Python for AI

Дәуренов Ә.Д

Дәріс 3. Python тіліндегі циклдер және деректерді қайталама өңдеу

Дәрістің мақсаты

Python тіліндегі циклдердің жұмыс істеу принципін түсіну және қайталанатын операцияларды автоматтандыру үшін for, while, range() конструкцияларын қолдану логикасын меңгеру.

Дәрістен кейін студент:

- қайталанатын алгоритмнің не екенін;
- цикл және итерация ұғымдарын;
- for және while циклдерінің айырмашылығын;
- range() функциясының жұмысын;
- санауыш пен жинақтаушы айнымалыларды;
- break және continue операторларының қызметін;
- цикл арқылы бірнеше өлшеуді өңдеу принципін;
- циклдердің деректерді талдау және AI есептеріндегі маңызын түсінуі тиіс.

Слайд 1. Неліктен циклдер қажет?

1 Мәселе

Бізде қозғалтқыш температурасының бес өлшемі бар:



90
92
95
98
101

2 Циклсіз шешім (5 рет жазу)

Python

```python
print(90)
print(92)
print(95)
print(98)
print(101)
```

3 Циклмен шешім (қысқа және ыңғайлы)

Python

```python
for t in [90, 92, 95, 98, 101]:
    print(t)
```

4 Егер автомобильден:


10 өлшем


100 өлшем


1 000 өлшем

⋮

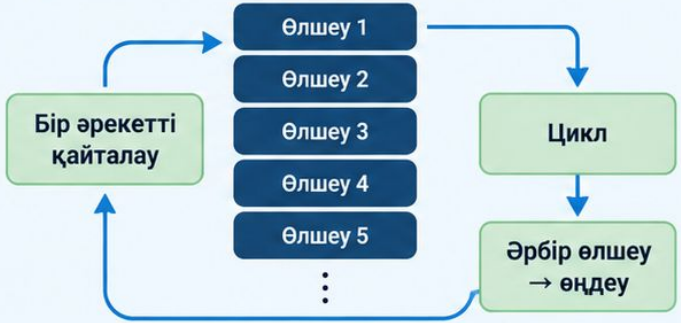

100 000 өлшем

5 Бір команданы мыңдаған рет жазу мүмкін емес

```
print(...)
print(...)
print(...)
...
print(...)
print(...)
print(...)
```

Мыңдаған жол код жазу тиімсіз!

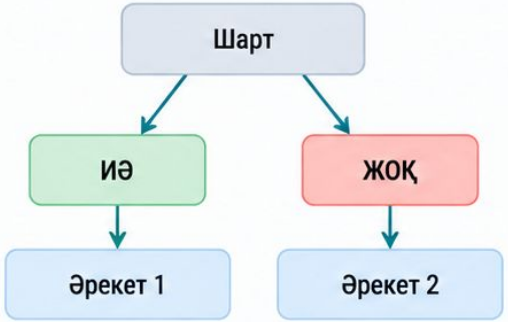
6 Цикл қолдану



Цикл — белгілі бір әрекеттерді бірнеше рет автоматты түрде орындауға мүмкіндік беретін алгоритмдік құрылым.

Слайд 2. Қайталанатын алгоритм

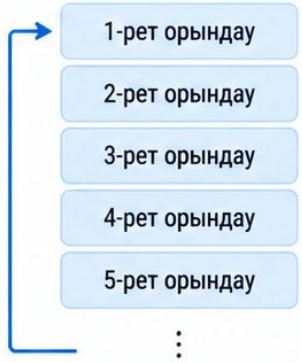
1 Тармақталған алгоритм (алдыңғы дәріс)



2 Қайталану идеясы



3 Қайталанатын алгоритмнің жұмыс істеуі



- Бір әрекет бірнеше рет орындалады.
- Қайталану саны белгілі немесе шарт бойынша аяқталады.
- Бұл процесс **цикл** деп аталады.

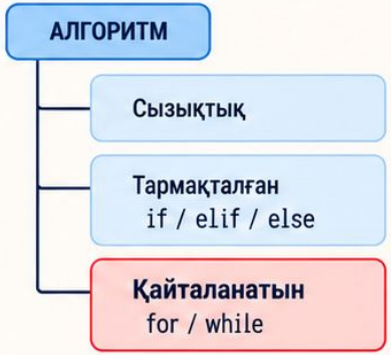
4 Қайталанатын алгоритмнің түрлері (Python)

Белгілі сан рет қайталау (for)	Шарт бойынша қайталау (while)	Қайталанатын әрекет (мысал)

`for i in range(5):`
`  print(i)`
	`i = 0`
	`while i < 5:`
	`  print(i)`
	`  i = i + 1`
		`for t in [90, 92, 95, 98, 101]:`
		`  print(t)`

Әрекет 5 рет орындалады.	Шарт True болғанша орындалады.	Тізімдегі барлық элементтерді шығару.

5 Жалпы құрылым



- Командалар бірінен кейін бірі орындалады.
- Шартқа байланысты әр түрлі әрекеттер.
- Бір әрекетті бірнеше рет орындау.

Слайд 3. Цикл және итерация

1 Негізгі ұғымдар

Цикл
Командалар блогын бірнеше рет орындау процесі.

Итерация
Цикл денесінің бір рет орындалуы.

2 Мысал: for циклі

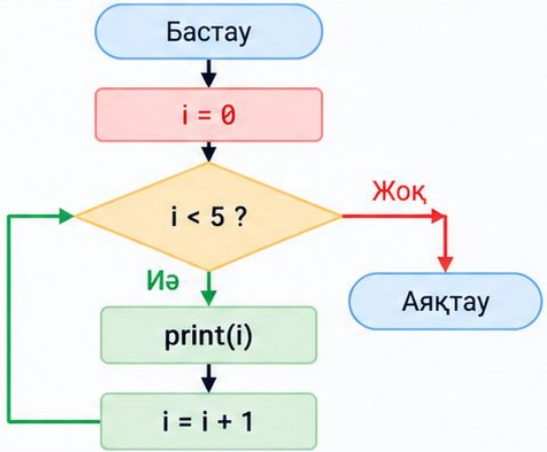
Python

```python
for i in range(5):
    print(i)
```

Нәтиже:

```
0
1
2
3
4
```

3 For циклінің блок-схемасы



4 Циклдің жұмыс істеуі (5 рет орындау)

1-итерация	2-итерация	3-итерация	4-итерация	5-итерация
`i = 0` `print(i)`	`i = 1` `print(i)`	`i = 2` `print(i)`	`i = 3` `print(i)`	`i = 4` `print(i)`
Экранға: 0	Экранға: 1	Экранға: 2	Экранға: 3	Экранға: 4

Итерация — циклдің **бір толық орындалуы**. Бұл мысалда цикл 5 рет орындалады (**5 итерация**).

Слайд 4. Python тіліндегі циклдердің негізгі түрлері

1 Циклдер дегеніміз не?

Цикл — бір әрекетті бірнеше рет автоматты түрде орындауға мүмкіндік беретін алгоритмдік құрылым.

2 Python тіліндегі циклдердің түрлері



3 for циклі

Қайталанатын элементтер немесе қайталану саны бепгілі болғанда қолданылады.

Python

```
1 for i in range(5):
2     print(i)
```

Нәтиже:

```
0
1
2
3
4
```

4 for цикліың жұмыс істеу процесі

1-итерация	2-итерация	3-итерация	4-итерация	5-итерация
`i = 0`	`i = 1`	`i = 2`	`i = 3`	`i = 3`
`print(i)`	`print(i)`	`print(i)`	`print(i)`	`print(i)`
↓	↓	↓	↓	↓
Экранға: 0	Экранға: 1	Экранға: 2	Экранға: 3	Экранға: 4

5 while циклі

Қайталау бепгілі бір шартқа тәуелді болғанда қолданылады.

Python

```
1 i = 0
2 while i < 5:
3     print(i)
4     i = i + 1
```

Нәтиже:

```
0
1
2
3
4
```

✓ `for` — элементтер немесе қайталану саны бепгілі болғанда. | `while` — шарт ақиқат болғанша.

Слайд 6. range() функциясы

1 range() не істейді?

`range()` функциясы

0-ден бастап, берілген санға дейін (сол санды қоспай) сандардың қатарын құрайды.

2 Негізгі жазылуы

Python

```
range(5)
```



3 Мысал

Python

```
for i in range(5):
    print(i)
```

Нәтиже:

```
0 1 2 3 4
```

4 Сипаттамасы

`range(5)`

- 0-ден басталады.
- **5 саны кірмейді.**
- Нәтижесінде 5 сан шығады: 0, 1, 2, 3, 4.

Маңызды:
`range(5)` → 0-ден басталады және **5 санын қоспайды.**

Слайд 7. range(start, stop, step) функциясы

1 Жалпы жазылуы

Python

```
range(start, stop, step)
```

- start – бастапқы мән (қосылады)
- stop – соңғы шекара (қоспайды)
- step – қадам (әр жолғы өсім)

2 Мысал: диапазон және қадам

Python

```
for i in range(2, 10, 2):
    print(i)
```

Нәтиже:

```
2
4
6
8
```

3 Параметрлердің мағынасы



4 Көрнекі түрде



`range(start, stop, step)` үш параметр қабылдайды. Осы мысалда 2-ден бастайды, 10 санын қоспайды және әр жолы 2-ге өседі.

Слайд 8. Циклдың орындалу логикасы

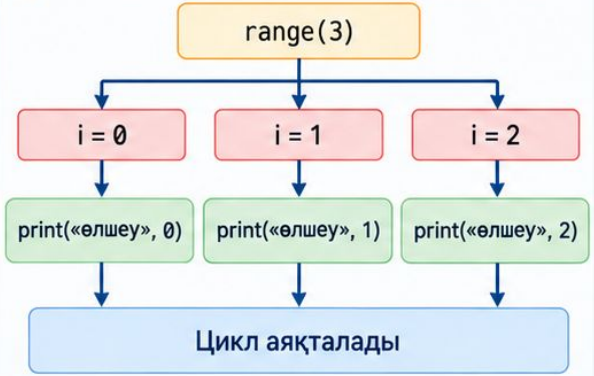
1 Мысал коды

Python

```
for i in range(3):
    print("өлшеу", i)
```

`i` 0-ден 2-ге дейінгі мәндерді қабылдайды (3 сан).

2 Орындалу логикасы



3 Нәтиже

Экранға шығатын нәтиже:

```
өлшеу 0
өлшеу 1
өлшеу 2
```

`range(3)` 0, 1, 2 мәндерін қосады (3 сан).

4 Инғайлы мысалы

Python

```
for i in range(1, 4):
    print("өлшеу", i)
```

Нәтиже:

```
өлшеу 1
өлшеу 2
өлшеу 3
```

`range(1, 4)` → 1, 2, 3 мәндерін береді.

Слайд 9. Цикл тек сандар үшін емес

1 Идеясы

Циклдың негізгі идеясы — **әр элементті кезекпен алу.**

`range()` тек сандар үшін емес, тізімдер, жолдар, және басқа итерацияланатын объектілер үшін де қолданылады.

2 Тізіммен жұмыс (сандар)

Python

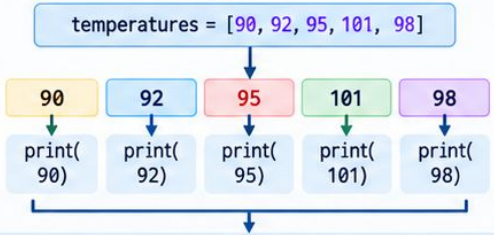
```
temperatures = [90, 92, 95, 101, 98]

for temperature in temperatures:
    print(temperature)
```

Нәтиже:

```
90
92
95
101
98
```

3 Орындалу логикасы



4 Нәтиже және түсініктеме

Нәтиже:

```
90
92
95
101
98
```

Бұл цикліміз тізімдегі элементтерді бірінан соң бірін (кезекпен) алып, өңдейді. Әр итерацияда `temperature` өзгермелісі тізімнің келесі мәнін алады.

Слайд 10. for + if

1 Мәселе қойылуы

Тізімдегі әр температураны тексеру қажет:
егер температура **100-ден** үлкен болса — "WARNING", әйтпесе — "NORMAL".

2 Python коды

Python

```python
temperatures = [90, 92, 105, 98, 110]

for temperature in temperatures:
    if temperatypa > 100:
        print(temperature, "WARNING")
    else:
        print(temperature, "NORMAL")
```

3 Блок-схема



4 Әр итерацияның орындалуы (temperature мәндері бойынша)

1-итерация	2-итерация	3-итерация	4-итерация	5-итерация
temperature = 90	temperature = 92	temperature = 105	temperature = 98	temperature = 110
90 > 100?	92 > 100?	105 > 100?	98 > 100?	110 > 100?
Жоқ	Жоқ	Иә	Жоқ	Иә
print(90, "NORMAL")	print(92, "NORMAL")	print(105, "WARNING")	print(98, "NORMAL")	print(110, "WARNING")

5 Нәтиже

Экранға шығатын нәтиже:

```
90 NORMAL
92 NORMAL
105 WARNING
98 NORMAL
110 WARNING
```

Қорытынды — `if` шешімі әр элемент үшін жеке орындалады, ал `for` барлық элементтерді кезекпен өңдейді.

Слайд 11. Санаушы — Counter

1 Мәселе

Тізімдегі температуралардың ішінен 100 °C-тан жоғары болған мәндер санын анықтау керек.

2 Python коды

Python

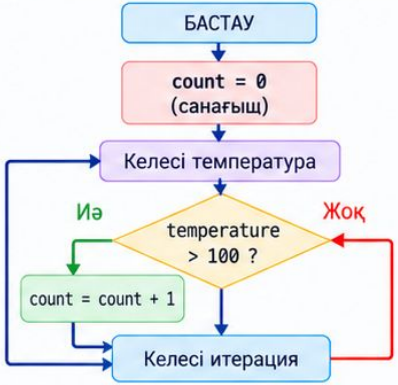
```python
temperatures = [90, 105, 95, 110, 98]

count = 0

for temperature in temperatures:
    if temperature > 100:
        count = count + 1

print(count)
```

3 Жұмыс логикасы



4 Орындау кестесі

Итерация	temperature	temperature > 100	count
1	90	False	0
2	105	True	1
3	95	False	1
4	110	True	2
5	98	False	2

5 Әр итерацияның орындалуы

1-итерация	2-итерация	3-итерация	4-итерация	5-итерация
temperature = 90	temperature = 105	temperature = 95	temperature = 110	temperature = 98
90 > 100? → Жоқ	105 > 100? → Иә	95 > 100? → Жоқ	110 > 100? → Иә	98 > 100? → Жоқ
count = 0 (өзгермейді)	count = 1 (көбейеді)	count = 1 (өзгермейді)	count = 2 (көбейеді)	count = 2 (өзгермейді)

6 Нәтиже

Python

```python
temperatures = [90, 105, 95, 110, 98]
print(count)
```

Экранға шығатын нәтиже:

2

Тізімде 100 °C-тан жоғары болған 2 температура бар (105 және 110).

Қорытынды: Циклды пайдаланып, белгілі шартты қанағаттандыратын элементтердің санын есептеу үшін **санағыш (counter)** қолданылады.

Слайд 12. Жинақтаушы — Accumulator

1 Жинақтаушы дегеніміз не?

Жинақтаушы (accumulator) — цикл ішінде мәндерді біртіндеп жинауға арналған айнымалы.

Негізгі идея:

- Алдымен жинақтаушыны бастапқы мәнмен орнатамыз (көбіне 0).
- Цикл әр итерацияда ағымдағы мәнді жинақтаушыға қосады.

2 Python коды

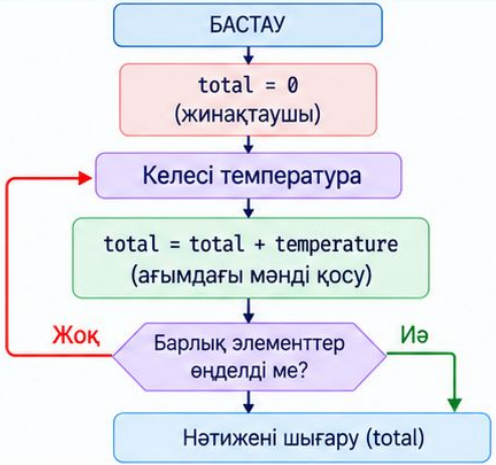
Python

```python
temperatures = [90, 92, 94]
total = 0

for temperature in temperatures:
    total = total + temperature

print(total)
```

3 Жұмыс логикасы (блок-схема)



4 Әр қадамның орындалуы

Итерация	temperature	Есептеу	total
Бастапқы	–	total = 0	0
1-итерация	90	0 + 90	90
2-итерация	92	90 + 92	182
3-итерация	94	182 + 94	276

5 Нәтиже

Python

```python
temperatures = [90, 92, 94]
total = 0

for temperature in temperatures:
    total = total + temperature

print(total)
```

Экранға шығатын нәтиже:

276

Жинақтаушы `total` айнымалысы тізімдегі барлық мәндердің қосындысын сақтайды.

Қорытынды: Жинақтаушы — цикл ішінде мәндерді біртіндеп жинау үшін қолданылатын айнымалы.

Слайд 14. while циклі

1 Негізгі түсінік

while циклі — шарт True болғанға дейін командалар блогын қайта-қайта орындайды.

Жалпы құрылымы:

```
while шарт:
    әрекет
```

2 Мысал коды

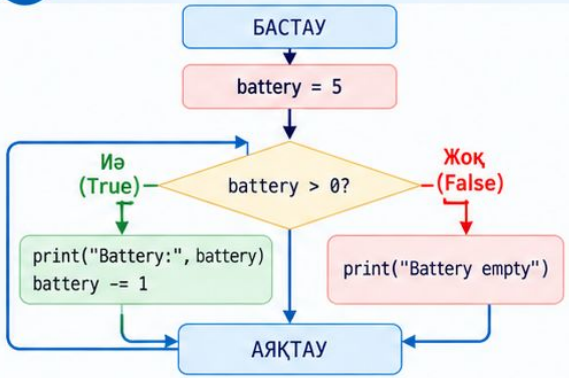
Python

```
battery = 5

while battery > 0:
    print("Battery:", battery)
    battery -= 1

print("Battery empty")
```

3 Жұмыс логикасы (блок-схема)



4 Нәтиже

Экранға шығатын нәтиже:

```
Battery: 5
Battery: 4
Battery: 3
Battery: 2
Battery: 1
Battery empty
```

`battery -= 1` означает **уменьшить значение переменной `battery` на 1**.
battery = battery - 1

5 Әр итерацияның орындалуы

Бастапқы күй	2-итерация	3-итерация	4-итерация	5-итерация	6-итерация
battery = 5	battery = 4	battery = 3	battery = 2	battery = 1	battery = 0
5 > 0 → Иә	4 > 0 → Иә	3 > 0 → Иә	2 > 0 → Иә	1 > 0 → Иә	0 > 0 → Жоқ
print("Battery:", 5) battery -= 1	print("Battery:", 4) battery -= 1	print("Battery:", 4) battery -= 1	print("Battery:", 3) battery -= 1	print("Battery:", 1) battery -= 1	print("Battery empty")
Жаңа мән: battery = 4	Жаңа мән: battery = 3	Жаңа мән: battery = 2	Жаңа мән: battery = 1	Жаңа мән: battery = 0	

Қорытынды: `while` циклі шарт True болғанша блокты қайталайды. Бұл мысалда battery мәні әр итерацияда 1-ге азаяды және 0 болғанда цикл тоқтайды.

battery -= 1 - **battery айнымалысының мәнін 1-ге азайту** дегенді білдіреді.

battery = battery - 1

battery -= 1 бұл жерде заряд мәнін біртіндеп азайту үшін қажет, нәтижесінде while циклі белгілі бір уақытта аяқталады.

Слайд 15. while қалай жұмыс істейді?

1 Идеясы

while циклі шарт ақиқат болғанша командаларды қайталап орындайды.

Логика:

1. Шартты тексеру
2. Шарт True болса — цикл денесін орындау
3. Қайтадан шартты тексеру
4. Шарт False болса — цикл тоқтайды

2 Блок-схема



3 Мысал коды

Python

```
battery = 5

while battery > 0:
    print("Battery:", battery)
    battery -= 1

print("Battery empty")
```

4 Нәтиже

Экранға шығатын нәтиже:

```
Battery: 5
Battery: 4
Battery: 3
Battery: 2
Battery: 1
Battery empty
```

Слайд 16. for және while айырмашылығы

1 Негізгі айырмашылықтар

Критерий	for	while
Элементтер бойынша қайталайды	Иә, белгілі диапазон немесе жинақ элементтері бойынша	Жоқ, шартқа негізделеді
Қайталану саны алдын ала белгілі	Иә	Жоқ, белгісіз болуы мүмкін
`range()` жиі қолданылады	Иә	Жоқ (міндетті емес)
Деректерді ретімен өңдеуге ыңғайлы	Иә	Жоқ, белгілі бір жағдай орын алғанша жұмыс істеуге ыңғайлы

2 Мысалдар

for — белгілі сан рет қайталану

Python

```
for i in range(10):
    print("Өлшеуді өңдеу")
```

Нәтиже:

```
(10 рет)
...
```

while — шарт орындалғанша

Python

```
temperature = 30
while temperature > 0:
    print("Температура қалыпты
       мәнге түскенше бақылау")
    temperature -= 1
```

Нәтиже:

```
(шарт орындалғанша)
...
```

Қорытынды: `for` белгілі сан рет немесе элементтер бойынша, ал `while` шарт ақиқат болғанша қайталайды.

Слайд 17. Шексіз цикл — Infinite Loop

1 Шексіз цикл дегеніміз не?

Шарт әрқашан True болып отырады, сондықтан циклл ешқашан тоқтамайды.

Мысал:

- x = 1
- x < 5 әрқашан True (x өзгермейді)
- Цикл мәңгі қайталанады.

2 Python коды (қате мысал)

Python

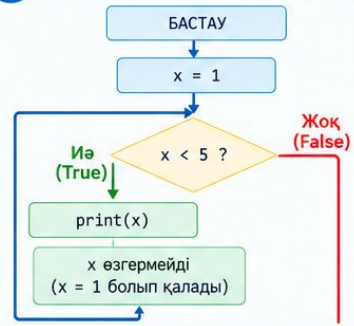
```
x = 1

while x < 5:
    print(x)
```

Неге шексіз?

- x мәні өзгермейді.
- x < 5 әрқашан True.
- Цикл тоқтамайды (мәңгі қайталанады).

3 Блок-схема



4 Нәтиже (үзінді)

```
1
1
1
1
1
1
1
...
```

Цикл тоқтамайды!

Қорытынды: `while` циклінде шарт `False` болмайтын болса, цикл шексіз қайталанады.

Слайд 18. break операторы

1 break дегеніміз не?

break — цикл ішіндегі жұмысты бірден тоқтататын оператор.

Қолданылатын жағдай:

- Белгілі бір шарт орындалғанда циклді тоқтату.
- Әсіресе `while` циклдерінде шексіз циклден шығу үшін қолданылады.

2 Python коды (мысал)

Python

```
temperatures = [90, 95, 102, 110, 98]

for temperature in temperatures:
    if temperature > 100:
        print("WARNING")
        break
```

Неге `break` қолданылады?

- `temperature > 100` шарты орындалған кезде "WARNING" шығарып,
- цикл бірден тоқтайды.

3 Блок-схема



4 Орындалу мысалы

Итерация	temperature	> 100 ?	Әрекет
1	90	False	жалғастыру
2	95	False	жалғастыру
3	102	True	"WARNING" және цикл тоқтайды
4	110	–	орындалмайды
5	98	–	орындалмайды

Қорытынды: `break` — циклдің орындалуын бірден тоқтатады. Бұл әсіресе `while` циклдерінде шексіз циклден шығу үшін өте пайдалы.

Слайд 19. continue операторы

1 continue дегеніміз не?

continue — циклдің ағымдағы итерациясын тоқтатып, келесі итерацияға бірден өткізетін оператор.

Негізгі идея:

- Шарт ақиқат болғанда, сол итерациядағы қалған код орындалмайды.
- Цикл тоқтамайды, келесі итерацияға өтеді.
- Әдетте, қажет емес элементтерді өткізу үшін қолданылады.

2 Python коды (мысал)

Python

```python
values = [90, -1, 95, 100]

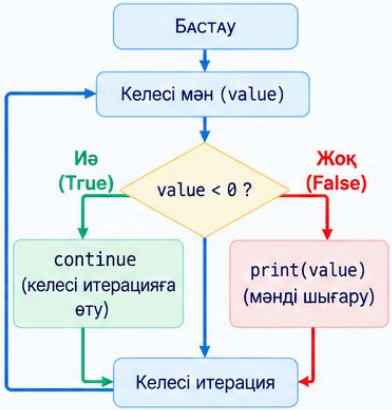
for value in values:
    if value < 0:
        continue

    print(value)
```

Логикасы:

1. Әр мәнді алып қарайды.
2. `value < 0` болса, `continue` келесі итерацияға өтеді.
3. `value ≥ 0` болса, `print(value)` орындалады.

3 Блок-схема



4 Нәтиже

Экранға шығатын нәтиже:

```
90
95
100
```

-1 өңделмеді, өйткені:

- `-1 < 0` шарты ақиқат.
- `continue` орындалады.
- `print(-1)` орындалмайды.
- Цикл келесі итерацияға өтеді.

5 Итерациялар бойынша орындалуы

Итерация	value	value < 0 ?	Әрекет	Нәтиже
1	90	False	print(90)	90
2	-1	True	continue (print орындалмайды)	—
3	95	False	print(95)	95
4	100	False	print(100)	100

6 Визуалды айырмашылық

break (циклді толық тоқтатады)

```python
for value in values:
    if value < 0:
        break
    print(value)
```

Шарт орындалғанда цикл толық тоқтайды. Келесі элементтер өңделмейді.

continue (келесі итерацияға өтеді)

```python
for value in values:
    if value < 0:
        continue
    print(value)
```

Шарт орындалғанда тек сол итерация өткізіледі, цикл жалғасып, келесі элементтер өңделеді.

Қорытынды: `continue` — циклдегі ағымдағы итерацияны өткізіп, келесі итерацияға бірден өтуді қамтамасыз ететін оператор.

Слайд 22. Циклдердің Data Science және AI-дегі рөлі

1 Неліктен циклдер маңызды?

Циклдер Data Science және AI бағытында көп рет қайталанатын операцияларды автоматтандырып, тиімді орындауға мүмкіндік береді.

Негізгі артықшылықтары:

- Уақытты үнемдейді
- Қателікті азайтады
- Үлкен көлемдегі деректермен жұмыс істеуге мүмкіндік береді
- Талдау, өңдеу, модельдеу қадамдарын жүйелейді

2 Мысал коды (Data Science)

Python

```python
temperatures = [90, 92, 95, 101, 98]

for temperature in temperatures:
    if temperature > 100:
        print("WARNING")
```

Бұл код не істейді?

- `temperatures` тізіміндегі әр мәнді қарайды.
- Егер температура 100-ден жоғары болса, `"WARNING"` шығарады.
- Нақты жобаларда мұндай логика бақылау, сүзу, тазарту (cleaning) т.б. үшін қолданылады.

3 Деректерді өңдеу пайплайны



4 Нақты қолдану мысалдары

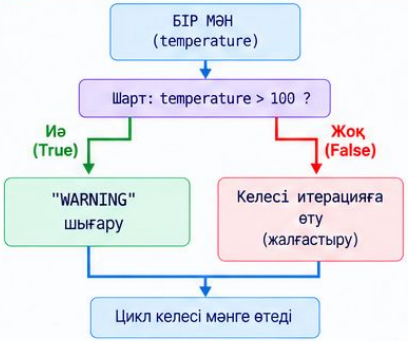
Деректерді оқу
Көп файлдарды цикл арқылы жүктеу (CSV, Excel, JSON).

Деректерді тазалау
Әр жазбаны тексеру, қателерді жою, бос мәндерді өңдеу.

EDA (зерттеу)
Бағандар бойынша статистиканы есептеу, графиктерді құру.

Модель оқыту
Әр параметрмен бірнеше модельді қайта-қайта оқыту (hyperparameter tuning).

5 Циклдің жұмыс логикасы



6 Кеңейтілген мысал (өмірге жақын)

Python

```python
import pandas as pd

# Температура деректері бар файл деп елестетейік
temperatures = [90, 95, 102, 110, 98]

high_temps = []
for temp in temperatures:
    if temp > 100:
        print(f"WARNING: {temp}")
        high_temps.append(temp)

print("100-ден жоғары температуралар:", high_temps)
```

7 Нәтиже

Экранға шығатын нәтиже:

```
WARNING: 102
WARNING: 110
100-ден жоғары температуралар: [102, 110]
```

Бұл нені көрсетеді?

- Цикл деректерді қарап шығып, шарт орындалған мәндерді тапты.
- Мұндай тәсіл үлкен датасеттерде де дәл осылай жұмыс істейді.
- Нәтижелер тізімге жиналып, кейінгі талдауға қолданылады.

8 Циклдердің кең қолданылуы

Файлдармен жұмыс
Көптеген файлдарды оқу және өңдеу.

Статистикалық есептеулер
Әр баған бойынша мәндерді есептеу.

Модель параметрлерін іздеу
Көп параметрмен модельді қайта-қайта оқыту.

Нәтижелерді сақтау
Әр итерацияның нәтижелерін тізімге немесе файлға жазу.

Қорытынды: | Циклдер Data Science және AI жобаларында деректерді өңдеуді, талдауды және модельдеуді автоматтандыратын негізгі құрал.