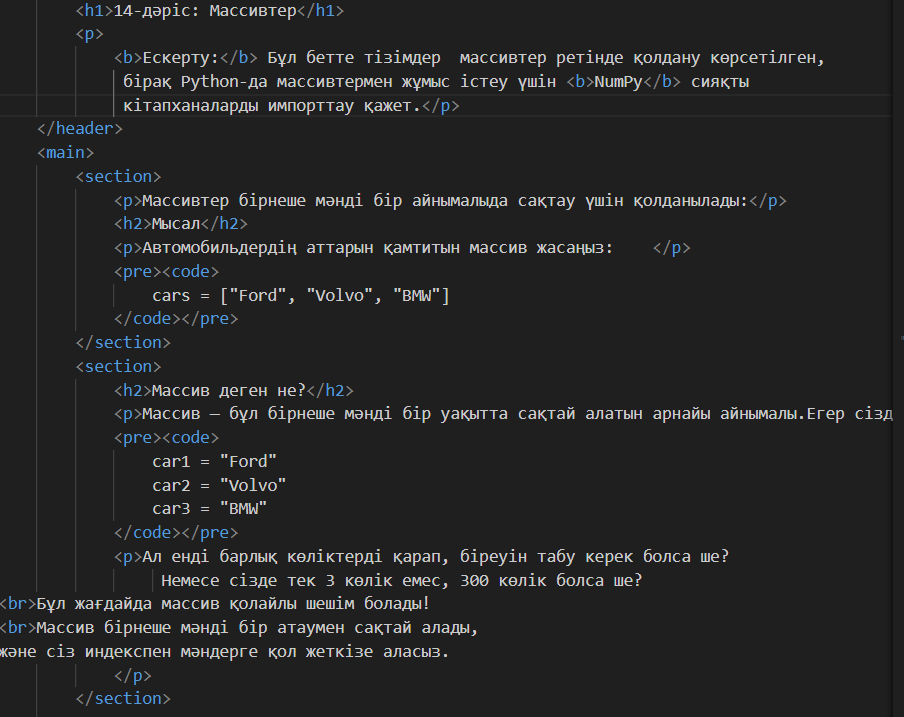
Сурет 28 дәрістің веб-беті

Python-да индекстер оң немесе теріс болуы мүмкін. Оң индекстер массивтің басынан (нөлден), ал теріс индекстер соңынан есептеледі. Теріс индекстер ыңғайлы, егер сіз массивтің соңында орналасқан элементке оның ұзындығын білмей тез жүгінуіңіз керек болса [23].

Бөлімдер бір уақытта бірнеше массив элементтерін алуға мүмкіндік береді. Массивтің белгілі бір бөлігін бөлектеу үшін бастапқы және соңғы индекстерді көрсетуге болады. Егер бастапқы индекс көрсетілмесе, кесу массивтің басынан басталады, ал егер соңғы индекс көрсетілмесе, кесу соңына дейін жалғасады [23].

Массив элементтерін индекс бойынша оларға қол жеткізу арқылы өзгертуге болады. Бұл ескі мәнді жаңасымен ауыстыруға мүмкіндік береді. Бұл мінез-құлық массивтерді өзгеретін деректермен жұмыс істеуге икемді етеді [23].

Кейде сіз массив арқылы өтіп, оның индексін пайдаланып әр элементпен әрекеттерді орындауыңыз керек. Индекстер Итерация процесін дәл бақылауға мүмкіндік береді.деректерді бір атпен сақтауға және индекстер арқылы басқаруға мүмкіндік береді. Массивтің әрбір элементіне оның индексі арқылы қол жеткізуге болады. Массив элементін оған индекс арқылы қол жеткізу арқылы өзгертуге болады. For циклі массивтің барлық элементтерін сұрыптау үшін қолданылады. Массивте қанша элемент бар екенін білу үшін len () функциясы қолданылады. Элементті массивке қосу үшін append () әдісін қолдануға болады. Элементті жою үшін жою () әдісі қолданылады [23]. Дәрістің құрылымдық кодын 29-шы суретте көруге болады.

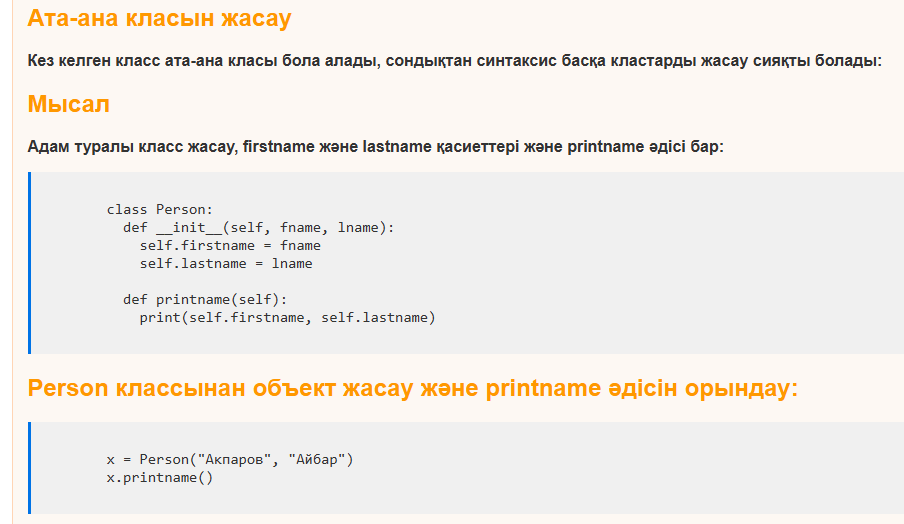


Сурет 29 14-ші дәрістің құрылымдық коды

Мұрагерлік-бұл бір сыныпқа басқа сыныптың қасиеттері мен әдістерін алуға мүмкіндік беретін объектіге бағытталған бағдарламалау механизмі. Бұл кодтың қайталануын болдырмауға көмектеседі және оның оқылуын жақсартады. Мұрагерліктің көмегімен олардың функционалдығын қосу немесе өзгерту арқылы бұрыннан бар элементтерді қолданатын жаңа сыныптар құруға болады. Бұл бағдарламаның икемділігін арттыруға және оны қолдауды жеңілдетуге мүмкіндік береді [24].

Мұрагерліктің негізгі артықшылықтарына мыналар жатады: ата-аналық класс кодын қайта пайдалану мүмкіндігі және бірегей функционалдылықты қосу арқылы жаңа сыныптарды құруды жеңілдету. Ата-ана сыныбы (немесе базалық сынып) Негізгі сипаттамалар мен әдістерді ұсынады, содан кейін оларды балалар сыныптарында қолдануға немесе өзгертуге болады [24].

Бала сыныбы ата-аналық класс әдістерінің мінез-құлқын өзгерте немесе кеңейте алады, бұл нақты жағдайларға бейімделу немесе бейімделу қажет болған кезде пайдалы. Python ата-аналық класс әдістеріне сілтеме жасау үшін `Super ()`функциясын қолданады. Бұл ата-аналық сыныптың әдістерін қайта анықтауға ғана емес, сонымен қатар кейбір функционалдылықты сақтау үшін оларды шақыруға мүмкіндік береді, мысалы, ескісін жоғалтпай әдістерге жаңа логика қосу кезінде маңызды [24]. Дәрістің веб-бетін 30-шы суретте көруге болады.



Сурет 30 дәрістің веб-беті

Кодта Person классы жасалған, оның конструкторы екі параметрді қабылдайды - fname (ат) және lname (тегі). Әдіс printname() адамның атын және тегін экранға шығарады [24].

class Person - жол person деп аталатын жаңа класс жасайды. Python-да класс кілт сөзі жаңа сыныпты анықтау үшін қолданылады. Сыныпта осы сыныптың объектілерін сипаттайтын әдістер (функциялар) және атрибуттар (айнымалылар) болуы мүмкін [24].

def \_\_init\_\_(self, fname, lname) - person класының конструкторы әдісі. \_\_Init \_ \_ () әдісі осы сыныптың жаңа объектісі жасалған кезде автоматты түрде шақырылады. Ол нысан атрибуттарын инициализациялайды.

Self - бұл жасалған нысанның өзіне сілтеме. Бұл параметр міндетті болып табылады және объектінің атрибуттары мен әдістеріне қол жеткізу үшін қолданылады [24].

fname және lname - бұл объект атрибуттарына мәндерді тағайындау үшін қолданылатын конструктор параметрлері (бұл жағдайда аты мен тегі).

self.firstname = fname - Бұл жол объектінің firstname атрибутының мәніне fname параметрінде берілген мәнді тағайындайды. Осылайша, жеке класс объектісінде адамның атын сақтайтын firstname атрибуты болады [24].

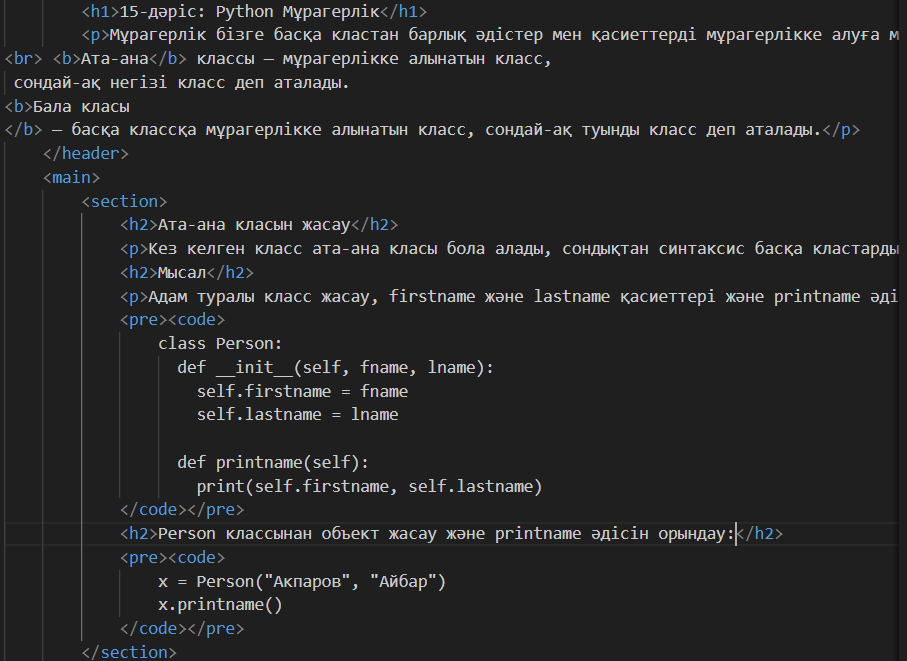
self.lastname = lname - Алдыңғы жолға ұқсас, бұл жол объектінің lastname атрибутының мәніне lname параметрінде берілген мәнді тағайындайды. Lastname атрибуты адамның тегін сақтайды [24].

def printname(self) - аты мен тегін басып шығаратын әдіс. Әдіс def кілт сөзін қолдану арқылы анықталады және конструктор сияқты, әдістің бірінші параметрі нысан атрибуттарына қол жеткізуге мүмкіндік беретін self болып табылады [24].

print(self.firstname, self.lastname) - бұл әдіс экранда аты мен тегін көрсетеді. self.firstname және self.lastname-конструкторда берілген нысан атрибуттары. Print () әдісі бос орынмен бөлінген осы атрибуттардың мәндерін шығарады [24].

x = Person («Акпаров», «Айбар») - Мұнда жаңа person класс нысаны жасалады. Дәлел ретінде «Акпаров» (fname үшін) және «Айбар» (lname үшін) жолдары беріледі. Бұл мәндер x нысаны үшін firstname және lastname атрибуттарына тағайындалады [24].

x.printname() - Бұл әдіс шақырылған кезде x нысаны үшін firstname және lastname атрибуттарының мәндерін экранға шығарады, яғни «Акпаров Айбар». Дәрістің құрылымдық кодын 31-ші суретте көруге болады.



Сурет 31 дәрістің құрылымдық коды

Python тіліндегі бағдарламалаудың негізгі тақырыптары, соның ішінде айнымалылармен, операторлармен, циклдармен, функциялармен және массивтермен жұмыс істеу және мұрагерліктің негізгі принциптері қарастырылды. Бұл дәрістер Python -ды игеруге және сол тілде бағдарламалар жасауға қажетті негізгі білімді қалыптастырады.

Python тілінде функциялар, лямбда функциялары, массивтер және мұрагерлік концепциялары бағдарламалау тілінің негізін құрайды. Функциялар кодты қайта қолдануға мүмкіндік береді, ал лямбда функциялары қарапайым операцияларды қысқа түрде орындауға қолайлы. Массивтер деректерді сақтаудың және өңдеудің тиімді құралдары болып табылады, ал мұрагерлік объектіге негізделген бағдарламалауда қолданылатын маңызды механизм болып саналады. Осы барлық тақырыптар бағдарламашының кодты тиімді және құрылымдық түрде жазуына мүмкіндік береді, сондай-ақ бағдарламаның оқылуын жеңілдетеді.

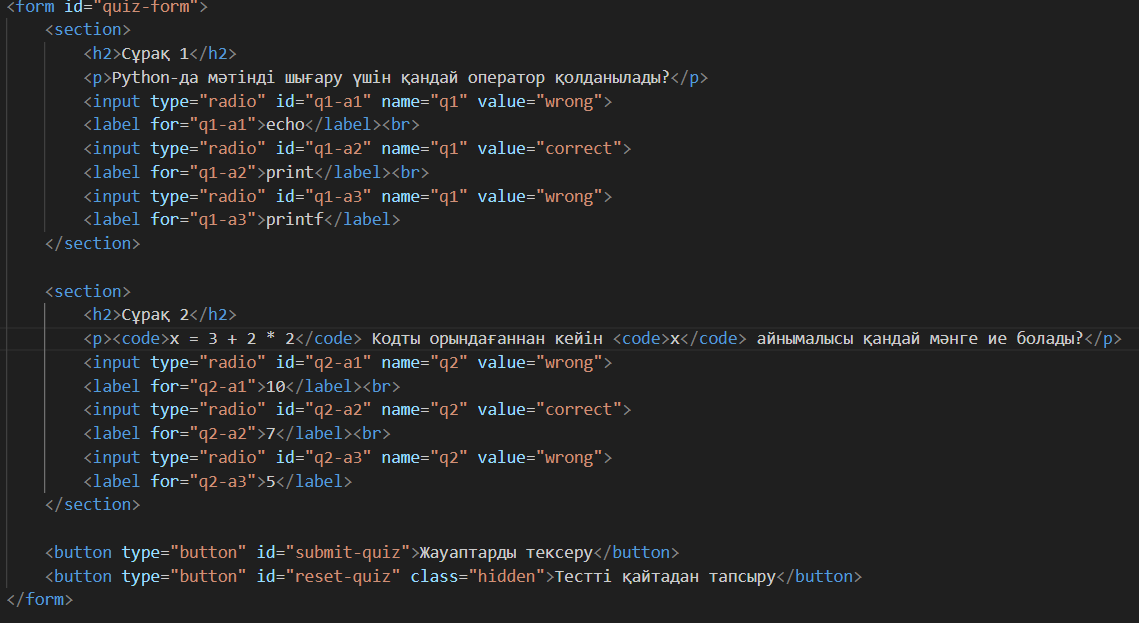
# 1.5.2 Тесттер құру

<form id=" quiz-form " > - JavaScript арқылы оңай сәндеуге немесе басқаруға болатын бірегей quiz-form идентификаторы бар пішін жасалады.

<section> - құрылымның ұйымшыл болуына көмектесетін пішін ішіндегі сұрақ бөлімін бөлектеу үшін қолданылады.

<h2> сұрақ 1</h2> - «сұрақ 1» нөмірі бар бірінші сұрақтың тақырыбы, бұл қазақ тілінде «сұрақ 1» дегенді білдіреді.

Тестке арналған үлгілер 32-41 суреттерде бейнеленеді.



Сурет 32 1-ші тест құрылымының коды

<p>Python тілінде "Сәлем" деп қалай шығарады? </ p> - Бұл Python-да "Сәлем" жолын қалай шығару керектігін сипаттайтын сұрақ мәтіні.

<input type= "radio" id= "q1 - a1" name="q1" value="correct">-бұл q1-a1 бірегей идентификаторы және оның дұрыс жауап екенін көрсететін correct мәні бар бірінші радио түймесі. Name="q1" атрибуты барлық радио түймелерді бір топқа топтастырады, осылайша олардың біреуін ғана таңдауға болады.

<label for= "q1 - a1"> print("Сәлем")</label> - Python-да жолды шығарудың дұрыс әдісін көрсететін "Print('Сәлем')" мәтіні бар бірінші радио батырма үшін белгі жасалады.

Келесі үш жол қате жауаптары бар ұқсас радио түймелерін жасайды:

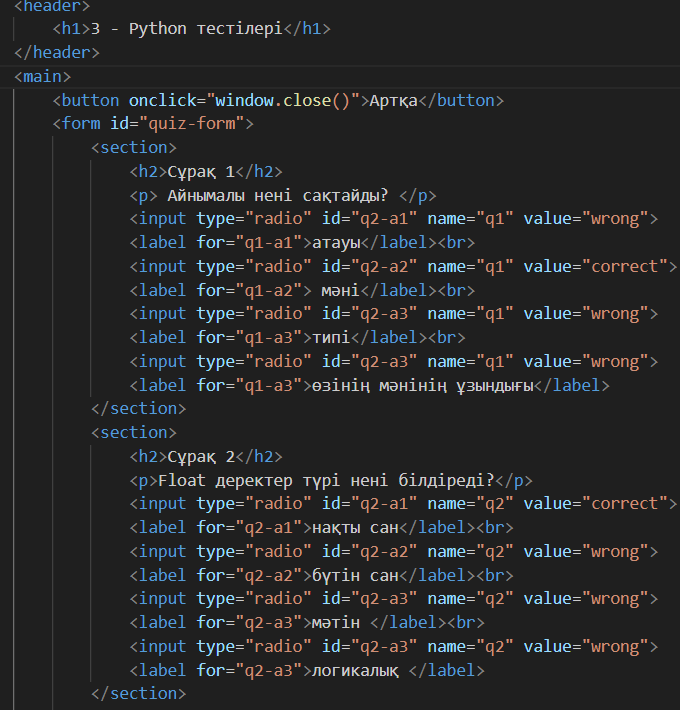
* p ("Сәлем") - Python-дағы синтаксистік қате, өйткені мұндай функция жоқ.
* echo ("Сәлем"): - Python-да шығару үшін дұрыс емес синтаксис.
* echo "Сәлем" - бұл мүмкін емес нұсқа, өйткені Python шығару үшін print() функциясын қолданады.

Әрбір жауап опциясы тиісті түймелерге байланған белгімен (<label>) бірге жүреді. 1-ші тест Python негіздерін, оның ерекшеліктері мен артықшылықтары сияқты қамтитын 10 сұрақтан тұрады. Сұрақтар Python синтаксисі, негізгі операциялар және деректер түрлері туралы білімді тексереді. Бұл тест бағдарламалау тілі мен оның әртүрлі салаларда қолданылуы туралы алғашқы түсініктерді нығайтуға көмектеседі.



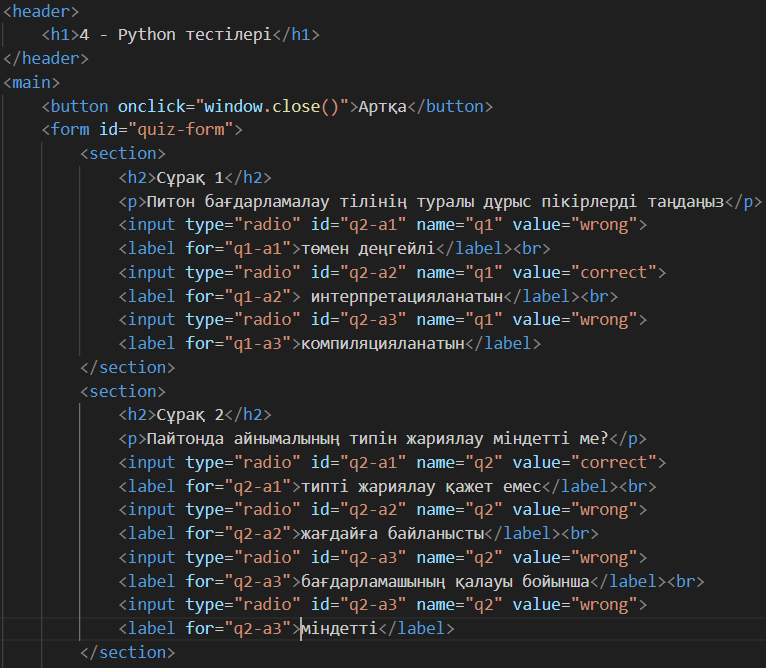
Сурет 33 2-ші тест құрылымының коды

2-ші тест сұрақтары айнымалылармен жұмыс істеудің негізгі принциптеріне, оларды анықтауға, сондай-ақ Python-дағы әртүрлі деректер түрлеріне қатысты. Жолдар, сандар, логикалық мәндер және олардың түрлендірулері сияқты деректер түрлеріне ерекше назар аударылады. Тест сұрақтары Python-да әртүрлі деректер түрлерімен қалай тиімді жұмыс істеу керектігі туралы білімді нығайтуға көмектеседі.



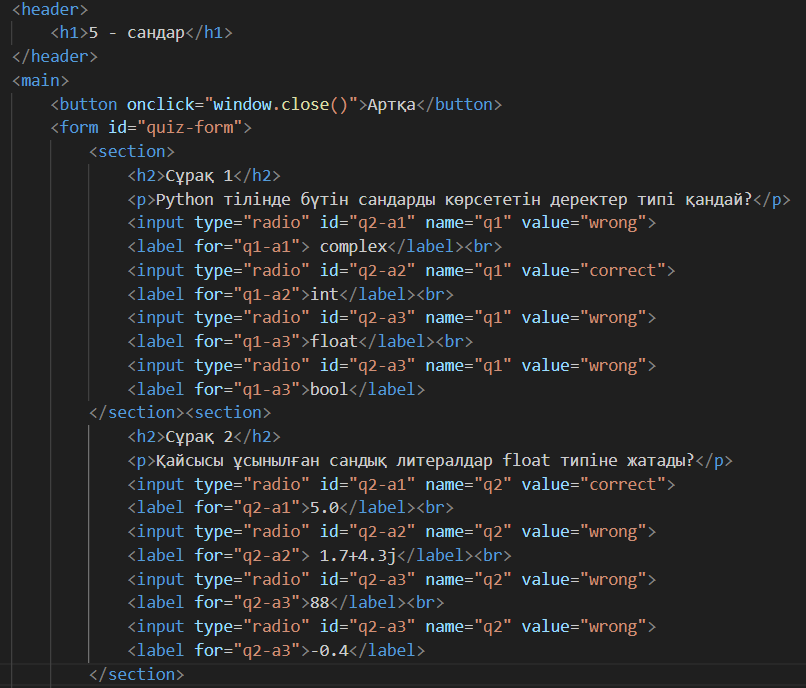
Сурет 34 3-ші тест құрылымының коды

3-ші тест сұрақтары Python-дағы сандар операцияларының барлық түрлеріне, соның ішінде арифметикалық амалдарға,бүтін және өзгермелі нүктелік сандарға қатысты. Сондай-ақ, тест операторлардың басымдығы және сандық типтермен жұмыс істеу ерекшеліктері туралы сұрақтарды қамтиды. Сұрақтар Python-да сандармен жұмыс істеу негіздерінің қаншалықты жақсы меңгерілгенін тексеруге көмектеседі.



Сурет 35 4-ші тест құрылымының коды

4-ші тестте сұрақтар жолдармен жұмыс істеу, олардың әдістері, біріктіру және индекстеу операциялары туралы білімді тексереді. Сұрақтар мәтіндік деректерді манипуляциялауды және жол функцияларын пайдалануды қамтиды. Тест мәтінмен жұмыс істеу, Python-да жолдарды бөлу және өзгерту қабілетін тексеруге көмектеседі.



Сурет 36 5-ші тест құрылымының коды

5-ші ест сұрақтары логикалық мәндермен және әртүрлі салыстыру операцияларымен жұмыс істеуге қатысты. Сұрақтар салыстыру операторлары туралы білімді тексереді, мысалы, «тең», «тең емес», «артық» және т.б. сонымен қатар логикалық операторлармен және шарттармен жұмыс істеуге баса назар аударылады, бұл Python-да логикалық дағдыларды дамытуға көмектеседі.



Сурет 37 6-ші тест құрылымының коды

6-шы тест сұрақтары Python-дағы операторлардың барлық түрлерінің білімін тексеруге көмектеседі: арифметикалық, логикалық және салыстырмалы. Тест сонымен қатар есептеулер, логикалық өрнектер және шарттар сияқты әртүрлі сценарийлерде операторларды дұрыс пайдалану қабілетін тексереді. Сұрақтар операторлардың қалай жұмыс істейтіні және бағдарламаның орындалуына қалай әсер ететіні туралы жақсы түсінік береді.



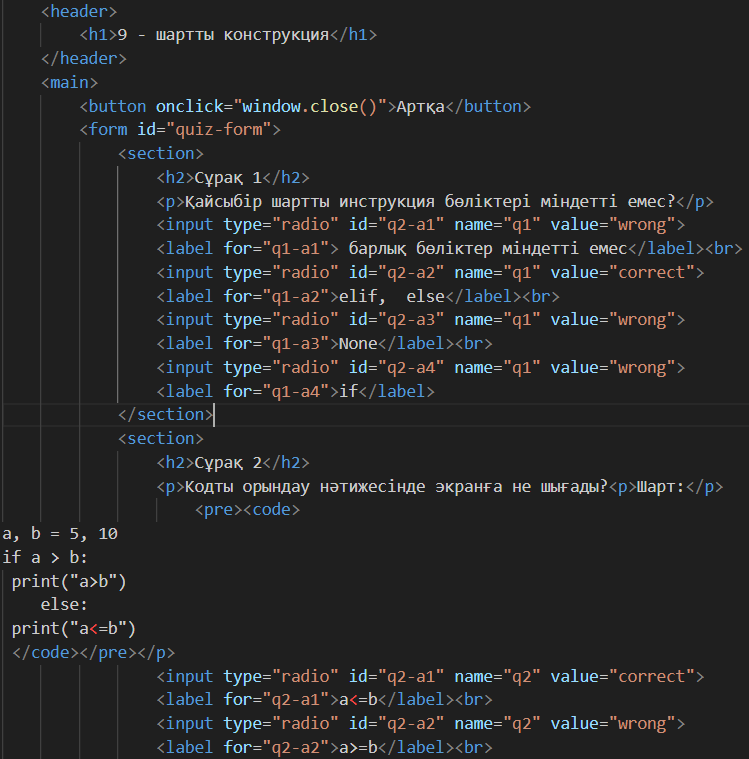
Сурет 38 7-ші тест құрылымының коды

7-ші тест сұрақтары Python-дағы ең маңызды деректер құрылымы болып табылатын тізімдерді құруға және олармен жұмыс істеуге қатысты. Тест тізім элементтерін индекстеу, сондай-ақ Қосу, жою және сұрыптау сияқты манипуляциялар туралы білімді тексереді. Сұрақтар сонымен қатар кірістірілген тізімдер мен бөлімдермен жұмыс істеу әдістерін қамтиды.



Сурет 39 8-ші тест құрылымының коды

8-ші тест бағдарламадағы жағдайларды ұйымдастыру үшін if және else операторларын пайдалану қабілетін тексеретін сұрақтардан тұрады. Сұрақтар Elif және үштік оператор сияқты шартты операторлардың әртүрлі түрлерін, сондай-ақ шарттардың дұрыстығын тексеруді қамтиды. Бұл бағдарламаның логикасын түсіну және шарттарға негізделген шешім қабылдау үшін маңызды тақырып.



Сурет 40 9-шы тест құрылымының коды

9-шы есттегі сұрақтар шарт орындалғанға дейін қайталанатын әрекеттерді орындау үшін while циклін қолдануға қатысты. Тест шексіз циклдарды құру, цикл айнымалыларымен жұмыс істеу және үзіліс операторын пайдаланып циклды дұрыс аяқтау туралы білімді тексереді. Сұрақтар Ииерация және жағдайды бақылау дағдыларын дамытуға көмектеседі.



Сурет 41 10-шы тест құрылымының коды

10-шы тест сұрақтары деректер жинақтары мен диапазондары бойынша Итерация үшін қолданылатын for циклімен жұмыс істеу туралы білімді тексеруге көмектеседі. Тест сонымен қатар range() функциясын пайдалану және коллекция индекстерін манипуляциялау бойынша сұрақтарды қамтиды. Сұрақтар деректер жинақтарымен жұмыс істеу және әртүрлі тапсырмалар үшін for циклін дұрыс ұйымдастыру қабілетін тексереді.

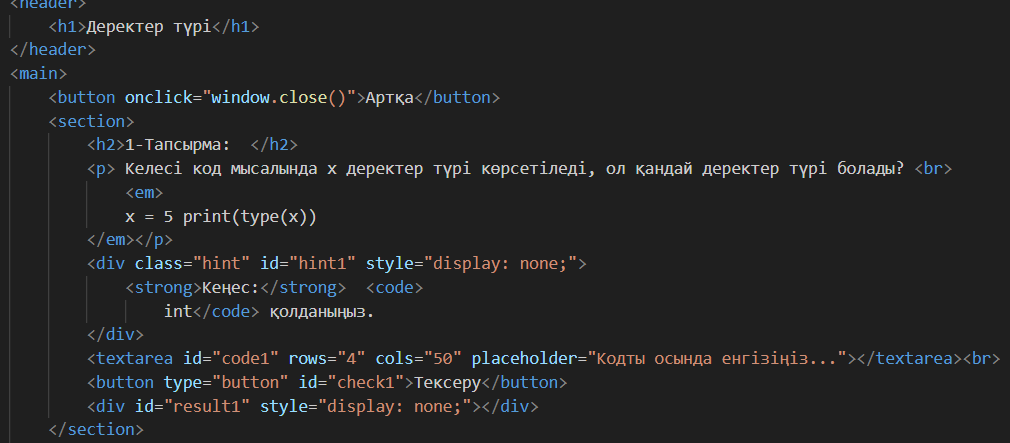
# 1.5.3 Практика

1-ші практикалық жұмыс «Айнымалылар» деп аталады. Практика барысында Python-да айнымалыларды құру, оларға мәндерді тағайындау және қолдану тәсілдері зерттелді. Айнымалыларды қолдану арқылы мәліметтерді сақтау және олармен жұмыс істеудің негізгі дағдылары меңгерілді. 1-ші практикалық жұмыстың 1-ші тапсырмасы 42-ші суретте көрсетілген.



Сурет 42 «Айнымалылар» құрылымының коды

2-ші практикалық жұмыс «Деректер түрі». Бұл практикалық жұмыс Python-да қолданылатын деректер түрлерін зерттеуді қамтиды. Сандар, жолдар, логикалық мәндер және күрделі деректер түрлерімен жұмыс істеу тәсілдері қарастырылды. Әртүрлі деректер түрлерін таңдау және оларды тиімді пайдалану дағдылары қалыптастырылады. 2-ші практикалық жұмыстың 1-ші тапсырмасы 43-ші суретте көрсетілген.



Сурет 43 «Деректер түрі» құрылымының коды

3-ші практикалық жұмыс «Жолдар». Бұл жұмыс Python-да мәтіндік деректермен жұмыс істеуді зерттеуге арналған. Жолдар құру, оларды біріктіру, өзгерту және қажетті ақпаратты шығару үшін әдістерді қолдану тәсілдері зерттелді. Сонымен қатар, Python-ның жолдармен жұмыс істеуге арналған ендірілген функциялары қолданылды. 3-ші практикалық жұмыстың 1-ші тапсырмасы 44-ші суретте көрсетілген.



Сурет 44 «Жолдар»құрылымының коды

4 практикалық жұмыс «Логикалық мәндер». Бұл практикалық жұмыс логикалық мәндер және оларды шартты операторларда пайдалану әдістеріне арналған. Практика барысында логикалық мәндермен операциялар орындалды және шешім қабылдау бағдарламаларын құру арқылы шарттарды тексеру тәсілдері зерттелді. 4-ші практикалық жұмыстың 1-ші тапсырмасы 45-ші суретте көрсетілген.



Сурет 45 «Логикалық мәндер» құрылымының коды

5 практикалық жұмыс «Тізімдер». «Тізімдер» тақырыбындағы бұл жұмыс мәліметтердің жинақтарымен жұмыс істеуді қарастырады. Тізімдер құру, олардың элементтерін қосу, жою және өзгертуді үйрену үшін әртүрлі тапсырмалар орындалды. Сонымен қатар, тізім элементтерімен қайталау және өңдеу тәсілдері зерттелді. 5-ші практикалық жұмыстың 1-ші тапсырмасы 46-ші суретте көрсетілген.



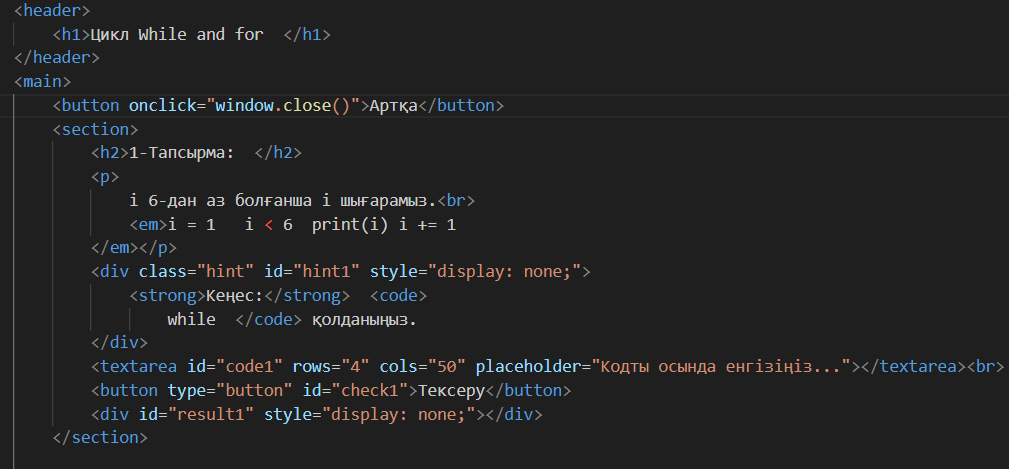
Сурет 46 «Тізімдер» құрылымының коды

6-шы практикалық жұмыс: «Операторлар». Бұл жұмыс Python-да арифметикалық, салыстыру және логикалық операторларды қолдануды қамтиды. Операторларды пайдалана отырып, есептеулер мен логикалық тексерулер жүргізу дағдылары дамытылды. Операторлардың әртүрлі типтері және олардың қолдану жағдайлары зерттелді. 6-шы практикалық жұмыстың 1-ші тапсырмасы 47-ші суретте көрсетілген.



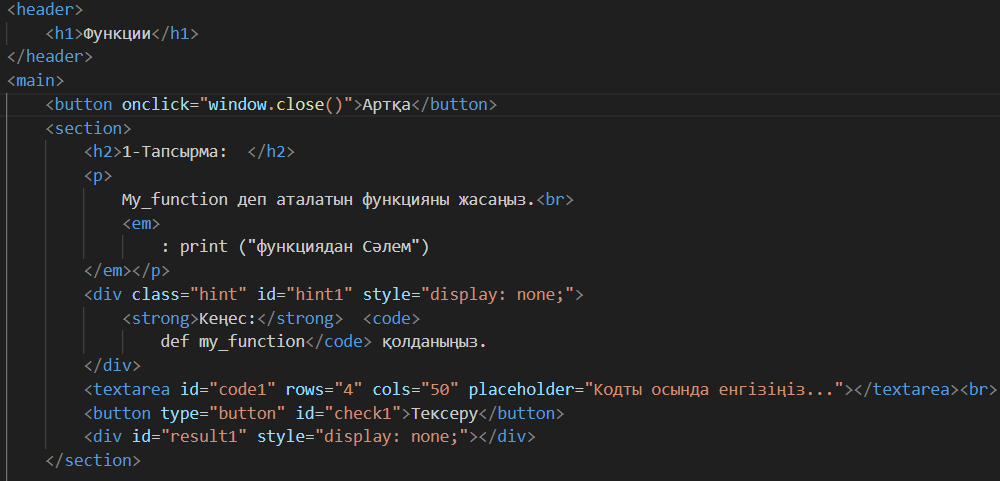
Сурет 47 «Операторлар» құрылымының коды

7-ші практикалық жұмыс «Цикл While және For». Бұл жұмыс Python-да қайталанатын әрекеттерді орындау үшін циклдармен жұмыс істеуге арналған. While циклы белгілі бір шарт орындалғанға дейін әрекеттерді қайталауды зерттесе, For циклы коллекция элементтерін қайталауға бағытталған. Циклдарды тиімді қолдану және қайталанатын есептерді шешу дағдылары жетілдірілді. 7-ші практикалық жұмыстың 1-ші тапсырмасы 48-ші суретте көрсетілген.



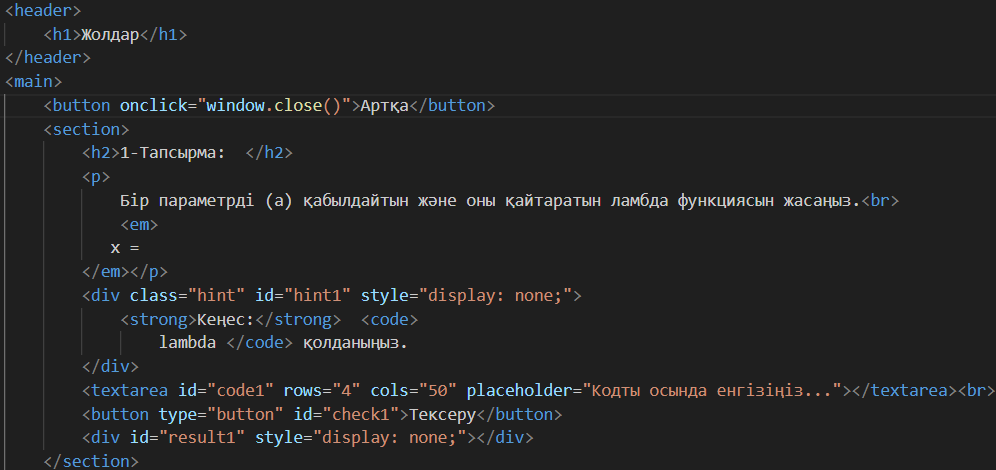
Сурет 48 «Цикл While және For» құрылымының коды

8-ші практикалық жұмыс «Функциялар» тақырыбына арналған. Бұл жұмыс бағдарламаларды құрылымдау және кодты қайта пайдалану әдістерін зерттейді. Функцияларды анықтау, оларға параметрлер беру және мәндерді қайтару тәсілдері қарастырылды. Практика барысында әртүрлі есептерді шешуге арналған жеке функциялар құрылды. 8-ші практикалық жұмыстың 1-ші тапсырмасы 49-ші суретте көрсетілген.



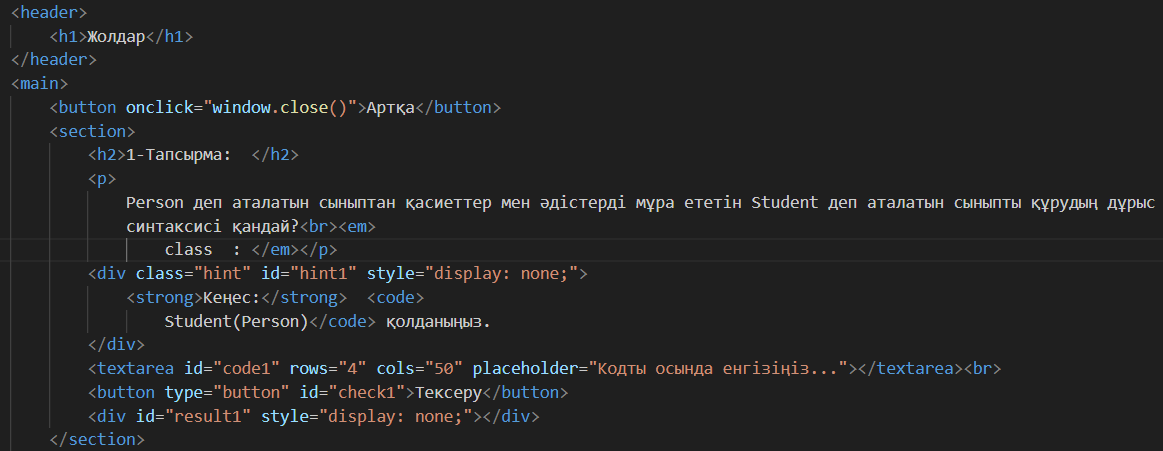
Сурет 49 «Функциялар» құрылымының коды

9-шы практикалық жұмыс «Лямбда». Бұл жұмыс Python-ның ықшам және анонимді функциялары болып табылатын лямбда функцияларымен жұмыс істеуге арналған. Лямбда функцияларын анықтау және оларды басқа функциялармен бірге қолдану әдістері зерттелді. Лямбда функцияларының жеңіл әрі жылдам шешім қабылдау үшін пайдаланылуы қарастырылды. 9-шы практикалық жұмыстың 1-ші тапсырмасы 50-ші суретте көрсетілген.



Сурет 50 «Лямбда» құрылымының коды

10-шы практикалық жұмыс «Мұрагерлік». Бұл жұмыс объектілі-бағытталған бағдарламалаудың негізгі принциптерінің бірі - мұрагерлік тақырыбын қамтиды. Ата-ана және бала класстарын құру және олардың әдістері мен қасиеттерін мұрагерлікке алу әдістері зерттелді. Мұрагерлікті қолдану арқылы кодтың құрылымын жақсарту және қайталануды азайту тәсілдері қарастырылды.10-шы практикалық жұмыстың 1-ші тапсырмасы 51-ші суретте көрсетілген.



Сурет 51 «Мұрагерлік» құрылымының коды

Әрбір практикалық жұмыс дәріс тақырыптарын тереңірек меңгеруге және Python бағдарламалау тілінің негізгі мүмкіндіктерін тәжірибе жүзінде зерттеуге бағытталды.

# 1.6 JavaScript көмегімен интерактивтілікті қосу

Оқулықтың интерактивтілігі JavaScript арқылы жүзеге асырылады.

Тест нәтижелерін тексеру және көрсету

* document.getElementById('submit-quiz').addEventListener('click', function () { Бұл жолда «submit-quiz» түймесіне басылған кезде орындалатын функция тіркеледі.
* const questions = document.querySelectorAll('form section'); Барлық сұрақтарды қамтитын «form» ішіндегі «section» элементтері таңдалады және questions айнымалысына сақталады.

let score = 0; және let incorrectAnswers = [];

* score дұрыс жауаптардың санын есептеу үшін пайдаланылады.
* incorrectAnswers массиві дұрыс емес немесе жауап берілмеген сұрақтардың нөмірлерін сақтау үшін пайдаланылады.
* questions.forEach((section, index) => { Сұрақтар тізімі бойынша цикл басталады, әр сұрақ үшін section және оның нөмірі index беріледі.
* const questionNumber = index + 1; Әр сұрақтың нөмірі есептеледі (index нөлден басталады, сондықтан +1 қосылады).
* const selected = section.querySelector('input[name="q' + questionNumber + '"]:checked'); Әр сұрақ үшін пайдаланушы таңдаған (белгіленген) жауап таңдалады.

Егер ешқандай жауап таңдалмаса, selected мәні null болады.

if (selected) Егер жауап таңдалған болса:

* if (selected.value === 'correct') Таңдалған жауаптың мәні correct болса, score біреуге артады.
* else { incorrectAnswers.push(...) } Қате жауап болса, сұрақ нөмірі incorrectAnswers массивіне қосылады.
* else { incorrectAnswers.push(...) } Егер жауап таңдалмаған болса, сұрақ нөмірі «жауап жоқ» деп массивке қосылады.

Нәтижелерді көрсету:

* document.getElementById('score').textContent = ...; Дұрыс жауаптар саны мен сұрақтардың жалпы саны көрсетіледі.
* document.getElementById('incorrect-answers').textContent = ...; Қате жауаптар немесе олардың болмағаны туралы хабарлама шығарылады.
* document.getElementById('quiz-results').classList.remove('hidden');

Нәтижелер көрсетіледі (бұрын жасырылған болса).



Сурет 52 Тест нәтижелерін тексеру коды

Радио батырмаларды құлыптау және тестті қайта тапсыру коды.

Радио түймелерін құлыптау:

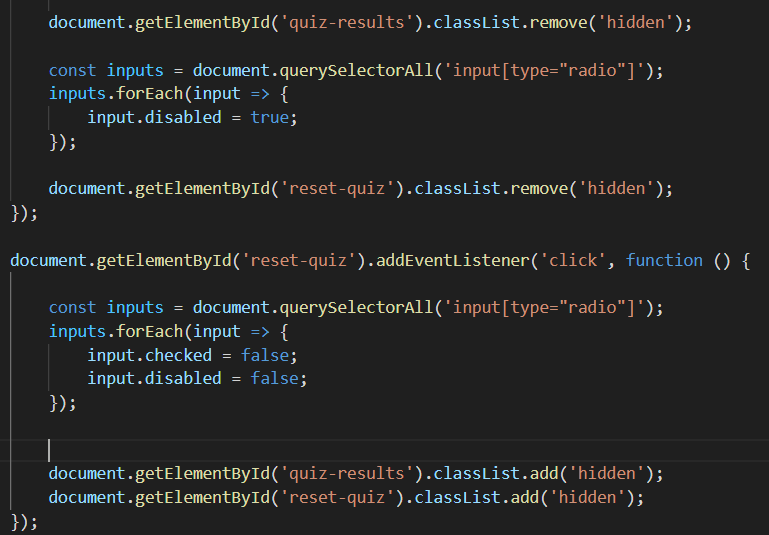
* const inputs = document.querySelectorAll('input[type="radio"]'); Барлық радио түймелер таңдалады.
* inputs.forEach(input => { input.disabled = true; }); Әр радио түйме disabled = true арқылы құлыпталады.
* document.getElementById('reset-quiz').classList.remove('hidden'); «reset-quiz» қайта тапсыру түймесін көрсететін болады.

Тестті қайта тапсыру:

* document.getElementById('reset-quiz').addEventListener('click', function () { «reset-quiz» түймесі басылған кезде орындалатын функция тіркеледі.
* inputs.forEach(input => { input.checked = false; input.disabled = false; }); Барлық радио түймелердің таңдау белгілері алынып, қайта таңдау үшін белсендіріледі.

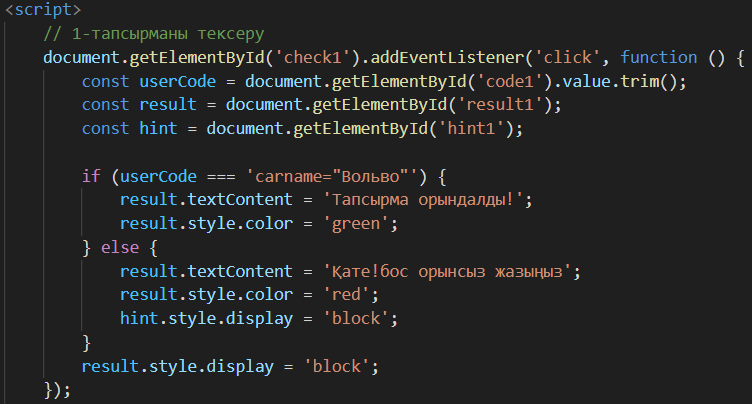
Нәтижелерді жасыру:

* document.getElementById('quiz-results').classList.add('hidden'); Нәтижелер бөлімін жасырады.
* document.getElementById('reset-quiz').classList.add('hidden'); Қайта тапсыру түймесі көрінбейді.



Сурет 53 Тестті қайта тапсыру коды

JavaScript-тің маңыздылығы оның жылдамдығы мен әмбебаптығында жатыр. Ол браузерде тікелей орындалатындықтан, пайдаланушыларға интерактивтілік үшін қосымша бағдарламаларды жүктеудің қажеті жоқ. Бұл әсіресе білім беру ресурстарында маңызды, себебі оқушылардың назарын сабаққа шоғырландыру және жылдам кері байланыс ұсыну үшін веб-беттердің жылдам жұмыс істеуі қажет. JavaScript арқылы оқушыларға тапсырмаларда көмек ретінде кеңестер беру, қателерді түсіндіру және қайта тапсыру мүмкіндігін ұсыну оңай жүзеге асады. Мысалы, 1-ші тапсырманы тексеру коды 54 суретте көрсетілген.



Сурет 54 1 1-ші тапсырманы тексеру коды

Бұл JavaScript коды пайдаланушы енгізген айнымалының мәнін тексереді.

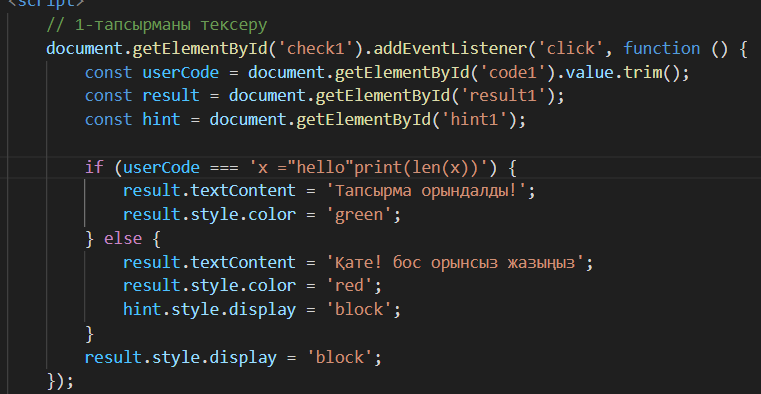
* document.getElementById('check1') элементін таңдайды және оған оқиға тыңдаушыны (click) қосады.
* document.getElementById('code1').value.trim() арқылы пайдаланушы енгізген код алынады.
* Егер енгізілген код берілген дұрыс жауапқа (carname="Вольво") сәйкес келсе, нәтиже жасыл түспен «Тапсырма орындалды!» деп көрсетіледі. Қате болса, пайдаланушыға «Қате!» деген хабарлама шығады, және кеңес (hint) көрсетіледі.

Екінші кодта тапсырма ретінде бүтін сан айнымалысын құрып, оның типін анықтау ұсынылады. Пайдаланушы енгізген код x =5print(type(x))int үлгісіне сәйкес болуы тиіс. Егер енгізілген код дұрыс болса, нәтиже «Тапсырма орындалды!» деген жасыл мәтінмен көрсетіледі. Қате жағдайда, «Қате! бос орынсыз жазыңыз» деген хабарлама және көмекші кеңес пайда болады. Бұл тексеру механизмі пайдаланушыға нақты қандай қатені түзету керектігін көрсетеді. 2-ші тапсырманы тексеру коды 55 суретте көрсетілген.



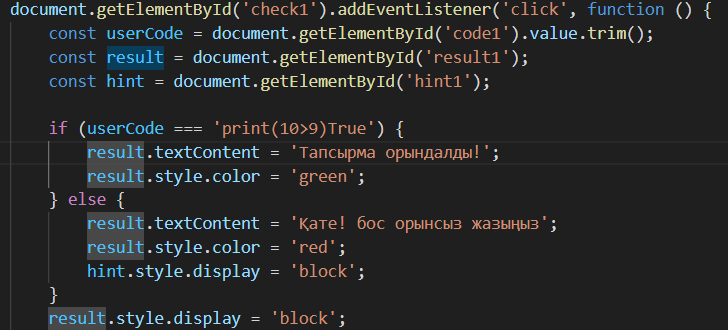
Сурет 55 2-ші тапсырманы тексеру коды

Үшінші кодта пайдаланушыдан жол айнымалысын құрып, оның ұзындығын анықтау сұралады. Енгізілген код x ="hello"print(len(x)) үлгісінде болуы қажет. Егер енгізілген жауап дұрыс болса, тапсырманың сәтті орындалғаны жасыл түспен көрсетіледі. Қате болған жағдайда, «Қате! бос орынсыз жазыңыз» деген қызыл мәтін пайда болып, пайдаланушыға кеңес беріледі. Бұл тапсырма жолдармен жұмыс істеу және олардың ұзындығын есептеуді үйретуге бағытталған. 3-ші тапсырманы тексеру коды 56 суретте көрсетілген.



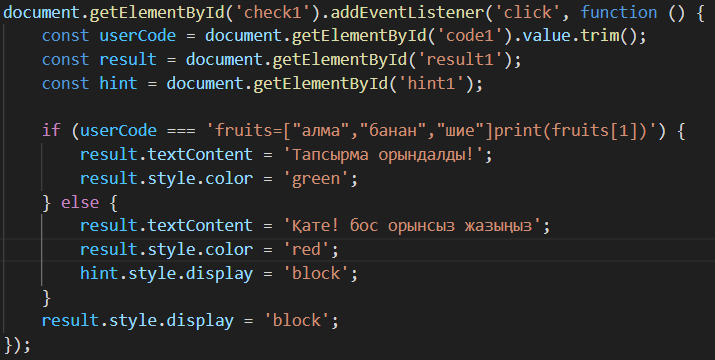
Сурет 56 3-ші тапсырманы тексеру коды

Төртінші кодта пайдаланушыдан 10 > 9 шартын тексеретін бағдарламаны жазуды талап етеді. Пайдаланушы енгізген код print(10>9)True үлгісіне сәйкес болуы керек. Егер код дұрыс болса, «Тапсырма орындалды!» деген жасыл мәтін пайда болады. Қате енгізілген жағдайда, «Қате! бос орынсыз жазыңыз» деген ескерту және көмекші кеңес көрсетіледі. Бұл тапсырма пайдаланушыға салыстыру операторларын қолдану арқылы логикалық мәндерді түсінуді үйретеді. 4-ші тапсырманы тексеру коды 57 суретте көрсетілген.



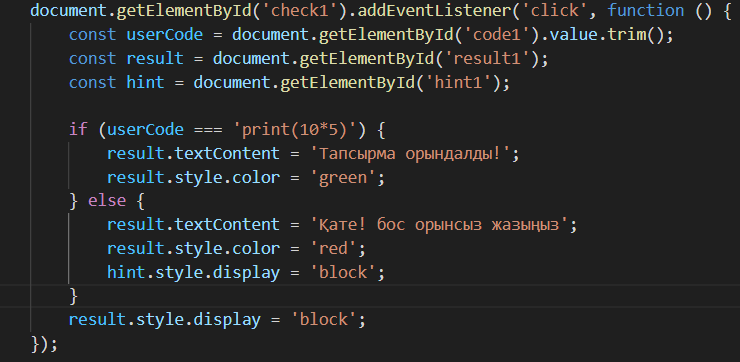
Сурет 57 4-ші тапсырманы тексеру коды

Бесінші кодта пайдаланушыдан тізім құрып, оның белгілі бір элементін таңдау сұралады. Күтілетін код: fruits=[«алма», «банан», «шие»]print(fruits[1]). Егер пайдаланушы дәл осылай енгізсе, тапсырма сәтті орындалғаны туралы жасыл мәтін көрсетіледі. Қате болған жағдайда, пайдаланушыға қате туралы хабарлама мен көмекші кеңес беріледі. Бұл тапсырма тізімдермен жұмыс істеу және элементтерді индекстеуді үйренуге бағытталған. 5-ші тапсырманы тексеру коды 58 суретте көрсетілген.



Сурет 58 5-ші тапсырманы тексеру коды

Алтыншы код арифметикалық операцияларды, атап айтқанда көбейтуді тексеруге арналған. Пайдаланушыдан 10 \* 5 мәнін шығаратын код жазу сұралады. Дұрыс код: print(10\*5). Егер енгізілген жауап дұрыс болса, «Тапсырма орындалды!» деген жасыл мәтін пайда болады. Қате енгізілген жағдайда, пайдаланушыға «Қате! бос орынсыз жазыңыз» деген қызыл хабарлама беріледі. Бұл тапсырма математикалық операцияларды қолдануды үйренуге арналған. 6-шы тапсырманы тексеру коды 59 суретте көрсетілген.



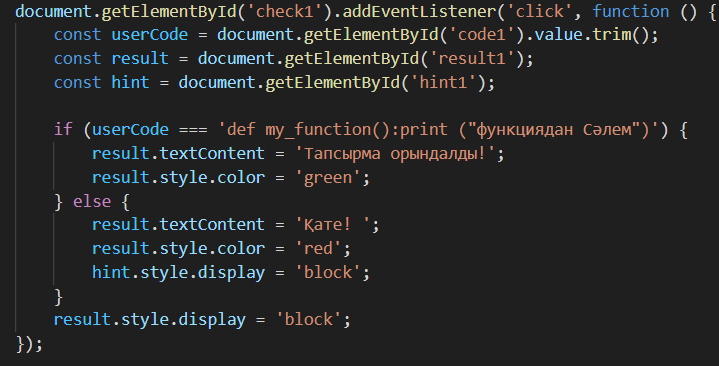
Сурет 59 6-шы тапсырманы тексеру коды

Жетінші кодта пайдаланушыдан while циклімен берілген шартқа сәйкес есеп шығару сұралады. Код үлгісі: i = 1 while i < 6 : print(i) i += 1. Егер пайдаланушы осы кодты дұрыс енгізсе, бағдарлама «Тапсырма орындалды!» деген жасыл мәтінді көрсетеді. Қате болған жағдайда, «Қате!» деген қызыл мәтін көрсетіледі, ал көмекші кеңес пайда болады. Бұл тапсырма циклмен жұмыс істеп, қайталауды жүзеге асыруды үйретеді. 7-ші тапсырманы тексеру коды 60 суретте көрсетілген.



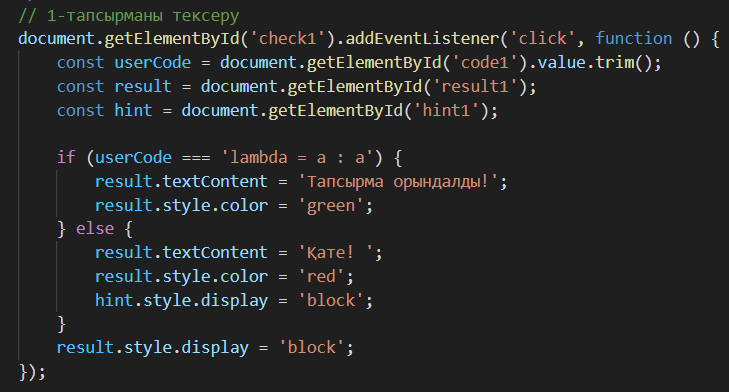
Сурет 60 7-ші тапсырманы тексеру коды

Сегізінші тапсырмада пайдаланушыдан Python тілінде функция анықтап, оны іске қосу сұралады. Дұрыс код: def my\_function():print ( «функциядан Сәлем»). Егер пайдаланушы функцияны дұрыс жазса, тапсырма сәтті орындалады, және экранда жасыл түс пен «Тапсырма орындалды!» мәтіні шығады. Қате енгізген жағдайда, қызыл түсті хабарлама көрсетіледі, ал көмекші кеңес қосылады. Бұл тапсырма функцияларды қалай дұрыс анықтап, шақыруды үйретуге арналған. 8-ші тапсырманы тексеру коды 61 суретте көрсетілген.



Сурет 61 8-ші тапсырманы тексеру коды

Тоғызыншы кодта лямбда функциясын пайдалану қажет. Күтілетін код: lambda = a : a. Егер пайдаланушы дәл осы кодты енгізсе, «Тапсырма орындалды!» деген жасыл хабарлама көрсетіледі. Қате енгізген жағдайда, «Қате!» деген қызыл хабарлама шығады, көмекші кеңес көрсетіледі. Бұл тапсырма лямбда функциясының негізгі синтаксисі мен жұмыс принципін түсінуге көмектеседі. 9-шы тапсырманы тексеру коды 62 суретте көрсетілген.



Сурет 62 9-шы тапсырманы тексеру коды

Оныншы тапсырмада Python-дағы мұрагерлік принципін қолдану қажет. Дұрыс код: class Student(Person): . Егер код дәл осылай енгізілсе, тапсырма орындалды деген жасыл хабарлама шығады. Қате енгізілген жағдайда, қызыл түсті қате туралы хабарлама мен көмекші кеңес беріледі. Бұл тапсырма Python-да класстардың мұрагерлік қатынасын орнатуды үйретуге арналған. 10-шы тапсырманы тексеру коды 63 суретте көрсетілген.



Сурет 63 10-шы тапсырманы тексеру коды

Әрбір тапсырма қолданушыдан кодты нақты және дұрыс жазуды талап етеді, сондықтан олар бағдарламалау тіліндегі синтаксис пен логиканы дұрыс түсінуді қалыптастырады. Әр тапсырма аяқталған соң, нәтижелер көрсетіледі: егер код дұрыс жазылса, тапсырма сәтті орындалғаны туралы хабарлама жасыл түспен шығады; қате болған жағдайда, пайдаланушыға қызыл түспен қате туралы хабарлама көрсетіледі, және көмекші кеңес қосылады.

Осылайша, мұндай тапсырмалар тек код жазу дағдыларын дамытып қана қоймай, сонымен қатар білім беру процесін интерактивті етіп, қолданушылардың қызығушылығын арттырады. Әрбір тапсырмада Python тілінің негізгі аспектілері - айнымалылар, операторлар, циклдер, функциялар және мұрагерлік сынды маңызды түсініктер қамтылған, бұл студенттерге олардың теориялық білімдерін практикада қолдануға мүмкіндік береді.

# 1.7 Тестілеу және жөндеу

Бұл бөлімде дипломдық жобада қолданылған интерактивті платформаның тестілеу және жөндеу жұмыстары сипатталады. Бағдарламаның дұрыс жұмыс істеуін қамтамасыз ету және қолданушыларға тиімді әрі түсінікті құрал ұсыну мақсатында тестілеу және жөндеу кезеңдері өткізілді. Осы бөлімде жүйенің функционалдық және техникалық сипаттамаларына сәйкес тестілеудің негізгі әдістері, тестілеу нәтижелері мен анықталған қателердің жөнделуі қарастырылады.

Тестілеу әдістері:

* Дипломдық жобада қолданылған интерактивті платформа Python тілінде жазылған тапсырмаларды орындауды қамтамасыз етеді. Әрбір тапсырманың дұрыс орындалуы үшін тестілеу кезеңінде оның функционалдылығы тексерілді.
* Тестілерді өткізу барысында платформаның әртүрлі элементтерінің (код енгізу өрістері, түймелер, нәтижелердің көрсетілуі) жұмыс істеп тұрғандығы тексерілді. Әрбір тапсырманың дұрыс немесе қате орындалуы, сәйкесінше жасыл немесе қызыл түс арқылы пайдаланушыға көрсетілді.
* Тестілер әртүрлі жауаптарды тексеру үшін қолданушылардың енгізген кодтарын бағалауға бағытталды. Платформа барлық тапсырмаларды дұрыс тексеруге және нәтижелерді нақты көрсетуге тиіс болды.

Тестілеу барысында бірнеше қателер анықталды, олардың негізгілері:

* Пайдаланушының енгізген жауаптары кейбір жағдайларда дұрыс бағаланбады. Бұл мәселені жөндеу үшін тестілеу логикасын қайта қарастырдық, нәтижесінде кодтың дұрыстығы мен тиімділігі артты.
* Қосымша функционалдық ақаулар да орын алған кезде, мәселен, дұрыс жауаптардың көрсетілуі үшін жауап мәтіндері мен түстердің стилін қайтадан орнаттық.
* Бағдарлама интерфейсіндегі кейбір элементтер дұрыс орналаспайтын жағдайлар туындады. Бұл мәселе үшін UI компоненттерінің орналасуы мен жауапкершілігін қайтадан тексердік.

Қолданушы интерфейсінің тестілеуі:

* Платформа интерфейсінің қолданушыға ыңғайлы болуы үшін тестілеу жүргізілді. Қолданушылардың жауаптары дұрыс қабылдануы үшін барлық элементтер жүйелі түрде тексерілді. Қолданушының күтуін қанағаттандыру үшін интерфейс тез әрі түсінікті түрде өзгертілді.
* Тестілер кезінде қолданушының интерфейспен жұмыс жасау барысында оңай әрі қарапайым жолмен тапсырмаларды орындауы үшін барлық мүмкін болатын қиындықтар мен мәселелер шешілді.

Қорытынды:

* Тестілеу және жөндеу жұмыстары платформаның сенімділігі мен тиімділігін арттыруға мүмкіндік берді. Барлық қателер жойылып, платформаның функционалдығы жоғары деңгейде қамтамасыз етілді. Жобаның соңғы нұсқасы қолданушыларға ыңғайлы, жұмыс істеуге дайын, барлық тапсырмаларды дұрыс орындауға қабілетті болып табылды.

Тестілеу кезеңінің нәтижесінде жүйенің тұрақтылығы мен сенімділігі тексерілді, және қолданушыға максималды ыңғайлылықты қамтамасыз ету үшін интерфейс тиімді түрде жетілдірілді.

# 2 ЭКСПЕРИМЕНТТІК БӨЛІМ

# 2.1 Эксперименттің мақсаты мен міндеттері

Цифрлық технологиялардың дамуы білім беру процесіне жаңа әдістер мен құралдарды енгізуге мүмкіндік беріп отыр. Осы орайда Python бағдарламалау тіліне арналған электрондық оқулықтың тиімділігін зерттеу - өзекті педагогикалық мәселе. Эксперименттің басты мақсаты - жасалған электрондық оқулықтың студенттердің пәнге деген қызығушылығына, теориялық білімін меңгеруіне және практикалық дағдыларды қалыптастыруына ықпалын анықтау.

* Эксперимент міндеттері ретінде төмендегілер анықталды:
* Студенттердің бастапқы білім деңгейін бағалау және оқулықпен жұмыс істеу арқылы білім сапасының өзгерісін бақылау;
* Электрондық оқулық құрылымының, мазмұнының және интерактивті мүмкіндіктерінің студенттерге әсерін зерттеу;
* Тәжірибе барысында жинақталған деректер негізінде оқулықтың тиімділігін нақтылау және болашақта оны жетілдіру жолдарын ұсыну;
* Педагогикалық эксперименттің нәтижесінде алынған мәліметтерді талдап, нақты қорытындылар мен ұсыныстар беру.

# 2.2 Эксперименттік топтар

Экспериментке педагогикалық жоғары оқу орнының 2-курс студенттері қатысты. Барлығы 30 студент қатысып, олар кездейсоқ таңдау арқылы екі топқа бөлінді:

* Эксперименттік топ – 15 студент электрондық оқулықпен жұмыс істеді;
* Бақылау тобы – 15 студент дәстүрлі оқыту әдістерін пайдаланды.

Екі топтың да бастапқы білім деңгейі ұқсас болғаны бастапқы тест нәтижелерімен расталды. Бұл эксперимент нәтижелерінің шынайылығын қамтамасыз ету үшін маңызды фактор болып табылады.

Топтағы студенттердің жас шамасы, академиялық үлгерімі, пәнге қызығушылық деңгейі алдын ала зерттеліп, оқу үдерісіне кедергі келтіретін факторлар жойылды. Барлық қатысушылар өз еркімен қатысуға келісім берді, ал сабақтар оқу кестесіне сәйкес жүргізілді.

# 2.3 Эксперимент өткізу барысы

Эксперимент үш негізгі кезеңнен тұрды:

1. Диагностикалық кезең. Бірінші кезеңде барлық қатысушыға Python бағдарламалау тілі бойынша бастапқы білім деңгейін анықтау мақсатында бірдей тест тапсырылды. Бұл кезеңде студенттерге 20 сұрақтан тұратын тест ұсынылды. Сұрақтар Python синтаксисі, деректер типтері, циклдер, шартты операторлар және функциялар туралы болды. Бұл кезеңнің мақсаты – бастапқы білім деңгейі бірдей екенін растау және алдағы нәтижелермен салыстыруға негіз қалыптастыру.

2. Эксперименттік оқыту кезеңі. Төрт апта бойы эксперименттік топ электрондық оқулық арқылы оқып, лекция материалдарын онлайн түрде қарап, тест тапсырмаларын орындап, практикалық жаттығулар шешті. Оқулықтың интерактивті элементтері - код редакторлары, тапсырма тексерушілері мен нәтижелерді бірден көрсету жүйесі - студенттердің белсенді қатысуына ықпал етті. Ал бақылау тобы дәстүрлі әдіспен - оқытушының түсіндіруі, тақтада жазу, дәптермен жұмыс - оқыды. Оларға ешқандай электрондық материалдар берілмеді.

3. Қорытынды кезең. Оқыту аяқталған соң, екі топқа бірдей қорытынды тест және анонимді сауалнама ұсынылды. Тест мазмұны бастапқы тестпен ұқсас болғанымен, күрделілік деңгейі жоғары болды. Бұл кезең оқулықтың тиімділігін сандық тұрғыдан бағалауға мүмкіндік берді.

# 2.4 Сауалнама және кері байланыс

Эксперименттік топқа арналған сауалнамада студенттердің электрондық оқулықпен жұмыс істеу процесі туралы пікірлері зерттелді. Сауалнамада келесі сұрақтар болды:

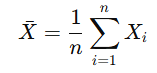
* Материалдардың түсініктілігі мен құрылымы қаншалықты ыңғайлы болды?
* Интерактивті тапсырмалар мен тесттер көмектесті ме?
* Практикалық дағдыларды игеруге әсері болды ма?
* Жалпы бағалауыңыз қандай (1-ден 5-ке дейін)?

Сауалнама нәтижелері:

* 12 студент оқулық материалдарының құрылымын «өте жақсы» деп бағалады;
* 13 студент интерактивтілік оқу мотивациясын арттырғанын айтты;
* 11 студент практикалық тапсырмалардың нақты жағдайларға жақын екенін атап өтті;
* Орташа қанағаттану көрсеткіші: 4.6 балл / 5.0.

Кері байланыс нәтижелері оқулықтың студенттерге түсінікті, қолжетімді әрі пайдалы құрал болғанын көрсетті.

# 2.5 Нәтижелерді өңдеу және есептеу формуласы

Эксперименттің нәтижелерін талдау үшін студенттердің тестілеуден алған ұпайлары негізінде орташа арифметикалық есептеу жүргізілді. Формула: мұндағы:

Xˉ– студенттердің орташа балы;

Xi – әрбір студенттің тесттен алған балы;

n – студенттер саны.

Мысал:

Эксперименттік топ (15 студент):

Балдар = 78, 82, 85, 90, 87, 88, 92, 80, 84, 86, 89, 91, 79, 83, 88

Жалпы: 1292 балл

Орташа балл:



Бақылау тобы (15 студент):

Балдар = 66, 72, 70, 75, 68, 74, 76, 69, 71, 73, 70, 68, 72, 67, 74

Жалпы: 1115 балл

Орташа балл:



# 2.6 Нәтижелерді талдау

Нәтижелер көрсеткендей, электрондық оқулықпен жұмыс істеген студенттердің орташа балы 86.13, ал дәстүрлі әдіспен оқыған студенттердікі 74.33 болды. Бұл айырмашылық шамамен 11.8%. Сонымен қатар, сауалнама нәтижелері студенттердің электрондық құралды оң қабылдағанын көрсетті.

Алынған мәліметтер негізінде, электрондық оқулық оқу процесін жеңілдетіп қана қоймай, студенттердің пәнге деген қызығушылығын арттыруға, теориялық білімдерін тереңдетуге және практикалық дағдыларын дамытуға нақты ықпал ететіні дәлелденді.

# ҚОРЫТЫНДЫ

Әзірленген оқулық Python тілін үйренуге арналған кешенді құрал болып табылады. Ол материалды жақсы игеруге ықпал ететін теорияны, тәжірибені және білімді тексеруді біріктіреді. Оқулық өзін-өзі зерттеу үшін де, білім беру процесінде де қолданыла алады. Оқулықта берілген тапсырмалар мен жаттығулар студенттердің теориялық білімдерін бекітуге және практикалық дағдыларын қалыптастыруға көмектеседі. Әрбір тақырып бойынша интерактивті тапсырмалар мен жаттығуларды орындау оқушының Python тілін терең түсініп, тиімді пайдалануына мүмкіндік береді.

Оқулықтың құрылымы әртүрлі деңгейдегі пайдаланушылар үшін қолайлы болып жасалған, яғни ол бастаушы деңгейден бастап, күрделі мәселелерді шешуге дейінгі барлық қажетті материалдарды қамтиды. Оқулықтағы әрбір тапсырма мен жаттығу студенттердің өз бетінше білім алуға ынталандыратын, әрі олардың бағдарламалау тілін түсінуін тереңдететін ерекше мүмкіндіктер ұсынады.

Осы дипломдық жобада қолданылған оқулық платформасы мен интерактивті құралдар Python тілін үйренуге қажетті барлық шарттарды қамтиды. Оқулықты пайдалану барысында студенттер Python тілін үйренудің барлық қырларын түсініп, бағдарламалаудың негіздерін меңгере алады. Сонымен қатар, оқулықта қолданылған тестілеу жүйесі білімді тексеруге және оқушының нәтижелерін объективті бағалауға мүмкіндік береді. Оқулықтың интерактивтілігі мен қолжетімділігі оқушылардың қызығушылығын арттырып, оқу процесін тиімді әрі қызықты етеді.

Оқулықты әрі қарай жетілдіру және оны қолданушылардың пікірлері мен ұсыныстарына негізделген жаңа мүмкіндіктермен толықтыру үшін тестілеу кезеңдері мен жөндеу жұмыстары жүргізілді. Бұл процесс оқулықтың тиімділігін арттыруға және оны пайдаланушылар үшін ыңғайлы әрі түсінікті етуге бағытталған. Болашақта оқулықты жаңа тақырыптармен толықтырып, әртүрлі деңгейдегі пайдаланушыларға арналған бөлімдер енгізу жоспарлануда.

Қорыта айтқанда, осы оқулық Python тілін үйренушілер үшін құнды ресурс болып табылады және оқу процесін тиімді жүргізуге көмектесетін бірегей құрал болып саналады.

# ПАЙДАЛАНЫЛҒАН ӘДЕБИЕТТЕР ТІЗІМІ

1. Уэстерман Б. «Mastering Visual Studio Code», 2-ші басылым, 124-бет. 2020 ж.
2. Роббинс Дж. «HTML және CSS: веб-әзірлеушілерге арналған», 4-ші басылым, 89-бет. 2018 ж.
3. Дюк К. «JavaScript зерттеулері», 3-ші басылым, 245-бет. 2019 ж.
4. Фримен Э. «Modern Web Development with Fetch API», 1-ші басылым, 132-бет. 2021 ж.
5. Гус К. «Chrome Developer Tools: тиімді пайдалану», 2-ші басылым, 78-бет. 2022 ж.
6. Жұмабаев Т. «Web-технологиялар:HTML»,10-бет,2020ж.
7. Жұмабаев Т. «Web-технологиялар:CSS»,26-бет,2020ж.
8. Жұмабаев Т. «Web-технологиялар:JavaScript»,46-бет,2020ж.
9. С. Смағұлов. «Web-технологиялар:HTML»,10-бет,2019ж.
10. Live Server сайты. кіру режимі : URL: <https://www.websitebuilderpoint.net/what-is-live-server-in-web-development/> (26.12.24)
11. Бекенова А.С. «Python бағдарламалау тілі: Айнымалылар», 12-бет. 2019 ж.
12. Бекенова А.С. «Python бағдарламалау тілі: Дерек түрлері», 18-бет. 2019 ж.
13. Бекенова А.С. «Python бағдарламалау тілі: Сандар», 25-бет. 2019 ж.
14. Бекенова А.С. «Python бағдарламалау тілі: Жолдар (strings)», 30-бет. 2019 ж.
15. Бекенова А.С. «Python бағдарламалау тілі: Логикалық мәндер», 37-бет. 2019 ж.
16. Бекенова А.С. «Python бағдарламалау тілі: Операторлар», 45-бет. 2019 ж.
17. Бекенова А.С. «Python бағдарламалау тілі: Тізімдер (list)», 52-бет. 2019 ж.
18. Бекенова А.С. «Python бағдарламалау тілі: If және else», 60-бет. 2019 ж.
19. Бекенова А.С. «Python бағдарламалау тілі: WHILE циклі», 70-бет. 2019 ж.
20. Бекенова А.С. «Python бағдарламалау тілі: FOR циклі», 78-бет. 2019 ж.
21. Бекенова А.С. «Python бағдарламалау тілі: Функциялар», 85-бет. 2019 ж.
22. Бекенова А.С. «Python бағдарламалау тілі: Lambda функциялары», 93-бет. 2019 ж.
23. Бекенова А.С. «Python бағдарламалау тілі: Массивтер», 100-бет. 2019 ж.
24. Бекенова А.С. «Python бағдарламалау тілі: Мұрагерлік», 110-бет. 2019 ж.
25. Python құжаттамасы. кіру режимі : URL <https://docs.python.org/3/> (15.12.2024)
26. W3Schools сайты. кіру режимі : URL: <https://www.w3schools.com/python/> (15.12.2024)
27. Люц М. Python-5 зерттеу, 5-ші басылым 53-бет. 2021ж.