

Блок 1.2. Цикл

1.2.1. Цикл операторы

Цикл – бұл бағдарламалауда белгілі бір әрекеттерді бірнеше рет қайталап орындауға мүмкіндік беретін негізгі құрал. Олар қайталанатын есептерді автоматтандыруға және мәліметтерді жүйелі түрде өңдеуге қолданылады. Python тілінде циклдер есептерді оңай әрі құрылымды түрде шешуге мүмкіндік береді. Циклдердің басты мақсаты – бағдарламаны ықшамдау, қайталанатын әрекеттерді оңтайлы ұйымдастыру және уақытты үнемдеу. Сонымен қатар, циклдер деректерді өңдеу барысында күрделі логиканы жеңілдетуге мүмкіндік береді.

Цикл түрлері және олардың ерекшеліктері

Циклдер екі негізгі түрге бөлінеді. Біріншісі – қадам саны алдын ала белгілі цикл, бұл жағдайда орындалу саны нақты анықталған. Мұндай циклдер көбінесе **for** операторымен жүзеге асырылады. Мысалы, белгілі бір тізімдегі барлық элементтерді өңдеу немесе реттік сандарды шығару үшін қолданылады. For циклінің басты артықшылығы – оның анық әрі қарапайым құрылымында.

Екіншісі – шартпен басқарылатын цикл, мұнда цикл белгілі бір шарт орындалғанға дейін жұмыс істейді. Бұл циклдер **while** операторымен орындалады және оның орындалу ұзақтығы шартқа тәуелді болады. While циклы икемділігі арқасында түрлі сценарийлерде қолданылады, мысалы, пайдаланушыдан дұрыс мәлімет алынғанша деректер енгізуді сұрау.

Циклдер бағдарламалау процесінде икемді және кеңейтілген мүмкіндіктер ұсынады. Қосымша бақылау үшін шарттарды күрделі логикалық операторлармен біріктіруге болады.

Циклдің құрылымы және басқару операторлары

Кез келген цикл үш негізгі бөліктен тұрады: бастапқы мән, қайталану шарты және қадам. Бастапқы мән арқылы цикл айнымалысына бастапқы дерек беріледі. Қайталану шарты циклдің жалғасу немесе аяқталу шартын анықтайды. Қадам цикл айнымалысының әр итерация сайын өзгерісін белгілейді. Бұл құрылымның дұрыс ұйымдастырылуы циклдің тиімді жұмысын қамтамасыз етеді.

Python тілінде циклдерді басқару үшін **break** және **continue** операторлары қолданылады. Break операторы циклді шартсыз тоқтатып, оның орындалуын мерзімінен бұрын аяқтайды. Бұл әдіс, мысалы, белгілі бір элемент табылғанда циклден шығу үшін пайдалы. Continue операторы ағымдағы итерацияны өткізіп жіберіп, келесі итерацияға өтуді қамтамасыз етеді. Бұл тәсіл циклдегі қажетсіз қадамдарды айналып өтіп, жалпы алгоритмді бұзбай жұмыс істеуге мүмкіндік береді.

Қосымша басқару мүмкіндіктері ретінде **else** блогын қолдануға болады. Егер цикл табиғи түрде аяқталса (break қолданылмаса), else блогы орындалады. Бұл мүмкіндік циклдердің орындалу нәтижесін талдау үшін пайдалы.

Циклдердің қолдану салалары

Циклдер мәліметтерді өңдеуде маңызды рөл атқарады. Олар тізімдер, массивтер немесе жиындар сияқты құрылымдарды жүйелі түрде өңдеу, сандарды арифметикалық амалдар арқылы есептеу немесе белгілі бір шартқа сәйкес элементтерді іріктеу үшін пайдаланылады. Мысалы:

- Массивтің барлық элементтерінің қосындысын табу;
- Ең үлкен немесе ең кіші мәнді анықтау;
- Көбейту кестесін құру;
- Белгілі бір шартқа сәйкес деректерді сүзу.

Циклдер сонымен қатар пайдаланушы интерфейстерін автоматтандыру, деректерді жинақтау және талдау, күрделі есептерді модельдеу сияқты әртүрлі салаларда кеңінен қолданылады. Олар графикалық интерфейстерді құру кезінде оқиғаларды өңдеу циклдерін ұйымдастыру үшін де пайдалы.

Циклдер – бағдарламалаудың ең маңызды құрылымдарының бірі. Олар қайталанатын әрекеттерді тиімді ұйымдастырып, бағдарламаның орындалу уақытын қысқартады. Циклдерді дұрыс қолдану күрделі есептерді құрылымды және оңтайлы түрде шешуге мүмкіндік береді. Python тіліндегі цикл операторлары бағдарламаны автоматтандыруға, пайдаланушыға ыңғайлы интерфейстер құруға және алгоритмдерді жетілдіруге жағдай жасайды. Бұл құралдардың дұрыс қолданылуы бағдарламаның сапасын арттырып, жұмыс процесін жылдамдатады. Сонымен қатар, циклдер оқушылар мен бағдарламашылар үшін логикалық ойлауды дамытуға ықпал етеді, бұл оларды тиімді және шығармашылықпен жұмыс істеуге үйретеді.

1.2.2. Цикл операторы while

Циклдер бағдарламалаудың негізгі элементтерінің бірі болып табылады, олар белгілі бір шарт орындалғанға дейін қайталанатын әрекеттерді ұйымдастыруға мүмкіндік береді. Python тілінде while циклі қайталану саны алдын ала белгісіз немесе тек логикалық шартқа байланысты болатын есептерді шешу үшін кеңінен қолданылады.

While циклі – логикалық шарт ақиқат болғанша орындалатын қайталану механизмі. Цикл денесі тек шарттың нәтижесі True болған жағдайда орындалады. Егер шарт бірден False болса, цикл бірде-бір рет орындалмай тоқтайды.

While циклінің құрылымы

While циклінің жалпы синтаксисі:

```
while <шарт>:  
    <цикл денесі>
```

Бұл жерде:

- Шарт – циклдің орындалуын басқаратын логикалық өрнек. Ол True болса, цикл жалғасады. Ал егер False болса, цикл аяқталады.
- Цикл денесі – шарт орындалған кезде орындалатын операторлар жиыны.

Python тілінде циклдерді басқару үшін **break**, **continue** және **else** операторлары қолданылады. Break операторы циклді шартқа қарамастан тоқтатады, ол циклді белгілі бір оқиға орын алған кезде немесе қосымша шарттарға байланысты аяқтау қажет болған жағдайда қолданылады; Continue операторы циклдің ағымдағы итерациясын тоқтатып, келесі итерацияны бастайды, бұл циклдің кейбір әрекеттерін өткізіп жіберуге мүмкіндік береді; Else блогы while циклінде цикл шарттың жалған болуынан тоқтаған жағдайда орындалады және бұл бөлім циклдің сәтті аяқталғанын немесе қосымша әрекеттерді орындау қажеттігін көрсету үшін қолданылады.

Есте сақта!

Итерация – Циклдің бір рет орындалуы.

Айнымалының өзгеруі – Циклдың орындалу шартын қамтамасыз ету үшін цикл айнымалысының мәні әр итерацияда өзгеруі керек.

Шексіз циклдерден сақтану – Егер цикл шарты ешқашан орындалмайтын болса, бағдарлама мәңгі қайталана береді.

While циклінің ерекшеліктері

Анықталмаған қайталану

While циклі алдын ала белгілі қайталану санын қажет етпейді. Цикл тек шартқа тәуелді жұмыс істейді. Бұл оны икемді етеді және әртүрлі есептерді шешуде қолдануға мүмкіндік береді.

Логикалық шарттардың рөлі

Шарт – бұл циклдің негізгі басқару механизмі. Шарттың дұрыс немесе қате берілуі цикл жұмысының нәтижесіне тікелей әсер етеді.

Шексіз циклдер

Егер шарт ешқашан False мәніне ауыспаса, цикл шексіз жалғасады. Бұл бағдарламаның тоқтауына және жүйелік ресурстардың сарқылуына әкелуі мүмкін.

Айнымалыны басқару

Цикл айнымалысының өзгеруі циклдің орындалу барысына әсер етеді. Айнымалының дұрыс өзгеруі циклдің тиімді орындалуын қамтамасыз етеді.

Python тілінде while циклі мәліметтерді өңдеу, күрделі алгоритмдерді іске асыру, жүйелерді модельдеу және бақылау сияқты әртүрлі есептерді шешуге мүмкіндік береді. Ол қайталанатын әрекеттерді ұйымдастырып, қажетті шарт орындалғанға дейін мәліметтермен жұмыс істеуді жалғастырады. While циклі жүйелік процестерді модельдеу мен бақылауда, пайдаланушының кірісін тексеруде және физикалық құбылыстарды модельдеуде кеңінен қолданылады. Бұл цикл икемділігі арқасында

үлкен көлемдегі мәліметтерді өңдеуде, шартқа негізделген қайталанатын есептеулерді реттеуде және серверлік процестерді ұйымдастыруда тиімді. While циклінің артықшылықтарына оның икемділігі, шарттарды қолдану

While циклінің артықшылықтары

- Икемділік: While циклі қайталану саны белгісіз немесе шарттарға тәуелді есептерде тиімді жұмыс істейді.
- Қолданудың қарапайымдылығы: Логикалық шарттарды қолдану арқылы циклді басқару жеңілдетіледі.
- Көп функционалдылық: Ол әртүрлі жағдайларда, мәліметтерді өңдеуден бастап, күрделі есептерді орындауға дейін қолданылады.
- Тиімділік: While циклі қайталанатын әрекеттерді автоматтандырып, уақытты үнемдеуге мүмкіндік береді.

While циклінің шектеулері

1. Шексіз циклдер қаупі
Егер циклдің аяқталу шарттары дұрыс анықталмаса, цикл шексіз орындалуы мүмкін. Бұл бағдарлама жұмысының тоқтауына және жүйе ресурстарының сарқылуына алып келеді.
2. Айнымалыны бақылау қажеттілігі
Циклдің дұрыс жұмыс істеуі үшін цикл айнымалысының өзгеруін мұқият қадағалау керек. Айнымалының қате өзгеруі шарттың орындалмауына әкеледі.
3. Логикалық қателіктер
Егер шарт дұрыс жазылмаса, цикл мүлдем орындалмай қалуы немесе қажетсіз әрекеттерді орындауы мүмкін.
4. Күрделі кодты түсіну қиындығы
Егер while цикліне тым көп шарттар мен операторлар қосылса, кодты түсіну мен басқару қиынға соғуы мүмкін.

While циклін тиімді қолдану үшін шарттарды нақты анықтап, айнымалының өзгеруін қадағалап, Break және Continue операторларын дұрыс пайдаланып, күрделі циклдерді модульдеп жазу ұсынылады. Бұл әдістер циклдің тиімділігін арттырып, кодтың құрылымын оңтайландырады. Python тіліндегі while циклі қайталанатын әрекеттерді басқару, күрделі есептерді шешу және процестерді автоматтандыруда маңызды рөл атқарады, бұл оны бағдарламалаудағы таптырмас құралдардың біріне айналдырады.

While циклі – Python бағдарламалау тілінің маңызды құралы. Ол логикалық шарттарға негізделген қайталанатын әрекеттерді басқару мүмкіндігін береді. While циклі мәліметтерді өңдеу, алгоритмдерді іске асыру, жүйелерді модельдеу сияқты көптеген салаларда қолданылады. Дұрыс ұйымдастырылған цикл бағдарламаның тиімділігін арттырып, кодтың құрылымын оңтайландырады. While циклін қолдану кезінде логикалық шарттарды дұрыс анықтау және циклдің орындалуын мұқият бақылау бағдарламаның тұрақты жұмыс істеуін қамтамасыз етеді. Бұл циклдің мүмкіндіктері оны бағдарламалаудағы маңызды құралдардың бірі етеді.

1.2.3. Цикл операторлары және функциялар

Цикл операторлары мен функциялар – Python бағдарламалау тілінің ең маңызды құралдарының бірі. Олар қайталанатын әрекеттерді автоматтандыруға, кодты қайта қолдануға және бағдарламаның құрылымын жақсартуға мүмкіндік береді. Бұл тарауда осы екі негізгі тақырыпқа теориялық тұрғыдан жан-жақты шолу жасалады.

Цикл операторлары

Циклдер – белгілі бір әрекеттерді қайталанатын түрде орындауға арналған құралдар. Python тілінде циклдер есептерді автоматтандыруға, мәліметтерді өңдеуге және қайталанатын процестерді оңай ұйымдастыруға мүмкіндік береді.

Циклдердің негізгі түрлері

While циклі – логикалық шарт ақиқат болғанға дейін қайталанатын әрекеттерді ұйымдастыруға арналған. Бұл цикл белгілі бір шартқа тәуелді әрекеттерді орындау қажет болғанда қолданылады.

For циклі – белгілі бір диапазон немесе реттелген мәліметтер жиыны бойынша қайталанатын әрекеттерді орындау үшін қолданылады. Бұл циклдің артықшылығы – оның нақты қадам санын алдын ала белгілеу мүмкіндігі.

Циклдерді басқару операторлары

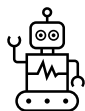
Циклдерді басқару операторлары бағдарламаның орындалу ретін тиімді бақылауға мүмкіндік береді. **Break операторы** циклді шартқа қарамастан мерзімінен бұрын тоқтатады және бұл қажетті нәтижеге тез жету үшін қолданылады. **Continue операторы** ағымдағы итерацияны өткізіп жіберіп, келесі қадамға өтуді қамтамасыз етеді, бұл циклдегі кейбір әрекеттерді айналып өтуге мүмкіндік береді. **Else блогы** цикл шарт орындалмай тоқтаған жағдайда жұмыс істейді және циклдің сәтті аяқталғанын немесе қосымша әрекеттер қажет екенін көрсету үшін пайдаланылады.

Циклдердің қолданылу салалары



Мәліметтерді өңдеу

Циклдер үлкен көлемдегі мәліметтермен жұмыс істеуде, мәліметтерді сұрыптауда, сүзуде және талдауда қолданылады.



Процесстерді автоматтандыру

Белгілі бір әрекеттерді қайталап орындау қажет болғанда циклдер тиімді шешім ұсынады. Бұл, әсіресе, пайдаланушының енгізген деректерімен жұмыс істегенде немесе үлкен есептеулерді орындағанда маңызды.



Алгоритмдерді жүзеге асыру

Күрделі есептерді шешу және логикалық операцияларды орындау үшін циклдер жиі пайдаланылады.

Функциялар

Функция – бұл белгілі бір тапсырманы орындау үшін жасалатын және қайта қолдануға болатын код блогы. Функциялар бағдарламаны логикалық блоктарға бөлуге, оның құрылымын жақсартуға және есептеулерді оңай басқаруға мүмкіндік береді.

Python тілінде функция – бұл кіріс ретінде белгілі бір деректерді қабылдап, оларды өңдеп, нәтиже қайтару үшін қолданылатын құрылым.

Функцияны анықтау үшін `def` кілттік сөзі қолданылады, ал оны қолдану үшін функция атауы және аргументтер тізімі беріледі.

Функцияның элементтері бағдарламалау тілдерінде оның құрылымын және жұмыс істеу ретін анықтайды. **Аты** функцияны анықтайтын және оны шақыру үшін қолданылатын атау болып табылады. **Аргументтер** функцияға берілетін кіріс деректерін білдіреді, олар арқылы функция қажетті есептеулерді орындайды. **Цикл денесі** функцияның негізгі логикасын құрайтын операторлар жиынынан тұрады және барлық есептеу процестері осы жерде орындалады. **Нәтиже (return)** функция орындалғаннан кейін қайтарылатын мәнді білдіреді, бұл функцияның жұмыс нәтижесін пайдалануға мүмкіндік береді.

Функцияның артықшылықтары

1. Қайта қолдану мүмкіндігі

Бір рет жазылған функцияны бірнеше рет қолдануға болады. Бұл кодтың қайталануын болдырмайды.

2. Кодты құрылымдау

Бағдарламаны логикалық блоктарға бөлу арқылы кодты түсіну мен басқаруды жеңілдетеді.

3. Оңай өзгерту мүмкіндігі

Функциядағы өзгерістер оның барлық шақыру орындарына автоматты түрде әсер етеді, бұл бағдарламаны оңтайлы түрде жаңартуға мүмкіндік береді.

4. Тиімділік

Функциялар бағдарламаның жалпы тиімділігін арттырады, себебі олар кодтың көлемін қысқартып, есептеулерді оңтайландырады.

Стандартты функциялар

Python тілі көптеген кіріктірілген функциялар жиынтығымен қамтамасыз етілген, бұл бағдарламашылардың жұмысын айтарлықтай жеңілдетеді. Ең жиі қолданылатын стандартты функцияларға мыналар жатады:

- `min()`: Тізімдегі немесе жиындағы ең кіші мәнді табады.
- `max()`: Ең үлкен мәнді табады.
- `len()`: Объектідегі элементтер санын анықтайды.
- `sum()`: Сандық мәндердің қосындысын есептейді.

Пайдаланушы анықтайтын функциялар

Python-да әзірлеуші өз қажеттіліктеріне сәйкес жаңа функциялар құра алады. Мұндай функциялар арнайы есептерді шешу үшін жасалады.

Функцияларды құру барысында олардың тиімділігі мен икемділігі үшін келесі ұсыныстарды ескерген жөн:

1. Функция атауын мағынасына сай қою.
2. Кіріс параметрлерін және қайтарылатын нәтижені нақты анықтау.
3. Функцияны қысқа әрі түсінікті ету.

Циклдер мен функциялардың өзара байланысы

Циклдер мен функцияларды бірге қолдану бағдарламаны тиімді ұйымдастыруға мүмкіндік береді. Мысалы:

- Циклдер қайталанатын әрекеттерді орындау үшін қолданылады.
- Функциялар осы әрекеттерді қайта қолданылатын блоктарға жинақтайды.

Мұндай байланыс бағдарламаның оқылымдылығын арттырып, күрделі есептерді оңай шешуге мүмкіндік береді.

Функция мен циклдердің өмірлік мысалдары

Циклдер мен функциялар бағдарламаның барлық салаларында қолданылады. Олардың ішінде:

- Деректер ғылымы: Мәліметтерді өңдеу, талдау және визуализация.
- Машиналық оқыту: Үлгілерді дайындау және талдау.
- Автоматтандыру: Ресурстарды басқару және күнделікті процестерді

жеңілдету.

- Веб-әзірлеме: Жауаптарды басқару және пайдаланушы сұрауларын өңдеу.

Циклдер мен функциялар Python бағдарламалау тілінің ажырамас бөлігі болып табылады. Олар күрделі есептерді шешу кезінде қолданылатын негізгі құралдар болып саналады. Циклдер қайталанатын әрекеттерді автоматтандырып, есептеу тиімділігін арттырса, функциялар кодты құрылымдап, оны түсінуді жеңілдетеді. Екі механизмді үйлесімді түрде пайдалану бағдарламалаудағы негізгі дағдыларды жетілдіріп, күрделі жобаларды жеңіл жүзеге асыруға мүмкіндік береді.