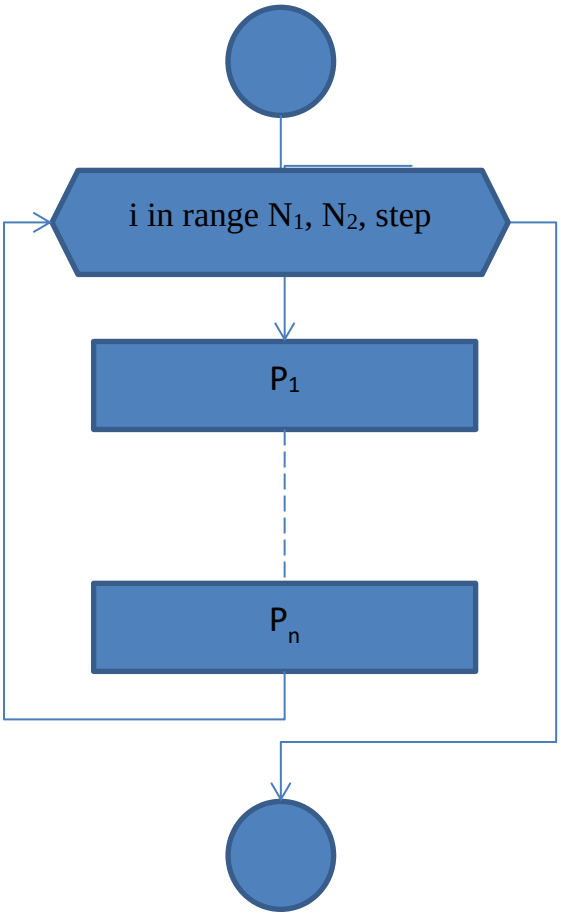
5 ЦИКЛДІК АЛГОРИТМ

5.1 Қарапайым циклдік процесс

Көбінесе бағдарламаларда белгілі бір мәлімдемелерді бірнеше рет орындау керек. Бұл әрекеттер тізбегін қатарынан жиырма-елу рет жазу қисынсыз. Мұндай жағдайларда циклдік есептеулер ұйымдастырылады. Егер белгілі бір қадамдар тізбегі берілген мәнге байланысты бірнеше рет орындалса, **цикл параметрі** деп аталатын,оның **алгоритмі циклдік** деп аталады. Параметр белгілі бір мәнді қабылдаған кезде цикл аяқталады. Белгілі қайталанулар саны бар циклдарды ұйымдастыру үшін Python тілінде **for** операторы қолданылады. Оның алгоритмдегі жалпы көрінісі 42-шісуретте көрсетілген.



Сурет 42 – **For** цикл операторының жалпы түрі

Python тіліндегі **for** циклында әр түрлі жазу формалары болуы мүмкін . Синтаксисын қарастырайық, бірінші түрі. Оны "параметрдің өсіп келе жатқан мәндеріндегі цикл" деп атайық. Егер біз P1...PN параметрлерін циклдың ішінде орындалсын десек онда шегіністерге назар аудару керек.

for i **in range** (N_1, N_2, **step**):

P_1
.
.
P_n } Цикл денесі

мұндағы **for** (үшін) – қызмет сөзі; **i** –элементтер мәні сақталатын айнымалы атауы, **P_1,...,P_n** - операторлар; **in** - в; **range** – **Python** тілінің кірістірілген функциясы; **step** - қадамы, міндетті емес параметр.

Range функциясының аргументтері тек бүтін сандар болуы керек. Бұл құрылымның **for** цикл операторының жұмысы келесід. Циклге бірінші рет кірген кезде **i** цикл параметрі N_1 төменгі шекарасының шамасына тең мән алады және цикл денесінде оператор немесе операторлар орындалады. Содан кейін параметр мәні **step** мәніне артады және цикл денесі қайтадан орындалады. Мұндай әрекеттер цикл параметрінің мәні N_2-1 мәніне тең болғанша қайталанады, содан кейін циклден шығу жүзеге асырылады. Егер қадам аргументі **range** функциясында жоқ болса, онда цикл параметрін өзгерту қадамы бірлікке(1) тең болады.

Есеп 5.1. 1-ден 50-ге дейінгі бүтін сандардың қосындысын табыңыз. Есепті шешу алгоритмінің блок-схемасы 43-ші суретте көрсетілген.

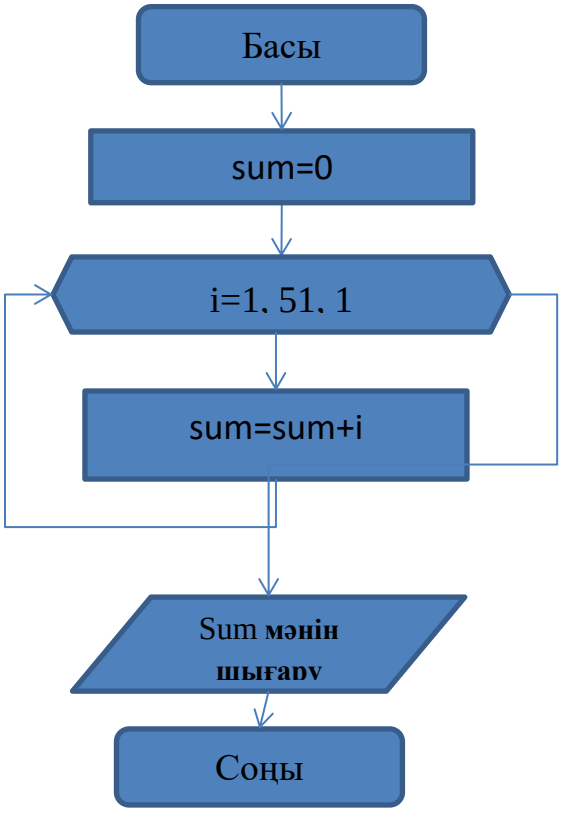


Рисунок 43 – 5.1 есептің шешу блок-схемасы

Цикл бірінші рет енгізілген кезде цикл параметрі бірлікке тең мәнді қабылдайды және **sum=sum+i** операторы орындалады, содан кейін **i**

параметрі біртіндеп бірлікке тең қадам мәніне көбейтіледі және әр уақытта циклде **sum=sum+i** операторы орындалады. Цикл параметрі **есептегіш** деп аталады, яғни мәні 50 -ге жеткенде цикл тоқтайды. Листингте есептің бағдарлама коды бар:

```
sum=0
for i in range(1, 51, 1):
   sum=sum+i
print("Сумма=", sum)
```

Егер **step** параметрі міндетті емес деп санасаңыз, онда цикл тақырыбын басқаша жазуға болады:

```
sum=0
for i in range(1, 51):
   sum=sum+i
print("Сумма=", sum)
```

Дәл осы бағдарламаны ,цикл параметрдің кему мәні бойынша жазайық. Оны **for** цикл операторының екінші түрі деп алайық.Range функциясындағы соңғы аргумент минус бірге тең. Сәйкес, **i** цикл параметрі 50-ден 1-ге дейін өзгереді, 0 мәні қайта таңдауда қосылмайды:

```
sum=0
for i in range(50, 0, -1):
   sum=sum+i
print("Сумма=", sum)
```

For цикл операторының үшінші формасын жазуда **range** функциясын және жоғарғы шекараның мәнін көрсететін бір параметрді қолдана отырып ұйымдастыруға болады. Осылайша, цикл параметрі 0-ден N_2-1 мәніне дейін өзгереді. Жалпы түрдегі оператордың синтаксисін келесідей жазуға болады:

for i **in range** (N_2):

P_1
.
.
P_n

Тело цикла

«1-ден 50-ге дейінгі бүтін сандардың қосындысын табу» бағдарламасын енді келесі операторлар тізбегі ретінде жазуға болады:

```
sum=0
for i in range(51):
   sum=sum+i
```

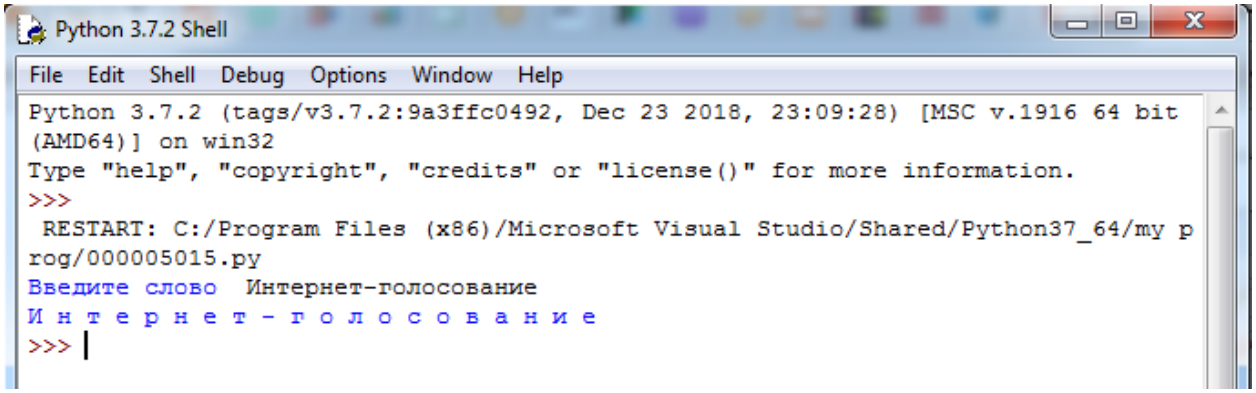
```
print("Сумма=", sum)
```

Қарастырылған барлық бағдарламалардың нәтижесі-1275 санына тең болады.

Біз төменде **for** цикл операторын қолданудың кең мүмкіндіктерін қарастырамыз, енді таңбалар тізбегін өңдеумен байланысты тағы бір мысал келтіреміз:

```
slovo=input("Введите слово  ")
for i in slovo:
    print(i, end=" ")
```

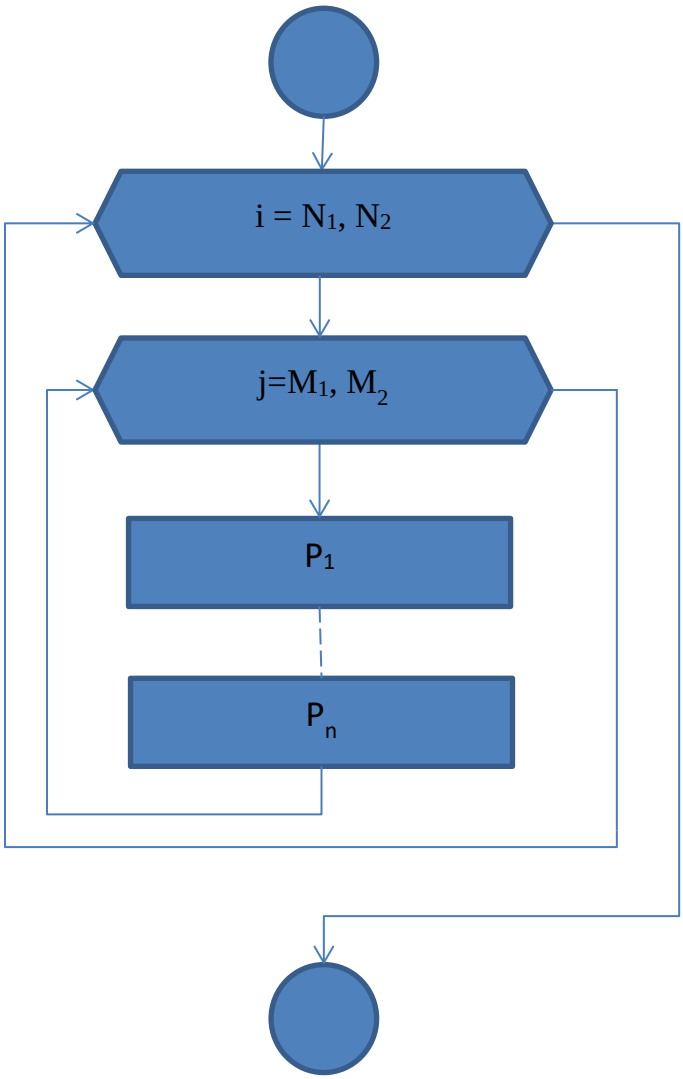
Жоғарыда келтірілген листингтен көрініп тұрғандай, циклде пайдаланушы пернетақтадан енгізетін сөздің(жолдың) таңбалары қайта есептеледі. Бағдарлама жұмысының нәтижесі 44-ші суретте көрсетілген. **Print** операторында, жоғарыда қарастырылған мысалдардан айырмашылығы, кішігірім қосымша қолданылады, атап айтқанда **end**=" " операторы, бұл бағдарлама нәтижелерін бұрын жасалғандай бағанға емес, жолға орналастыруға мүмкіндік берді.



Сурет 44 – **For** циклмен таңбалар тізбегін өңдеуге арналған бағдарлама нәтижесі

5.2 Күрделі циклдік процесс. Кірістірілген циклдар

Егер цикл денесі циклдік құрылым болса, онда мұндай циклдер кірістірілген деп аталады. Басқа циклді қамтитын *цикл* **сыртқы** деп аталады, ал басқа циклдің денесіндегі *цикл* **ішкі** деп аталады. Күрделі циклдік процесс алгоритмінің блок-схемасының фрагментінің жалпы көрінісі 45-ші суретте көрсетілген.



Сурет 45 – Күрделі циклдік процестің блок-схемасының үзіндісі

Күрделі цикл операторларының синтаксисі төменде келтірілген. Ішкі цикл үшін жасалған шегіністерге және онда орындалатын **P_1**... **P_n** операторларына назар аударыңыз:

for i **in range** (N_1, N_2): # сыртқы цикл
  for j **in range** (M_1, M_2): # ішкі цикл
    P_1
    .
    .
    P_n
    } цикл денесі

Күрделі циклдік процестің жұмысын қарастырайық. Циклге бірінші рет кірген кезде сыртқы **i** цикл параметрі **N_1** мәнін алады. Басқару ішкі циклге беріледі, онда **j** циклінің параметрі **M_1**-ге тең мән алады және ішкі циклде жазылған оператор (операторлар) орындалады. Содан кейін **j** ішкі циклінің параметрі бір-біріне артады (егер қадам аргументі **range** функциясында қабылданбаса) және цикл денесі қайтадан орындалады. Содан кейін **i** сыртқы цикл параметрі бірлікке артып, ішкі цикл қайтадан жұмыс істей бастайды,

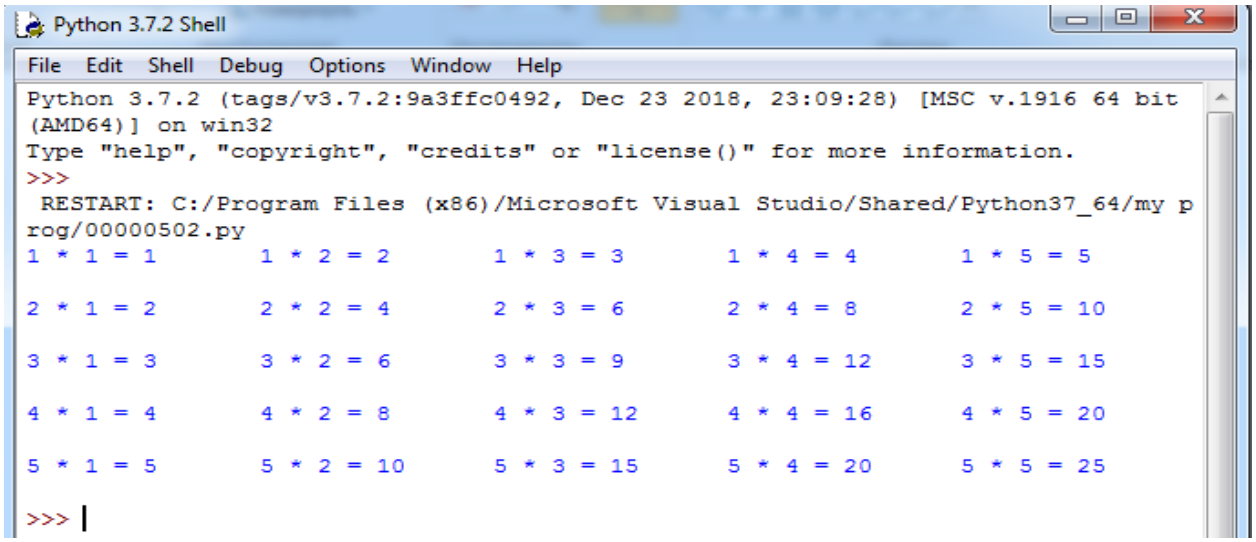
онда **j** цикл параметрі $\mathbf{M_1}$-ден $\mathbf{M_2}$-ге дейін өзгереді және циклдің әр өтуінде $\mathbf{P_1, \ldots, P_n}$ операторлары орындалады.

Есеп 5.2. 1-ден 5-ке дейінгі мәндер үшін көбейту кестесін көрсететін қосымшаны жасаңыз.

Шешімі. Шешім тізімі төменде келтірілген:

```
for i in range(1, 6):
    for j in range(1, 6):
        print(i, '*', j, '=', i*j, end='\t')
    print()
```

Сонымен, күрделі циклдік процестің алгоритміне сәйкес ішкі және сыртқы циклдердегі параметрдің өзгеру шекараларын 1-ден 6-ға дейін белгілейміз. Ішкі циклде **print(i, '*', j, '=', i*j, end='\t')** операторы орындалады, онда параметрлер мәндерінің нәтижесі, көбейту және теңдік белгілерін экранға шығару үшін жол өрнектері және іс жүзінде **i*j** әрекеті біріктіріледі, бұл параметрлердің көбеюін қамтамасыз етеді. "Бос" print () операторы (шегініске назар аударыңыз) ішкі циклде орындалмайды, бірақ жауапта шығатын бағандарының арасындағы жолды өткізіп жіберуге қызмет етеді. Бағдарлама жұмысының нәтижесі 46-суретте көрсетілген.



Сурет 46 – көбейту кестесін шығару

5.3 For циклдарды қолдануға арналған тапсырмаларды түсіндіру

Есеп 5.1. Цикл денесінде d=5 операторы қанша рет орындалады?

```
d=4
r=15
for i in range(d+1, r,1):
    d=5
```

Шешімі . d және **r** айнымалыларының мәндерін қойып отырып, **i** цикл параметрі 5-тен 14-ке дейін өзгереді. Циклге алғаш кірген кезде цикл параметрі 5 мәнін алады, содан кейін ол **14** жоғарғы шекарасының мәніне жеткенше автоматты түрде **+1**-ге артады. Демек, оператор **10** рет орындалады.

Есеп 5.2.Бір топ операторлар орындалғаннан кейін **r** ұяшығында қандай мән болатынын анықтаңыз

```
r=50
s=0
for i in range(5, 0, -1):
   s=1
   r=r-s
print("r = ", r)
```

Шешімі . Циклға алғаш кірген кезде цикл параметрі 5 мәнін қабылдайды. Бағдарламада цикл мәні төмендейді, сондықтан цикл денесінің әр орындалуында цикл параметрі -1-ге азаяды. Циклге алғаш кірген кезде **s** айнымалысы 1-ге тең мән алады. **r=r-s** операторы орындалады. ол орындалғаннан кейін **r** ұяшығында 49 мәні болады. Содан кейін цикл параметрі бірге азайып соңында төртке тең болады. Ары қарай **s=l** және **r= r-S** операторлары қайтадан орындалады. оларды әрдайым орындап, алынған мәндерді есептеп, **r** ұяшығында 45 мәнін аламыз.

Есеп 5.3. Бір топ операторлар орындалғаннан кейін **y** ұяшығы қандай мәнге тең болатынын анықтаңыз?

```
a=7
d=5
y=0
for i in range(1,4,1):
   y=d
   y=a+2
print("y = ", y)
```

Шешімі. For операторы бар циклде цикл параметрі 1-ден 3-ке дейін өзгереді. Циклде екі оператор орындалады: **y=d** және **y = a+2**. Операторлар үш рет орындалады. Цикл өткен сайын **y** ұяшығында 5 мәні болады(y=d), ал y=a+2 операторы орындалғаннан кейін **y** ұяшығының мәні **9** болады. Сондықтан, операторлар тобын орындағаннан кейін **y** ұяшығында **9** мәні болады

Есеп 5.4. Операторлар тобын орындағаннан кейін **y** ұяшығында қандай мән болатынын анықтаңыз және цикл денесіндегі **i** ұяшық қандай мәндерді қабылдай алады?

```
a=17.0
d=a
for i in range(3):
   print("i = ", i)
   if a!=d:
      a=a+1
   else:
      y=a
print("y = ", y)
```

Шешімі. For операторымен циклде логикалық өрнек тексеріледі a!=d. A және d мәндері тең болғандықтан, **a=17.0** және **d=a** орындалғаннан кейін , логикалық өрнек әрдайым *жалған* болып қалады, сондықтан циклдың әр өтуінде **else** тармағында **y=a** операторының орындалуына кепілдік беріледі. Осылайша, **print** операторының көмегімен экранда **17.0** мәні көрсетіледі. Жоғарыда қарастырылған теорияға сәйкес **i** цикл параметрі 0-ден 2-ге дейінгі реттік мәндерді қабылдайды.

Есеп 5.5. Операторлар тобын орындағаннан кейін **y** ұяшығында қандай мән болатынын анықтаңыз?

```
a=1
y=1
for i in range(2,6,1):
   a=a+10
   y=a+10
print("y = ", y)
```

Шешімі. Циклге бірінші рет кірген кезде цикл параметрі төменгі шекараның мәнін алады , екіге тең болады. Циклде екі оператор орындалады, осылайша **a** ұяшығында **11** мәні болады, ал **y=a+10** операторы орындалғаннан кейін **y** ұяшығының мәні 21-ге тең болады. Параметр бірлікке артады және үшке тең болады, **a=a+10** операторы орындалады, ал **a** ұяшығы 21-ге тең болады , ал **y** ұяшығы 31-ге тең болып шығады. Осы қадамдарды тағы екі рет орындағаннан кейін бізде **y** ұяшығы 51-ге тең болады.

Есеп 5.6. Операторлар тобын орындағаннан кейін **y** ұяшығында қандай мән болатынын анықтаңыз?

```
a=5
d=5
```

```
y=0
for i in range(7,1,-1):
   a=d
   y=a+10
print("y = ", y)
```

Шешімі. Циклге алғаш кірген кезде цикл параметрі 7-ге тең болады. **a** ұяшығының мәні **5**-ке тең болады, ал **y** ұяшығында **y=a+10** операторы орындалғаннан кейін мәні **15**-ке тең болады. Параметрдің кему мәндері бойынша циклде **i** ұяшықтың мәні бір бірлікке азаяды және **6**-ға тең болады. Одан кейін **a=d** операторы орындалады және **5** саны қайтадан **a** ұяшығында пайда болады. **У** ұяшығында **y=a+10** операторы орындалғаннан кейін мәні **15**-ке тең болады. Бұл әрекеттер цикл параметрі **2**-ге тең болғанша қайталанады, сол кезде циклден шығады. **У** ұяшығының мәні **15**-ке тең болады.

Есеп 5.7. For операторымен цикл орындалғаннан кейін **y** ұяшығында қандай мән бар екенін анықтаңыз

```
a=5
d=5
y=0
for i in range(2,6,1):
   y=d+10
y=a+10
y=a*d
print("y = ", y)
```

Шешімі. Бағдарламаның осы бөлігінде циклды орындау кезінде тек бір **y=d+10** операторы оңға жылжиды. Демек, ол орындалады. Циклдің бірінші өтуінде **y** ұяшығында **15**-ке тең мән көрсетіледі. Цикл параметрі бірлікке артады. **Y=d+10** операторы қайтадан орындалады және **d** ұяшығының мәні өзгермегендіктен, ол қайтадан **15**-ке тең болады. Бұл әрекеттер параметр мәні **5**-ке тең жоғарғы шек мәніне жеткенше қайталанады. Осыдан кейін циклден шығу пайда болады және **y** ұяшығында **15** мәні болады.

Осы тапсырмаға басқаша сұрақ қойылады: "Бағдарламаның үзіндісін орындағаннан кейін **y** ұяшығының мәні нешеге тең болады?",- **y** мәні өзгеретін тағы екі операторды талдап, **25**-ке тең мәнді алу керек.

Есеп 5.8. Бағдарламаның келесі үзіндісінде цикл неше рет орындалады?

```
n=2
s=13
y=0
```

```
for i in range(s+2,n+3,-1):
   y=y+1
```

Шешімі. Айнымалылардың бастапқы мәндерін цикл тақырыбына ауыстырғаннан кейін, **i** цикл параметрі **-1** қадаммен өзгеріп, **15-тен 6-ға** дейін өзгереді. Циклге алғаш кірген кезде цикл параметрінің мәні 15-ке тең болады, содан кейін ол 14, 13, 12 және т.б. болады және параметр **6-ға** тең болғанша цикл орындалады. Осылайша, цикл **10** рет орындалады.

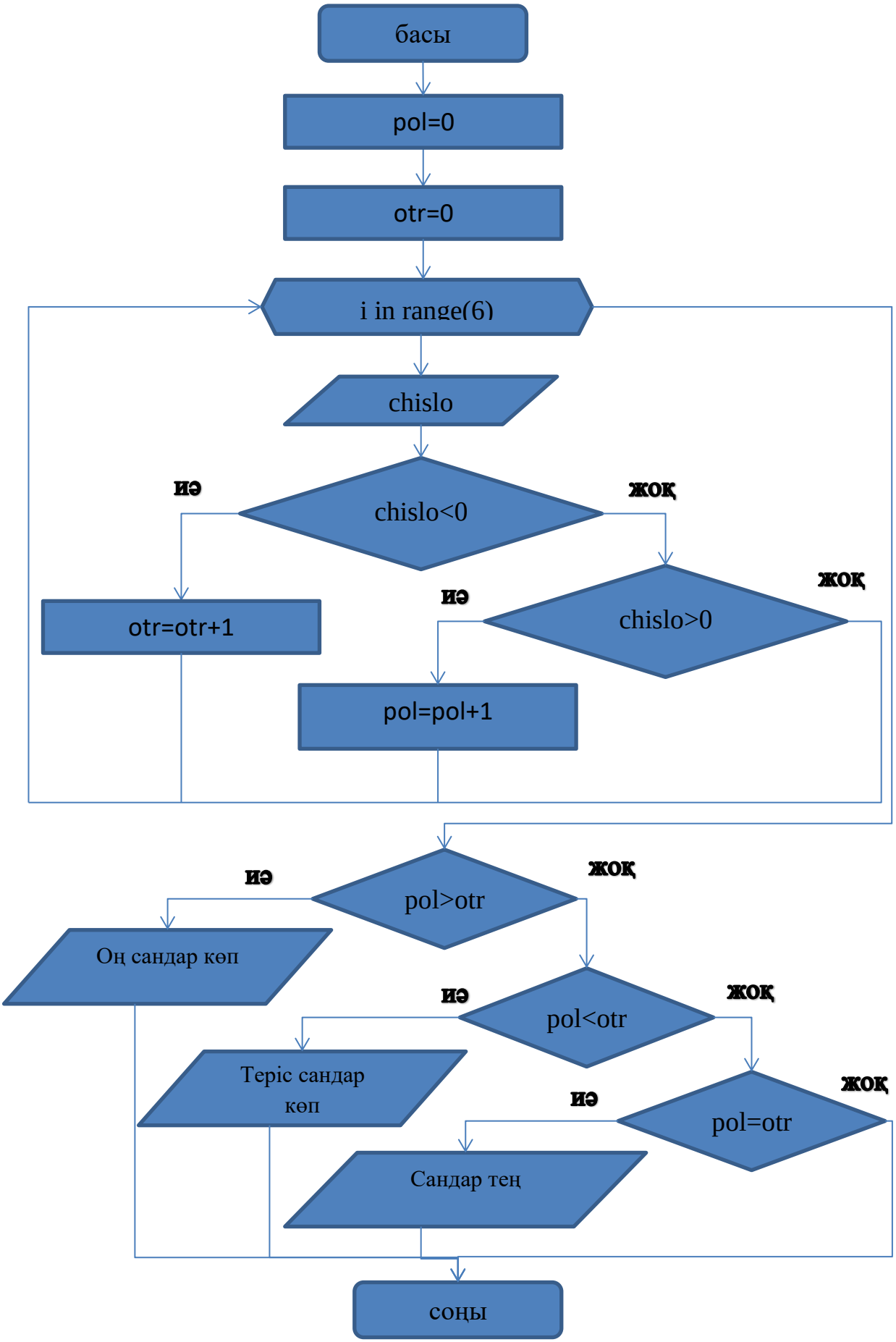
5.4 Есептерді шығару мысалдары

Есеп 5.4.1. Алты бүтін сан ретімен енгізіледі. Олардың ішінде қайсы сандар көбірек екенін анықтаңыз: оң немесе теріс.

Шешімі.Есепті шешу алгоритмінің блок-схемасы 47-ші суретте көрсетілген.

Rol және **otr** ұяшықтары есептегіштердің рөлін атқарады. Циклдегі есептегіштерді бірге арттырады келесі операторлар арқылы **otr=otr+1** және **pol=pol+1** , сондықтан қорытынды нәтиже бұрмаланбауы үшін ұяшықтарды **rol=0** және **otr=0** операторлары алдын-ала нөлге теңестіреді.

Төмендегі листингте есепті шешуге арналған бағдарлама коды берілген:



Сурет 47 – 5.4.1 есептің шешу алгоритмы

```
pol=0 #Счетчик положительных чисел предварительно обнуляется
otr=0 #Счетчик отрицательных чисел предварительно обнуляется
for i in range (6):
   chislo=int(input("Введите число "))
   if chislo<0:
      otr=otr+1 #Счетчик отрицательных чисел увеличивается на
единицу
   elif chislo>0:
      pol=pol+1 #Счетчик положительных чисел увеличивается на
единицу
if pol>otr:
   print("Положительных чисел больше")
if pol<otr:
   print("Отрицательных чисел больше")
if pol==otr:
   print("Количество чисел одинаковое")
```

Есеп 5.4.2. Бес нақты сан ретімен енгізіледі. Олардың ішінен оң сандардың арасынан ең кіші санды табу керек.

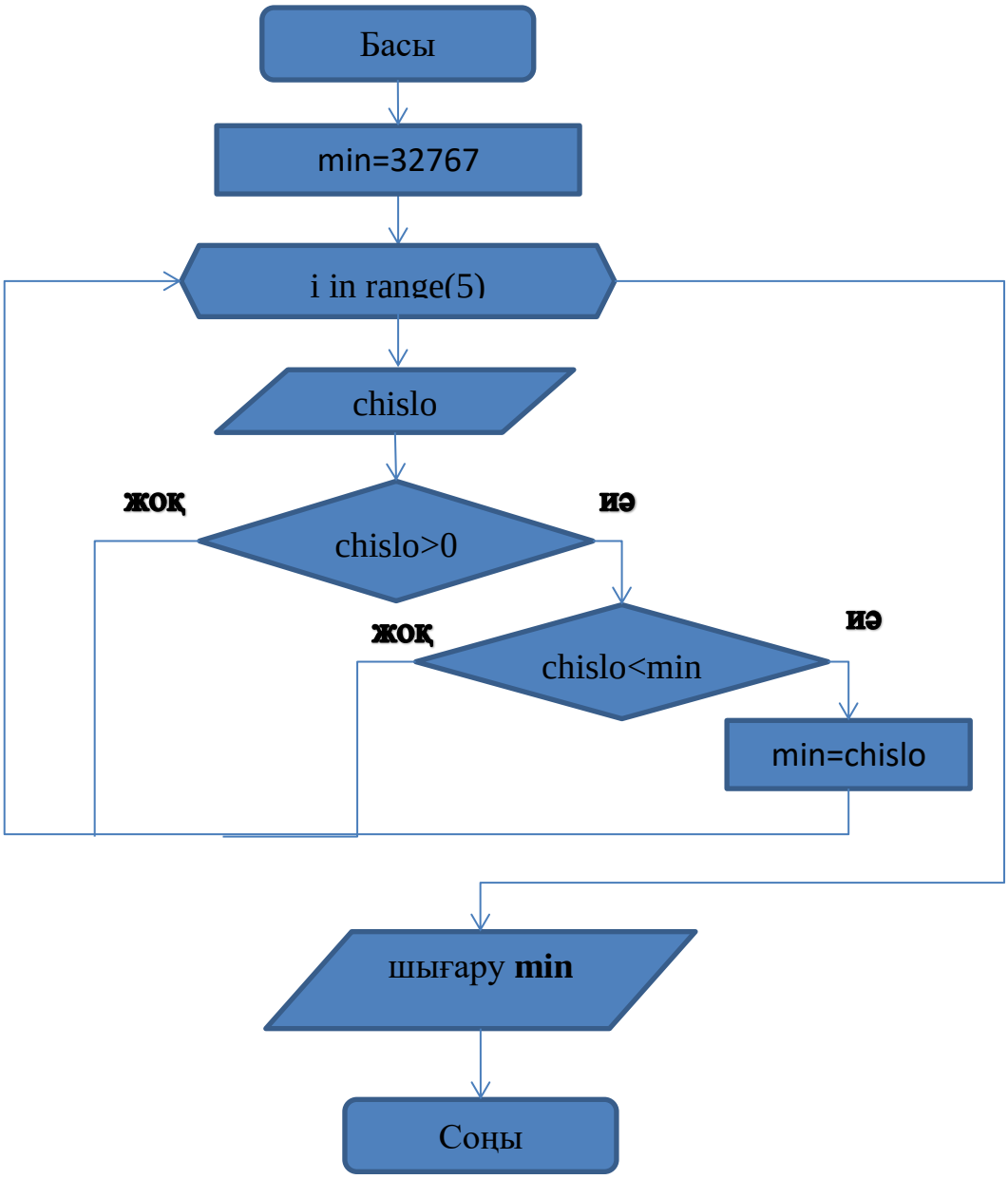
Шешімі. Есептің шешу алгоритмы 48-ші суретте көрсетілген

Минималды санды іздеу алгоритмін орындау үшін біз кез-келген үлкен мәнді алдын-ала **min** ұяшығына енгіземіз, мысалы, осы бағдарламада +32767, оператор **min=32767**. Циклде пайдаланушы енгізген санды **min** ұяшығында сақталған санмен салыстырады, ал егер ол кіші болса, енгізілген сан **min=chislo** операторымен **min** ұяшығында сақталады. Осылайша, цикл аяқталған кезде **min** ұяшығында минималды элемент болады.

Мұндай мәселелерді шешкен кезде, олардың дұрыс жұмыс істеуі үшін алдыңғы тақырыптарда көрсетілген ерекшеліктерді өңдеу әдістерімен бағдарламалардың кодын толықтыруға немесе шарттарды қосымша тексеруге болатындығын ескеру қажет (бұл тапсырмада пайдаланушы барлық теріс сандарды енгізген кезде қамтамасыз етілуі мүмкін).

Төмендегі листингте есепті шешуге арналған бағдарлама коды берілген:

```
min=32767
for i in range (5):
   chislo=float(input("Введите число "))
   if chislo>0: #Проверка на положительность очередного введенного
числа
      if chislo<min: #Поиск минимального положительного элемента
         min=chislo
print("Минимальное положительное число = ", min)
```

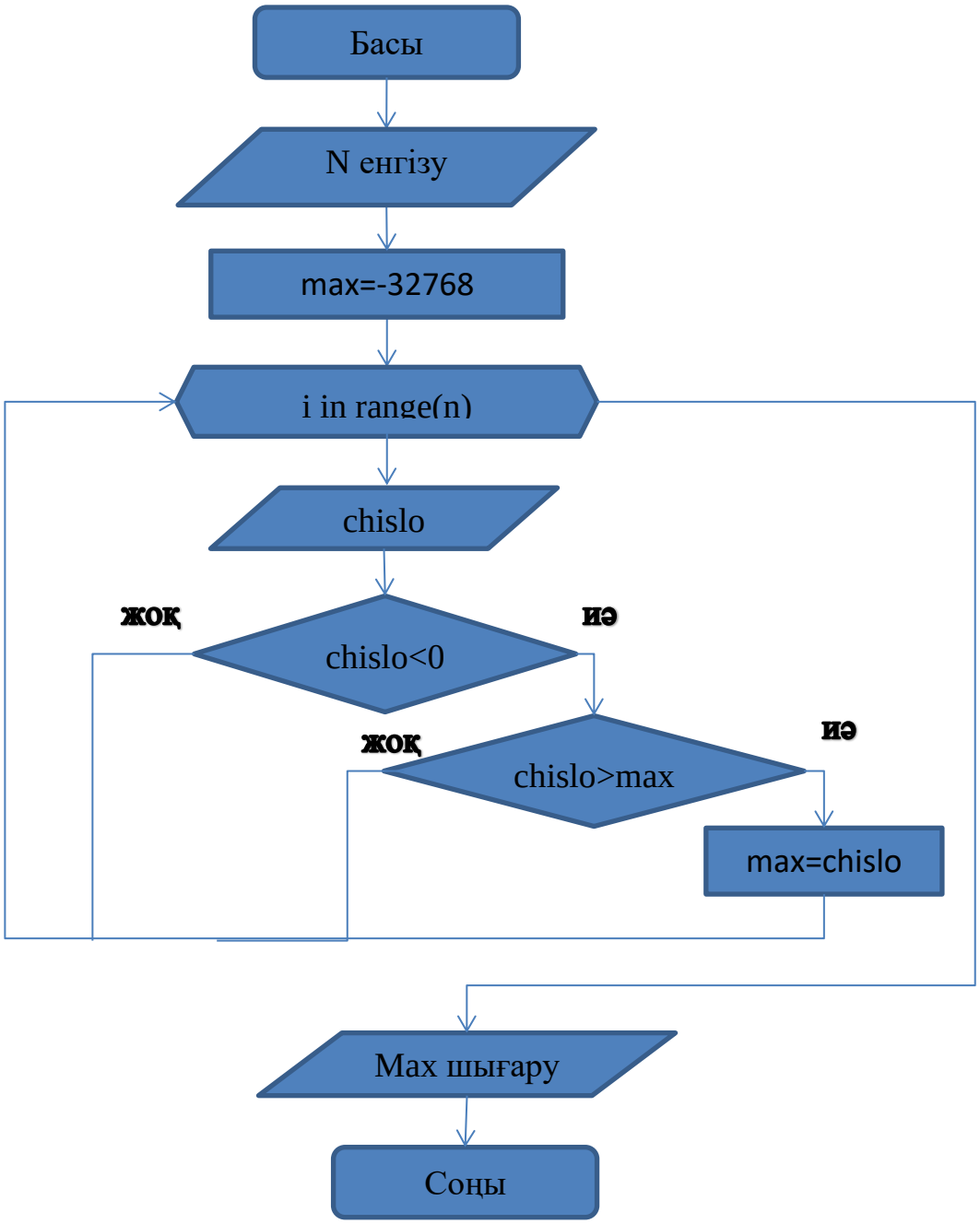


Сурет 48 – 5.4.2 есептің шешу алгоритмы блок схемасы

Есеп 5.4.3. N бүтін сандар тізбекпен енгізіледі. Теріс мәндердің ең үлкенін(максимумын табыңыз).

Шешімі. Есепті шешу алгоритмінің блок-схемасы 49-ші суретте көрсетілген.

Максималды элементті іздеу алгоритмі ең аз санды іздеу алгоритміне қарама-қарсы. Максималды санды іздеу алгоритмін орындау үшін біз кез - келген кіші мәнді алдын - ала енгізетін **max** ұяшығын белгілейміз, мысалы, осы бағдарламада теріс сан-**32768**, max=-32768 операторымен. Циклде пайдаланушы енгізген санды **max** ұяшығындағы санмен салыстырады, егер ол үлкен болса, онда енгізілген сан **max** ұяшығына сақталады келесі оператормен **max=chislo** . Осылайша, цикл аяқталған кезде **max** ұяшығында барлық енгізілген теріс сандардың арасындағы максималды элемент анықталады.



Сурет 49 – 5.4.3 есептің блок-схемасы

Төмендегі листингте есептің шешуге арналған бағдарлама коды берілген:

```
#Ввод количества чисел для цикла
n=int(input("Введите количество чисел N = "))
max=-32768
for i in range(n):
    chislo=int(input("Введите число "))
    if chislo<0: #Проверка на отрицательность очередного введенного числа
        if chislo>max: #Поиск максимального из отрицательных элементов
```

```
            max=chislo
print("Максимальное отрицательное число = ", max)
```

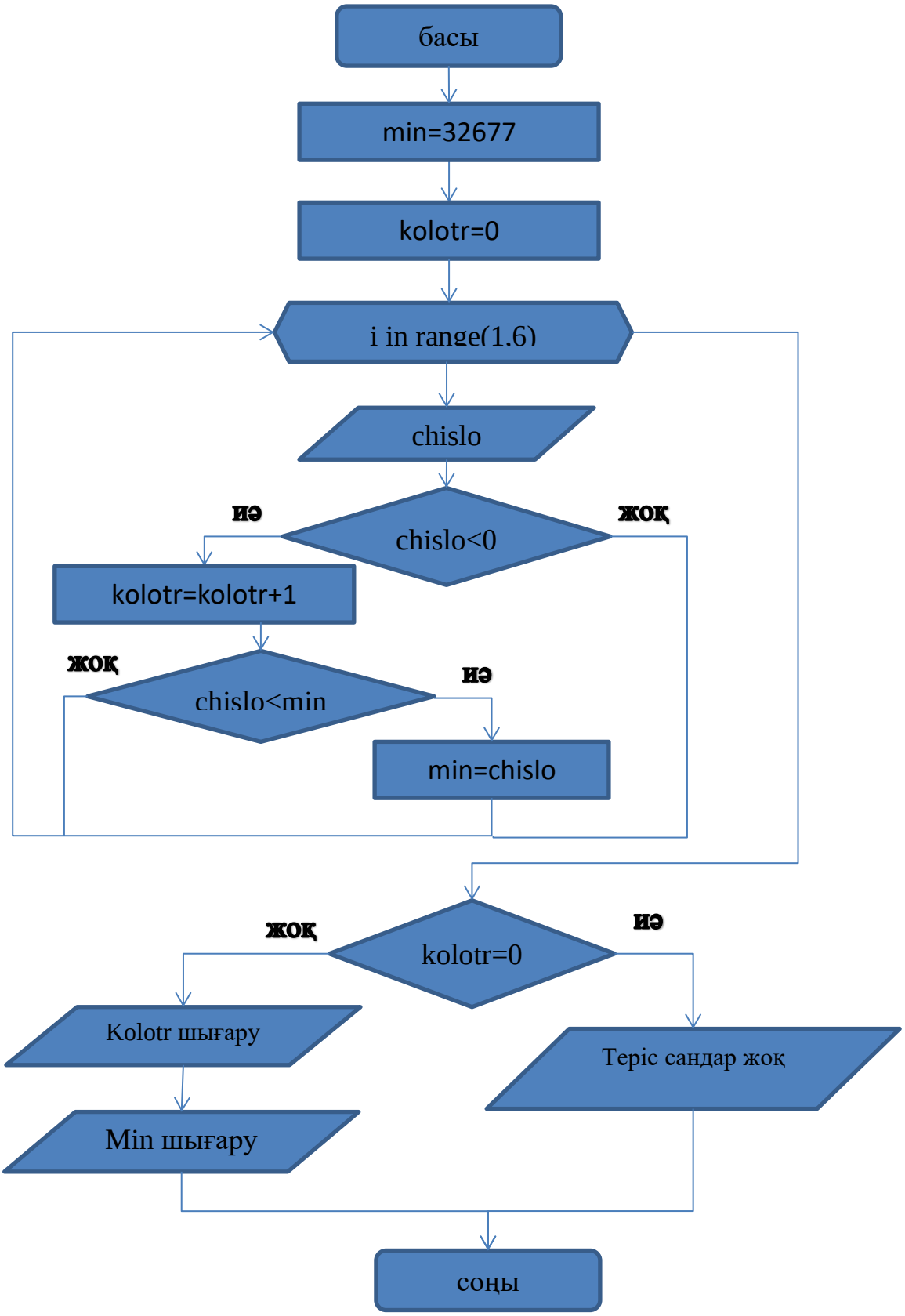
Есеп 5.4.4. Бес нақты сантізбекті енгізіледі. Теріс сандардың санын және теріс сандардың минимумын табыңыз.

Шешімі. Есепті шешу алгоритмінің Блок-схемасы 50-ші суретте көрсетілген.

Теріс элементтен ең аз элементті іздеу алгоритмі 5.4.3 тапсырмасындағы ең аз санды іздеу алгоритміне ұқсас. Минималды санды іздеу алгоритмін орындау үшін біз кез-келген үлкен мәнді алдын-ала **min** ұяшығына енгіземіз, мысалы, осы бағдарламада **min= 32767**. Теріс сандардың санын есептеу үшін **kolotr = 0** айнымалысын енгізіп, бастапқы мәнін нөлге теңейміз. Циклде пайдаланушы енгізген сан **chislo<0** теріс мәнге тексеріледі және **true** жағдайында теріс сандар санауышы **kolotr=kolotr+1** бірлігіне артады. Әрі қарай, теріс сандар циклінде енгізілген санның бұрын енгізілген барлық теріс сандардың ішіндегі ең кішісі **chislo<min** екендігі тексеріледі және осы шарт орындалған кезде енгізілген сан **min=chislo** операторымен **min** ұяшығына сақталады. Осылайша, цикл аяқталған кезде **min** ұяшығында барлық енгізілген теріс сандардың минималды элементі немесе теріс сандар болмаған кезде бастапқы енгізілген **32767** саны болады. Цикл аяқталғаннан кейін біз **kolotr==0** шартымен енгізілген теріс сандардың болуын тексереміз. Теріс сандар болмаған жағдайда біз "теріс сандар жоқ" хабарламасын басып шығарамыз, ал теріс сандар болған кезде теріс сандардың санын **kolotr** және олардың **min** ең кішісін басып шығарамыз .

Төмендегі листингте есептің шешуге арналған бағдарлама коды берілген:

```
min=32767
kolotr=0
for i in range(1,6):
   chislo=float(input("Введите число = "))
   if chislo<0:
      kolotr=kolotr+1
      if chislo<min:
         min=chislo
if kolotr==0:
   print("Нет отрицательных чисел ")
else:
   print("Количество отрицательных чисел = ", kolotr)
   print("Минимальное из отрицательных чисел = ", min)
```

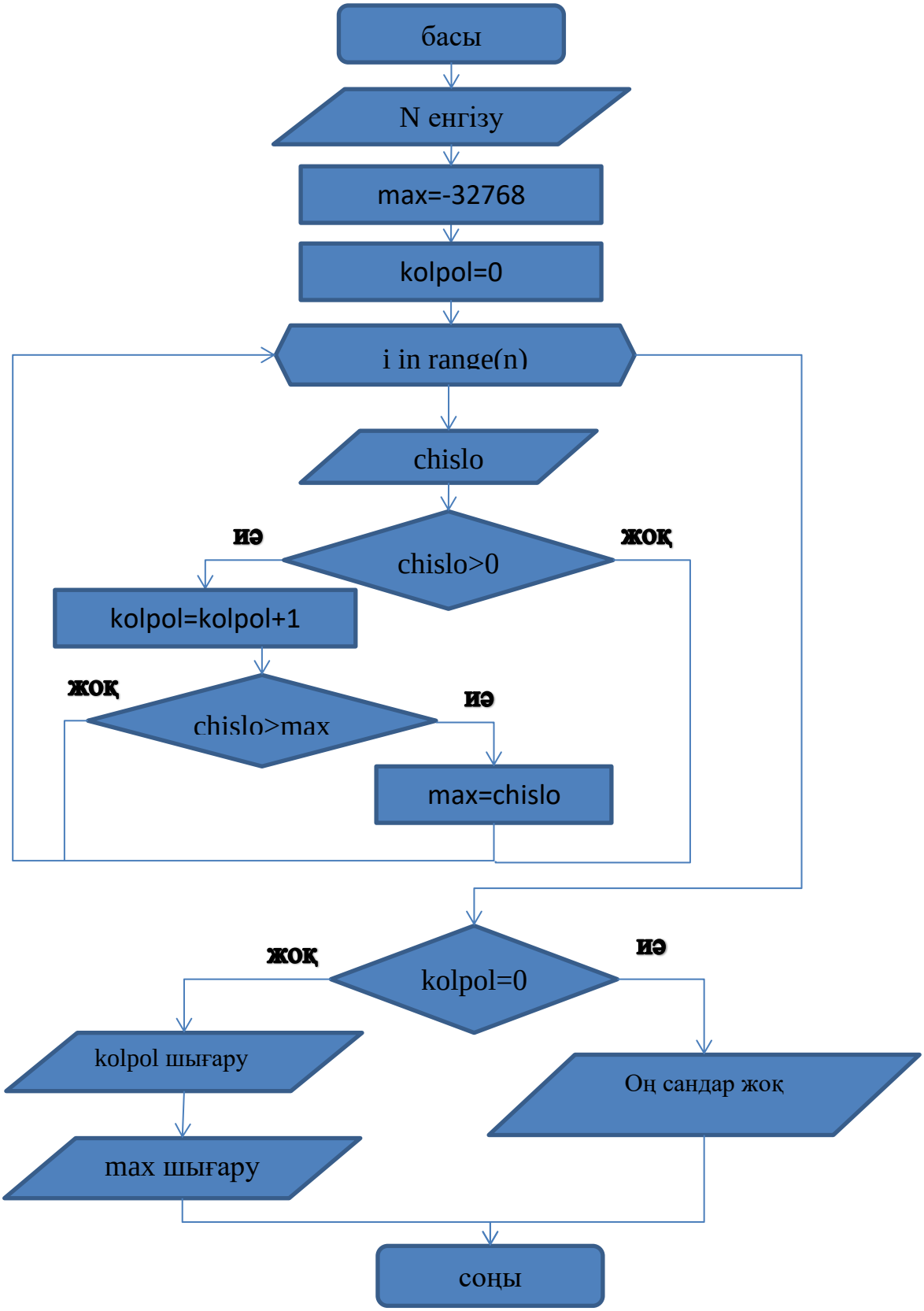


Сурет 50 –5.4.4 есептің блок-схемасы

Есеп 5.4.5. N нақты сандар қатарынан енгізіледі. Оң сандардың санын және олардың максимумын табыңыз.

Шешімі. 5.4.5 есебінің алгоритмі 5.4.4 алгоритміне ұқсас, бірақ керісінше.

Есепті шешу алгоритмі алгоритмінің блок-схемасы 51-ші суретте көрсетілген.



Сурет 51 – 5.4.5 есептің блок-схемасы

Төмендегі листингте есепті шешуге арналған бағдарлама коды берілген:

```
n=int(input("Ведите количество чисел N = "))
max=-32768
kolpol=0
for i in range(n):
   chislo=float(input("Введите число = "))
   if chislo>0:
      kolpol=kolpol+1
      if chislo>max:
         max=chislo
if kolpol==0:
   print("Нет положительных чисел ")
else:
   print("Количество положительных чисел = ", kolpol)
   print("Максимальное из положительных чисел = ", max)
```

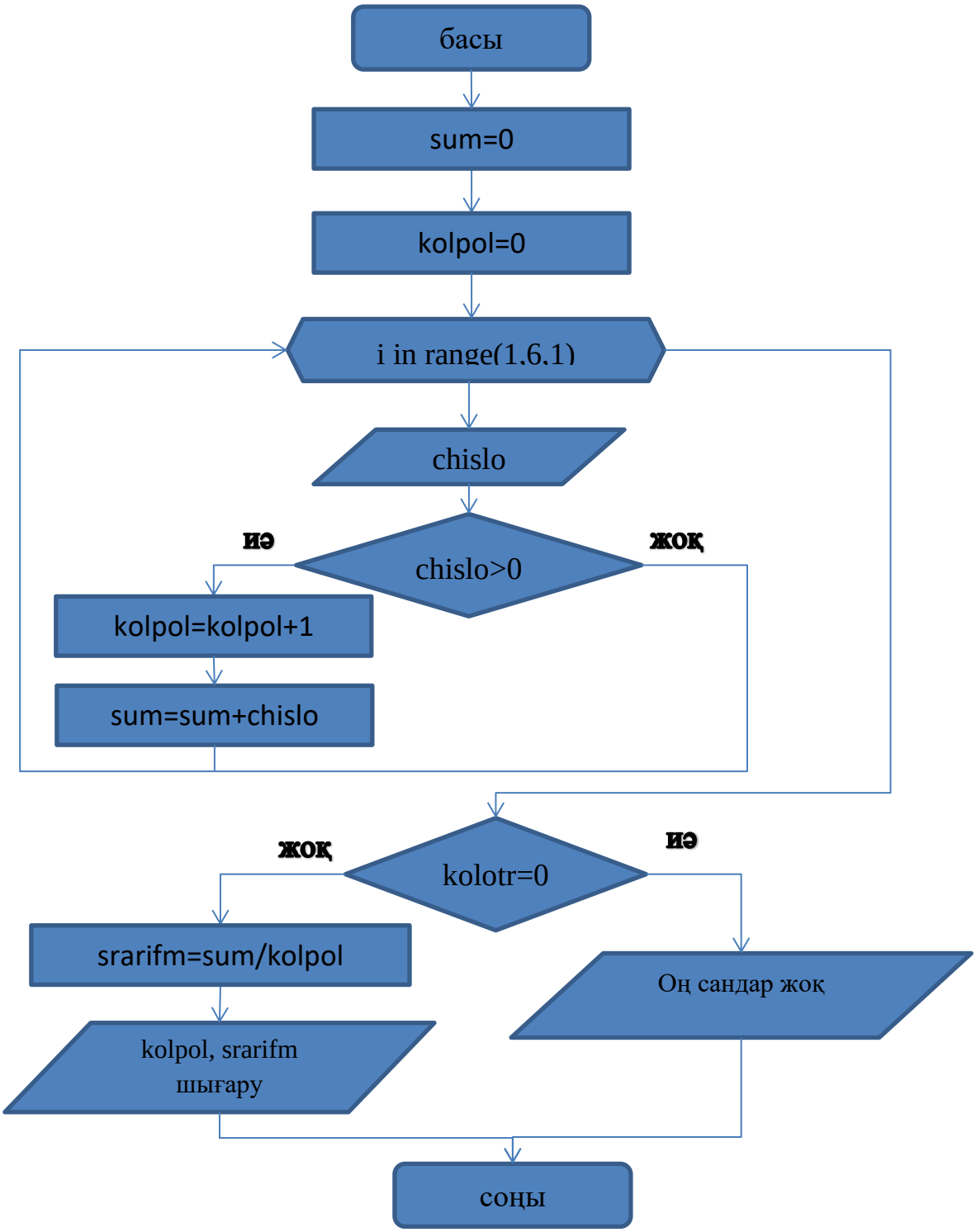
Есеп Бес бүтін сан тізбекті түрде енгізіледі. Оң сандардың санын және олардың арифметикалық ортасын табыңыз.

Шешім. Оң мәндердің арифметикалық ортасын іздеу үшін сандардың қосындысын (**sum** ұяшығы) табу керек және олардың санын (**kolpol** ұяшығы) анықтап, арифметикалық орташа мәнді есептеудің формуласы арқылы есептеуді жүзеге асыру керек **srarifm=sum/kolpol**.

Төмендегі листингте есептің шешуге арналған бағдарлама коды берілген:

```
kolpol=0
sum=0
for i in range(1,6,1):
   chislo=int(input("Введите число = "))
   if chislo>0:
      kolpol=kolpol+1
      sum=sum+chislo
if kolpol==0:
   print("Нет положительных чисел ")
else:
   srarifm=sum/kolpol
   print("Количество положительных чисел = ", kolpol)
   print("среднее арифметическое положительных значений = ",srarifm)
```

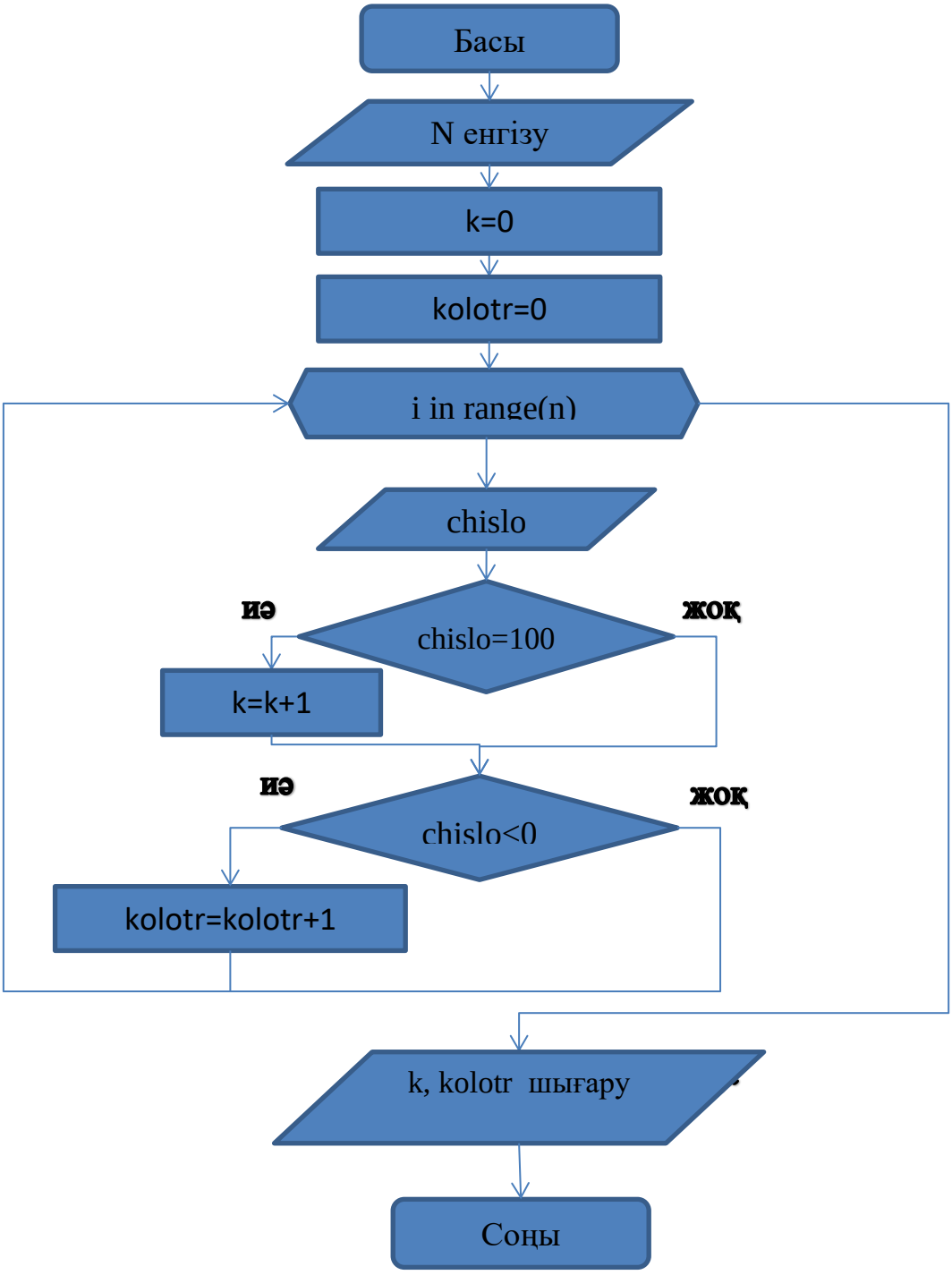
52-ші суретте есептің шешу алгоритмының блок схемасы көрсетілген



Сурет 52 – 5.4.6 есептің шешу алгоритмының блок-схемасы

Есеп 5.4.7. N бүтін сандар тізбегі енгізіледі. Ондағы 100 санына тең қанша санды, сондай-ақ теріс сандардың санын табыңыз.

Шешімі. Есепті шешу алгоритмінің блок-схемасы 53-ші суретте көрсетілген.



Сурет 53 – 5.4.7 есептің шешу алгоритмының блок-схемасы

Төмендегі листингте есепті шешуге арналған бағдарлама коды берілген:

```
n=int(input("Ведите количество чисел N = "))
k=0
kolotr=0
for i in range(n):
    chislo=int(input("Введите число = "))
    if chislo==100:
```

```
        k=k+1
    if chislo<0:
        kolotr=kolotr+1
print("Количество чисел равных сотне = ", k)
print("Количество отрицательных чисел = ", kolotr)
```

Есеп 5.4.8. Қатарынан он нақты сан енгізіледі. Олардың қаншасы бірінші енгізілген санға сәйкес келетінін анықтаңыз.

Шешімі. Есепті шешу алгоритмінің блок-схемасы 54-ші суретте көрсетілген.

Chislo1 ұяшығына бірінші санды енгізіңіз. Барлық кейінгі енгізілген сандарды, екіншіден бастап, циклде **chislo1** ұяшығындағы санмен салыстырылады. Сәйкестік жағдайында біз есептегіш(санауыш) арқылы **kol** ұяшығының мәнін бірлікке арттырамыз .

Төмендегі листингте есепті шешуге арналған бағдарлама коды берілген:

```
chislo1=float(input("Ведите первое число = "))
kol=0
for i in range(2,11):
    chislo=float(input("Введите число = "))
    if chislo==chislo1:
        kol=kol+1
if kol==0:
    print("Нет совпадений с первым числом")
else:
    print("С первым числом совпало = ", kol)
```

Есеп 5.4.9. Қатарымен он бүтін сан енгізіледі. Ең үлкен және ең кіші сандардың арасындағы айырмасын анықтаңыз.

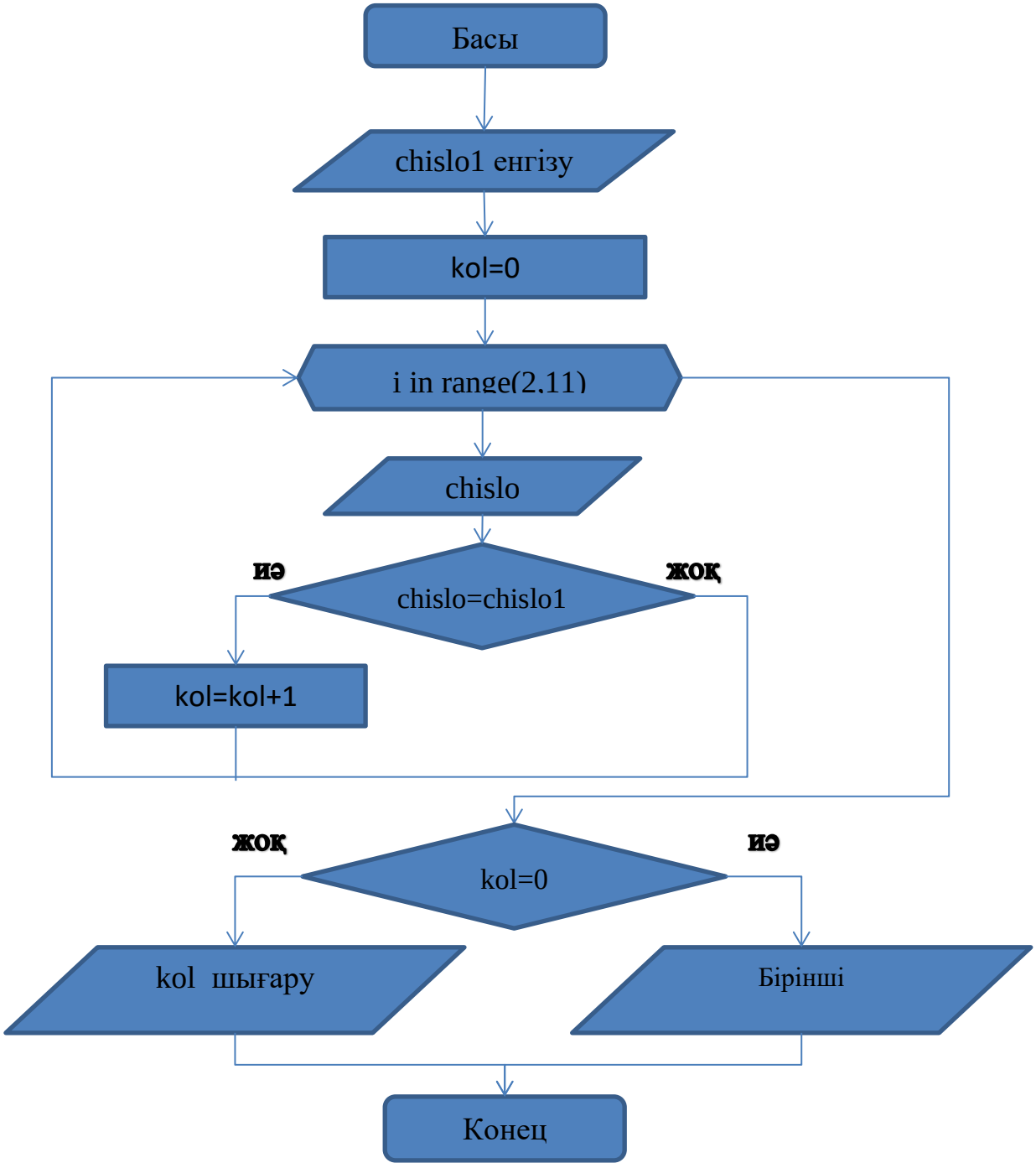
Шешімі. Есепті шешу алгоритмінің блок-схемасы 55-ші суретте көрсетілген.

Тапсырмада жоғарыда сипатталған ең кіші(минималды) және ең үлкен(максималды) элементті іздеу алгоритмдері орындалады. Цикл аяқталғаннан кейін **maxim** ұяшығында максималды элемент болады, ал **minim** ұяшығында минималды элемент болады. **Raznost=maxim-minim** операторы арқылы сандардың айырмасы есептеледі.

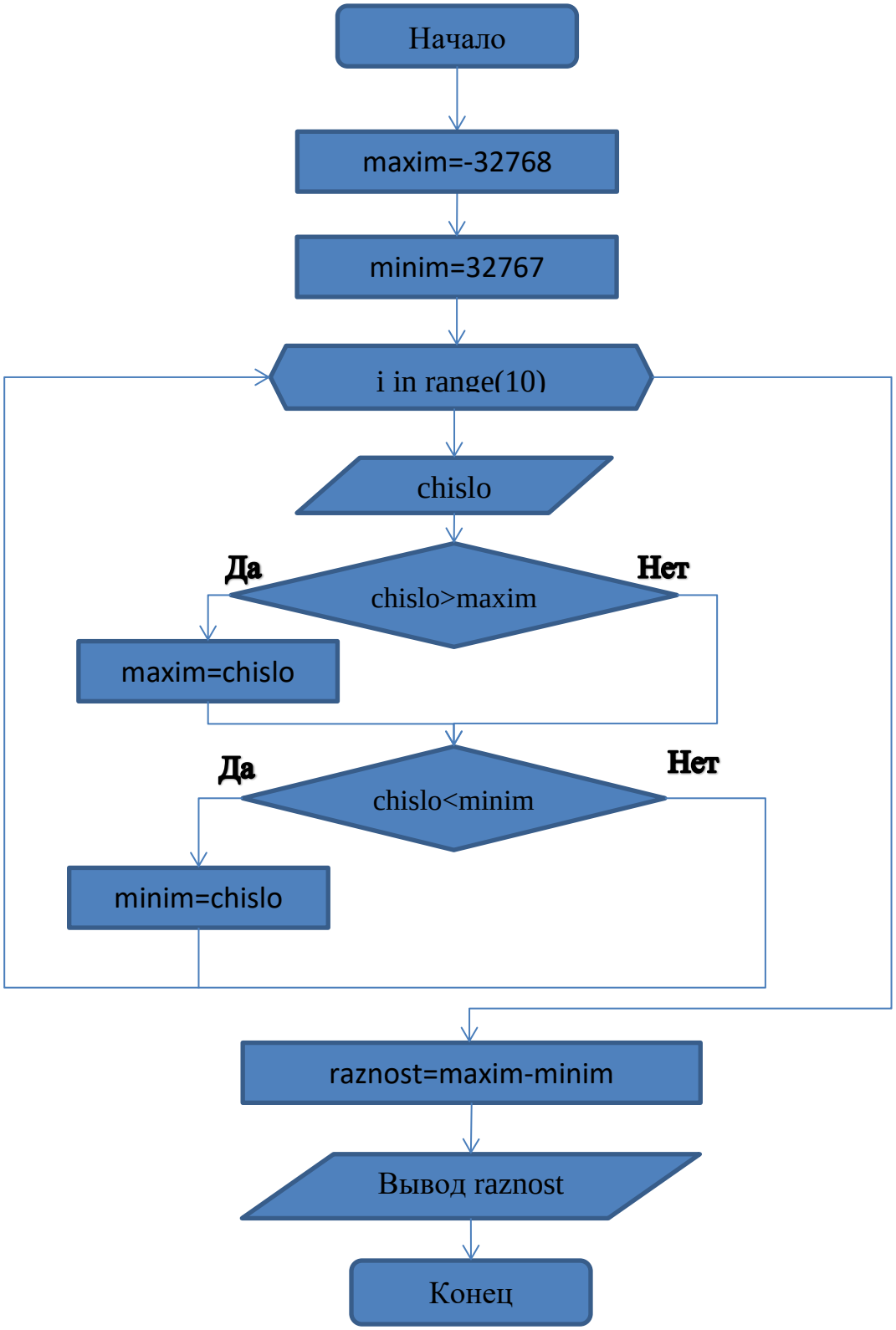
Төмендегі листингте есепті шешуге арналған бағдарлама коды берілген:

```
maxim=-32768
minim=32767
for i in range(10):
```

```
        chislo=int(input("Введите число = "))
        if chislo>maxim:
            maxim=chislo
        if chislo<minim:
            minim=chislo
    raznost=maxim-minim
    print("Разность между максимальным и минимальным числами = ", raznost)
```



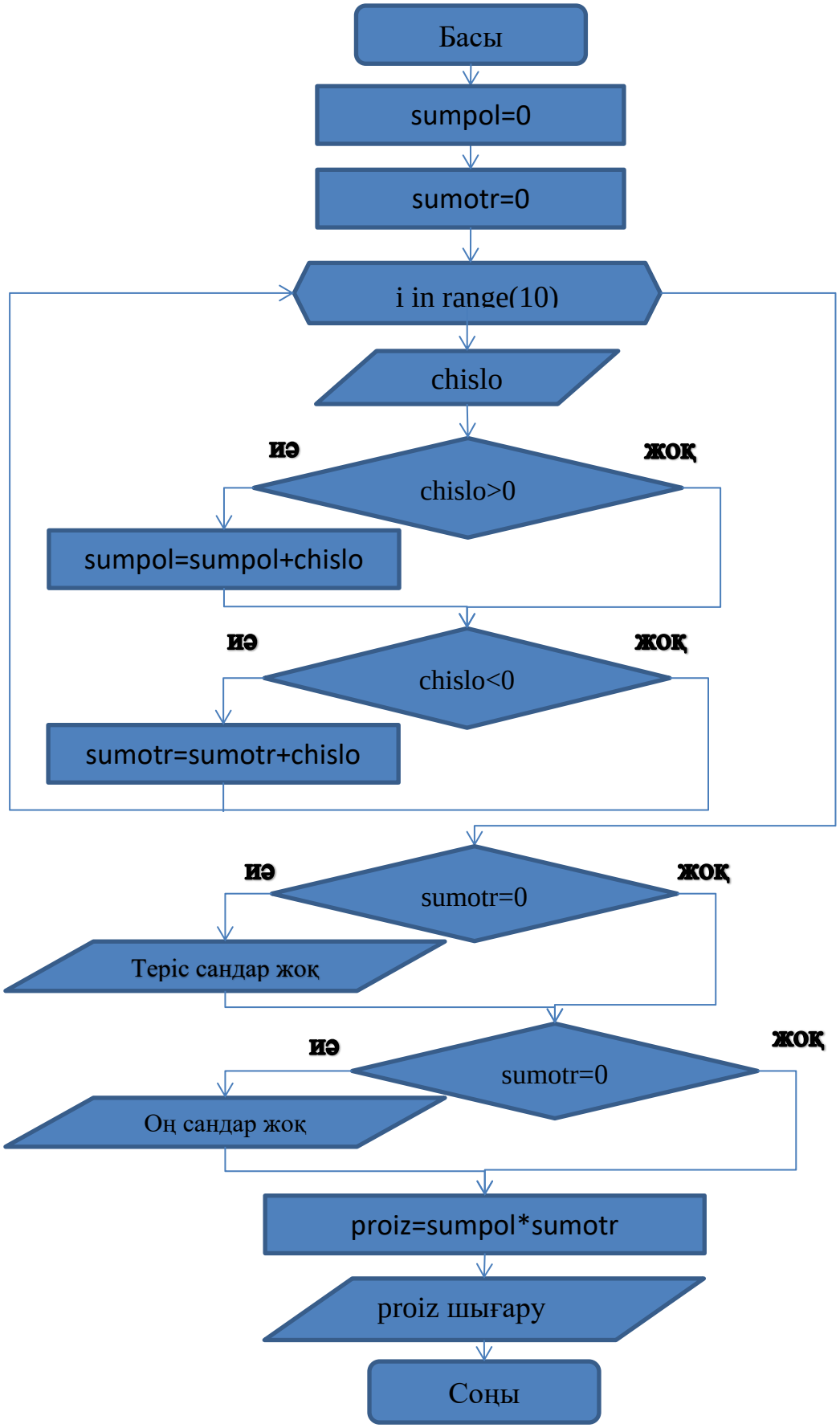
Сурет 54 – 5.4.8 есептің шешу алгоритмының блок-схемасы



Сурет 55-5.4.9 есептің шешу алгоритмының блок-схемасы

Есеп 5.4.10. Қатарымен он бүтін сан енгізіледі . Барлық оң және барлық теріс сандардың қосындыларының көбейтіндісін табыңыз. Мұны істеу үшін барлық оң және барлық теріс сандардың қосындысын алдын-ала есептеңіз.

Шешімі . Есепті шешу алгоритмінің блок-схемасы 56-ші суретте көрсетілген.



Сурет 56 – 5.4.10 есептің шешу алгоритмының блок-схемасы

Төмендегі листингте есепті шешуге арналған бағдарлама коды берілген:

```
sumpol=0
sumotr=0
for i in range(10):
   chislo=int(input("Введите число = "))
   if chislo>0:
      sumpol=sumpol+chislo
   if chislo<0:
      sumotr=sumotr+chislo
if sumpol==0:
  print("Нет положительных чисел")
if sumotr==0:
  print("Нет отрицательных чисел")
proiz=sumpol*sumotr
print("Произведение сумм всех положительных и всех отрицательных чисел = ", proiz)
```

5.5 Бақылау сұрақтары

1. Циклдік алгоритмге анықтама беріңіз.
2. **For** цикл оператордың параметрінің өсуі бойынша, алгоритмінің жалпы түрі мен синтаксисінің жұмысын көрсету керек.
3. **For** цикл оператордың параметрінің кемуі бойынша, циклдың жұмысын сипаттау керек.
4. Күрделі циклдік процестің алгоритмнің жалпы түрі мен синтаксисін көрсету керек.
5. Қандай цикл сыртқы және ішкі деп аталады?

5.6 Өздігімен шешуге арналған есептер

1. Келесі есептің шешу алгоритмі мен бағдарламасын құрыңыз. Температураның N мәндері бар. Теріс температураның санын табыңыз.
2. Келесі есептің шешу алгоритмі мен бағдарламасын құрыңыз. Температураның N мәндері бар. Орташа температураны табыңыз.
3. Келесі есептің шешу алгоритмі мен бағдарламасын құрыңыз. Температураның N мәндері бар. Теріс температураның орташа мәнін анықтаңыз.
4. Келесі есептің шешу алгоритмі мен бағдарламасын құрыңыз. Температураның N мәндері бар. Максималды температураны анықтаңыз.
5. Келесі есептің шешу алгоритмі мен бағдарламасын құрыңыз Температураның N мәндері бар. Температураның максималды және минималды мәндерін табыңыз.
6. Келесі есептің шешу алгоритмі мен бағдарламасын құрыңыз. Азғана ақша сомасы банкке пайызбен салынған. Әр жылдың соңында салым мөлшерін анықтаңыз.