КОММЕРЦИЯЛЫҚ ЕМЕС АКЦИОНЕРЛІК ҚОҒАМЫ
«ӘБІЛҚАС САҒЫНОВ АТЫНДАҒЫ ҚАРАҒАНДЫ ТЕХНИКАЛЫҚ УНИВЕРСИТЕТІ»

Киберқауіпсіздік және жасанды интеллект кафедрасы

Аубакирова А.Е.

«PYTHON ОБЬЕКТІГЕ БАҒЫТТАЛҒАН БАҒДАРЛАМАЛАУ»

ПӘНІ БОЙЫНША
ДӘРІСТЕРГЕ
ӘДІСТЕМЕЛІК НҰСҚАУЛАР

Қарағанды 2026

КОММЕРЦИЯЛЫҚ ЕМЕС АКЦИОНЕРЛІК ҚОҒАМЫ
«ӘБІЛҚАС САҒЫНОВ АТЫНДАҒЫ ҚАРАҒАНДЫ ТЕХНИКАЛЫҚ УНИВЕРСИТЕТІ»

Киберқауіпсіздік және жасанды интеллект кафедрасы

Аубакирова А.Е.

«PYTHON ОБЬЕКТІГЕ БАҒЫТТАЛҒАН БАҒДАРЛАМАЛАУ»

ПӘНІ БОЙЫНША
ДӘРІСТЕРГЕ
ӘДІСТЕМЕЛІК НҰСҚАУЛАР

6B06105 «Data Science» білім беру бағдарламасы

Қарағанды 2026

ДӘРІС КОНСПЕКТІЛЕРІ

Тақырып 4. Логикалық операторлар

Кейде бір уақытта бір емес, бірнеше шартты тексеру қажет. Мысалы, санның жұп екенін тексеруге болады (N % 2 == 0) (n-ді 2-ге бөлудің қалдығы 0-ге тең). Егер n және m екі бүтін сандардың жұп екенін тексеру қажет болса, екі шарттың да әділдігін тексеру қажет: n % 2 == 0 және m % 2 == 0. Ол үшін оларды and (логикалық n) логикалық операторының көмегімен біріктіру керек: n % 2 == 0 and m % 2 == 0.

Питонда стандартты логикалық операторлар бар: логикалық көбейту И, логикалық қосу ИЛИ, логикалық теріске шығару НЕ.

Логикалық көбейту И екілік оператор болып табылады (яғни екі операндты оператор: сол және оң) және and формасына ие. And операторы True мәнін қайтарады, егер оның екі операндында да True мәні болса.

Логикалық қосу ИЛИ екілік оператор болып табылады және кем дегенде бір Операнд True болғанда ғана True мәнін қайтарады. Оператор логикалық ИЛИ or түрінде болады.

Логикалық емес (теріске шығару) унарлы (яғни бір операндпен) оператор болып табылады және not формасына ие, содан кейін жалғыз операнд. Егер операнд жалған болса және керісінше болса, логикалық НЕ шын мәнін қайтармайды.

Мысал.

a немесе b сандарының кем дегенде біреуі 0-ге аяқталатынын тексерейік:

```
a = int(input())
b = int(input())
if a % 10 == 0 or b % 10 == 0:
   print('YES')
else:
   print('NO')
```

a санының оң, ал b санының теріс емес екенін тексерейік:

```
if a > 0 and not (b < 0):
```

Not (B < 0) орнына (b >= 0) жазуға болады.

Шарттарға қарапайым бағдарламаларды қарастырыңыз.

Цикл операторлары

Циклдар құрылымдық бағдарламалаудың шартты операторлар сияқты маңызды бөлігі болып табылады. Циклдардың көмегімен код бөлімдерінің қайталануын ұйымдастыруға болады. Бұл қажеттілік жиі туындайды.

Мысалы, пайдаланушы сандарды дәйекті түрде енгізеді және олардың әрқайсысы жалпы сомаға қосылуы керек. Немесе экранға 1-ден 100-ге дейінгі бүтін сандардың квадраттарын шығару керек.

While цикл операторы

While-Python-дағы ең әмбебап циклдердің бірі. Оператор цикл шартының мәні болғанша цикл денесін орындайды.

```
i = 5
while i < 15:
   print(i)
   i = i + 2
```

Шығады:
5
7
9
11
13
While операторында оның денесі аяқталғаннан кейін бағдарлама цикл тақырыбына оралып, шартты қайтадан тексереді. Егер логикалық өрнек шындықты қайтарса, онда цикл денесі қайтадан орындалады. Содан кейін біз қайтадан тақырыпқа ораламыз және т.б.
Цикл өз жұмысын тақырыптағы логикалық өрнек өтірікті қайтарған кезде ғана аяқтайды, яғни циклды орындау шарты енді орындалмайды. Осыдан кейін цикл блогының астында орналасқан операторлар орындалады. Олар "циклден шығу бар" дейді.
While циклімен екі ерекше жағдай болуы мүмкін:
1. Егер циклге бірінші рет кірген кезде логикалық өрнек жалған болса, онда цикл денесі бір рет орындалмайды. Бұл жағдайды қалыпты деп санауға болады, өйткені белгілі бір жағдайларда бағдарламаның логикасы цикл денесінің өрнектерін орындаудың қажеті жоқ деп болжауы мүмкін.
2. Егер while тақырыбындағы логикалық өрнек ешқашан жалғандықты қайтармаса, бірақ әрқашан шындыққа тең болса, онда цикл ешқашан аяқталмайды, егер оның денесінде циклден мәжбүрлі шығу операторы болмаса (break - үзіліс) немесе бағдарламадан шығу функцияларына қоңырау шалу – Python жағдайында quit (), exit (). Егер цикл шексіз рет қайталанса, онда бағдарламада цикл болады. Осы уақытта ол қатып қалады және өзін-өзі аяқтай алмайды.
Мысалды қарастырайық:

```
x1 = 100
i = 0
while i < 5:
    n = int(input())
    x1 = x1 - n
    i = i + 1
print("Қалды", x1)
```

For цикл операторы
Python-да цикл for кілт сөзінен басталады, содан кейін айнымалының ерікті атауы, for циклі мәндер жиынтығында жүгіреді, әр мәнді айнымалыға орналастырады, содан кейін циклде біз осы айнымалымен әр түрлі әрекеттерді жасай аламыз. Жалпы синтаксис for...in Python да келесідей:

```
for айнымалы in мәндер жиынтығы:
    операторлар
```

Цикл орындалған кезде Python жиынтықтағы барлық мәндерді дәйекті түрде алады және олардың айнымалысын береді. Жиынтықтағы барлық мәндер қайталанғаннан кейін цикл өз жұмысын аяқтайды.

Мәндер жиынтығы ретінде, мысалы, таңбалар жиынтығын білдіретін жолды қарастыруға болады.
Мысалды көрейік:

```
message = "Hello"
for x in message:
   print(x)
```

Цикл x айнымалысын анықтайды, in операторынан кейін "Hello"жолын сақтайтын message айнымалысы қайталанатын жиын ретінде көрсетіледі. Нәтижесінде, for циклі message жолындағы барлық таңбаларды дәйекті түрде сұрыптап,оларды x айнымалысына орналастырады.

Range () функциясы

Range () функциясы сандар тізбегін құру үшін қолданылады. Егер біз range(10) тапсырсақ, ол 0-ден 9-ға дейінгі сандарды жасайды. Range () функциясының синтаксисі төменде келтірілген.

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

- Start итерацияның басталуын білдіреді.
- Stop цикл stop-1-ге дейін қайталанатынын білдіреді. range (1,5) 1-ден 4 итерацияға дейінгі сандарды жасайды. Бұл қосымша параметр.
- size итерацияда белгілі бір сандарды өткізіп жіберу үшін қолданылады. Оны пайдалану міндетті емес. Әдепкі бойынша, қадам өлшемі 1-ге тең.

Келесі мысалдарды қарастырыңыз:

Мысал 1.

Сандарды ретімен басып шығаруға арналған бағдарлама.

```
for i in range(10):
      print(i,end = ' ')
```

Мысал 2.

Берілген Сан үшін көбейту кестесін басып шығаруға арналған бағдарлама.

```
n = int(input("Enter the number "))
for i in range(1,10):
   c = n*i
   print(n,"*",i,"=",c)
```

Бағдарламаны орындағаннан кейін біз аламыз:

```
Enter the number 5
5 * 1 = 5
5 * 2 = 10
5 * 3 = 15
5 * 4 = 20
5 * 5 = 25
5 * 6 = 30
5 * 7 = 35
5 * 8 = 40
```

5 * 9 = 45

Мысал 3.
Range () ішіндегі қадам өлшемін пайдаланып жұп санды басып шығаруға арналған бағдарлама.

```
n = int(input("Enter the number "))
for i in range(2,n,2):
    print(i)
```

For циклі while циклінен әлдеқайда жылдам. Бұл нұсқаулық цикл денесін бірнеше рет орындайды.
Мысалы:
1-ден k-ге дейінгі бүтін сандардың қосындысын табыңыз

```
sum = 0
k=int(input('введите к  '))
for i in range(1, k+1):
    sum = sum + i
print(sum)
```

Итерация қадамын 2-ге қою үшін жазыңыз

```
for n in range(1, k+1, 2):
```

Теориядан қысқаша мәліметтер

Жол таңбалар тізбегі деп аталады: әріптер, сандар, тыныс белгілері және т.б. жолды белгілеу үшін апострофты (') немесе қос тырнақшаларды қолдануға болады. Осының арқасында сіз апострофтар арқылы белгіленген жолдың ішінде тырнақшаға ие бола аласыз. Мысалы, " ол "сәлем " деді'..

Python 3-те барлық жолдарда юникод бар. Егер Python3-те сізге байт тізбегі қажет болса, (b) таңбасы қолданылады: b"бұл байт жолы".

Основные функции и методы работы со строками.

Функция / әдіс	**Сипаттамасы**
S1 + S2	Байланыс (жолдарды қосу)
S1 * n	Жолды қайталау N рет
S[i]	Индекс бойынша өтініш. S-жол айнымалысының атауы.
S[i:j:step]	Кесуді алу. S-жол айнымалысының атауы.
len(S)	Жолдың ұзындығы. S-жол айнымалысының атауы.
S.find(str, start,end)	Жолдағы ішкі жолды іздеу. Бірінші енгізу нөмірін немесе -1 қайтарады. Бастау және аяқтау параметрлері міндетті емес.

S.replace(шаблон, замена)	Үлгіні ауыстыру. Ескі ішкі жолдың барлық кірістерін жаңасына ауыстырады.
S.split(separator)	Жолды бөлгіш бойынша бөлу. Көрсетілген арнайы таңбаны пайдаланып жолды бөледі және ішкі жолдар тізімін қайтарады.
S.upper()	Жолды жоғарғы регистрге түрлендіру
S.lower()	Жолды төменгі регистрге түрлендіру

Мысал, s1= “Hello “, s2=”Maks”
Y = s1+s2. Y = “Hello Maks” аламыз

Python түріне түрлендіру

Кейде бүтін санды жол ретінде немесе керісінше жазу пайдалы. Егер жол сандардан тұрса, онда сіз бұл жолды сан түрінде ұсына аласыз, сонда онымен арифметикалық амалдарды орындауға болады. Ол үшін атауы тип атауына сәйкес келетін функциялар қолданылады, яғни int, float, str.

Мысалы, int ('123') 123 бүтін санды қайтарады, str (123) '123' жолын қайтарады, ал келесі Нұсқаулық:

print(str(2 + 2) * int('2' + '2'))

22 рет қайталанған "4" таңбасын көрсетеді.

Жұмысты орындау тәртібі

1. Теориялық материалды зерттеу.
2. Тапсырмаларды орындау.
3. Бақылау сұрақтарына жауап беріңіз.

1 тапсырма

Пайдаланушыдан сұрайтын бағдарламаны жазыңыз:
- Аты-жөні ("сіздің аты-жөніңіз?")
- жасы ("сіз неше жастасыз?")
- тұрғылықты жері ("Сіз қайда тұрасыз?")

Осыдан кейін мен үш жолды көрсетер едім:
"Сіздің атыңыз"
"Сіздің жасыңыз"
"Сіз өмір сүресіз"

2 тапсырма

1. Пернетақтадан 5 рет енгізілген сөзді қайталау бағдарламасын жасаңыз.
2. Сөздің і әрпін шығару бағдарламасын жасаңыз. І Индекс пернетақтадан енгізіледі.
3. Пернетақтадан енгізілген жолдың ұзындығын анықтаңыз.

4. Жолды кесу туралы нұсқаулықты зерттеңіз. Зерттеу нәтижелерін бағдарлама мысалдарымен жазыңыз.
5. Жолдағы ішкі жолды іздеу нұсқаулығын зерттеңіз. Зерттеу нәтижелерін бағдарлама мысалдарымен жазыңыз.
6. Жолды жоғарғы регистрге аудару бағдарламасын жасаңыз.

3 тапсырма

1. Пернетақтадан енгізілген сөйлемдегі сөздердің санын анықтаңыз. Сөздердің арасында тек бір бос орын бар.
2. Пернетақтадан енгізілген сөйлемде берілген сөзді іздеу бағдарламасын жасаңыз.
3. Пернетақтадан енгізілген жолда барлық Қос нүктелерді (:) сұрақ белгісімен (?).

Есеп мазмұны

1. Титулдық бет.
2. Орындалған жұмыстардың нәтижелері.
3. Бақылау сұрақтарына жауаптар.

Бақылау сұрақтары

1. Python-да жолдардың қандай түрі (типы) бар?
2. Python-да жолдар қалай белгіленеді?
3. Python-да байланыстыру әрекетін қалай орындауға болады?
4. Python-да І-ші таңбаны жолдан қалай шығаруға болады?
5. Python-да бүтін санды жолға түрлендіруге мысал келтіріңіз.