# 6 ЦИКЛДЫҚ АЛГОРИТМ WHILE

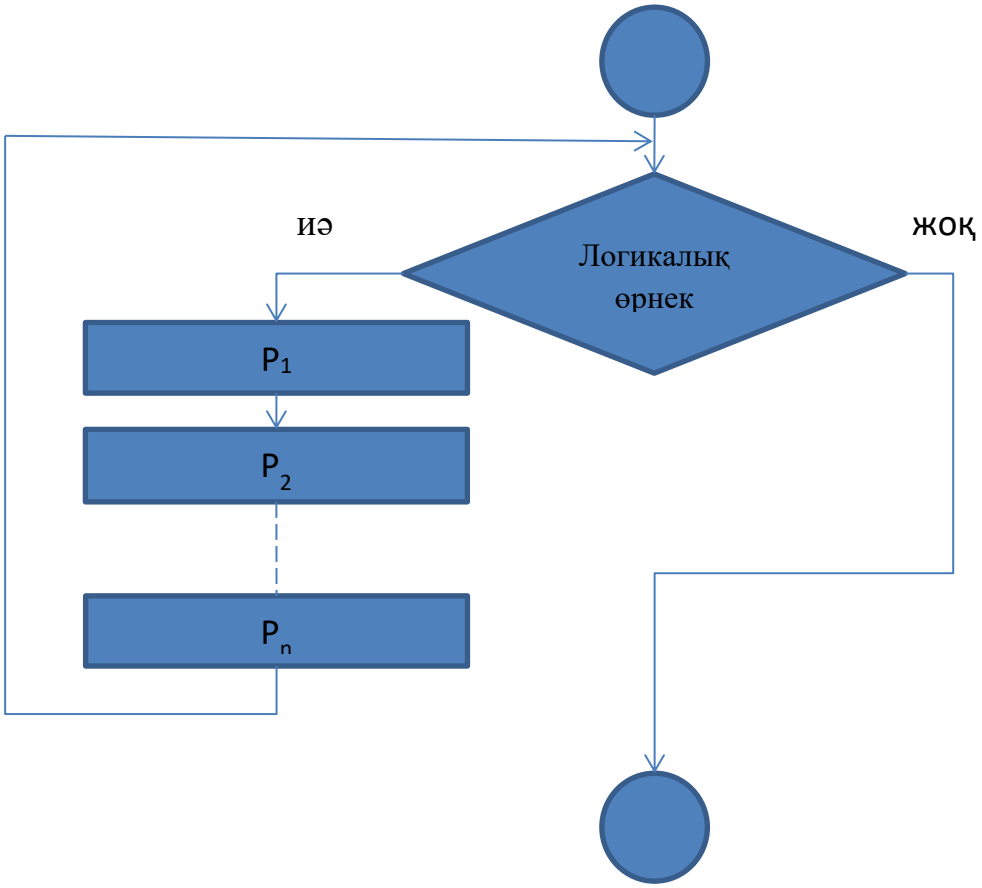
## 6.1 Цикл операторы while

**For** операторын орындау үшін цикл операторы (операторлары) қанша рет орындалатынын анықтайтын параметрлерді орнату қажет. **For** операторына баламалы болып қайталау саны белгісіз онда логикалық өрнек белгілі бір мәнді қабылдағанға дейін оператор(операторлар) орындалады.

Циклді қайталау саны алдын-ала белгісіз және алгоритмді орындау барысында ғана анықталатын циклдік құрылым **итеративті** деп аталады.

Мұндай циклдарды қанша рет цикл қайталанатынын нақты біле алмайтын тапсырмаларда қолданылуы керек. Мысалы, пайдаланушы қандай да бір бағдарламамен жұмыс істеуді бастау үшін парольді енгізуі керек. Ол қанша әрекетті қолданады? Белгісіз.

Сондықтан, мұндай міндеттерді жүзеге асыру үшін оларды шешудің тиісті әдістеріне ие болу керек. Python тілінде цикл құрылысын белгісіз қайталанулармен жүзеге асыру үшін **while** операторы немесе шартты алдын-ала тексеретін цикл қолданылады (57-ші сурет).




Сурет 57 – **while** операторының алгоритмының блок-схемасы

**while** операторының синтаксисі келесідей:
**Бастапқы мәнді инициализациялау**
**while** логикалық өрнек:
$P_1$
$P_2$

$P_n$

мұндағы $P_1$, $P_2$, ... $P_n$ - операторлар; **while** (әзірге, соған дейін ) – **Python** тілінің қызмет сөзі.

Егер **while** қызметтік сөзден кейінгі логикалық өрнек **True**(Ақиқат) болса, онда $P_1$, $P_2$, ... $P_n$ операторлары орындалады, содан кейін логикалық өрнекті қайта тексеріледі. Егер логикалық өрнек **False**(жалған) болса, онда циклден шығадыұцикл тоқтайды). Егер цикл тақырыбындағы жағдай басынан ақиқат болмаса, **while** циклі орындалмайды. Цикл құрылымындағы **for** операторының параметрі үнсіз келісім бойынша орнатылады, **while** циклінде жоқ болғандықтан, онда цикл параметрінін құру қажет болады. Сонымен қатар, бағдарламалау кезінде циклға дейін оның инициализациясын қамтамасыз ету қажет, ал циклде бұл айнымалы мәнді белгілі бір қадамға ұлғайту қажет.

Мысалы, төмендегі Листингте, "Студент"сөзі енгізілгенге дейін, **while** операторы бар цикл деректерді өңдейді.

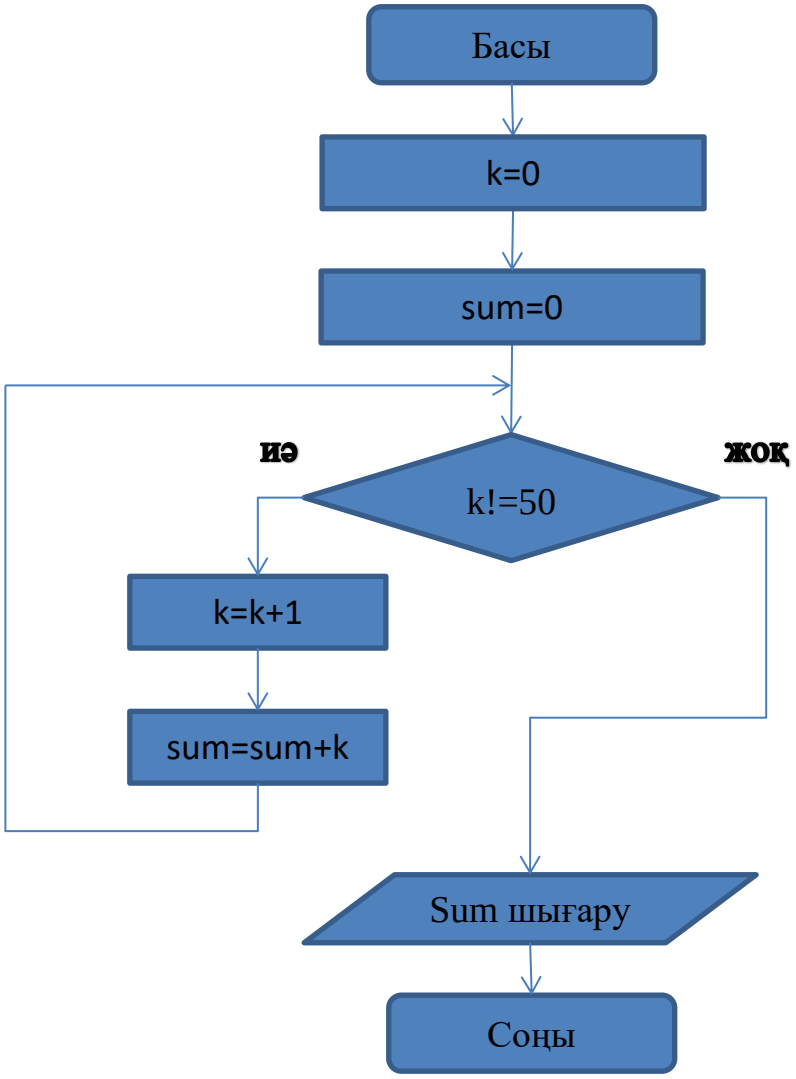
```
name=""
while name !='Студент':
   name=input("Введите любое слово для печати или слово Студент для выхода: ")
   if name!='Студент':
      print("Ответ = ", name)
```

**Есеп 6.1.** . 1-ден 50-ге дейінгі бүтін сандардың қосындысын ,**while** цикл операторын қолдана отырып табыңыз.

**Шешімі.** Бұрын осы есепті оператор **for**-ды пайдаланып шештік. Ең алдымен, бұл жағдайда кез-келген айнымалы циклде 1-ден 50-ге дейін өзгеруі керек. Цикл параметрі сияқты мән **while** құрылымында жоқ болғандықтан, бұл мысалда **k** айнымалысы осындай рөл атқарады. Осылайша, біз барлық елу шартты тұжырымдаймыз және жауапта 1275 санын аламыз. Есепті шешу алгоритмінің блок-схемасы 58-ші суретте көрсетілген.

Листингте есептің шешуге жауап беретін бағдарлама коды жазылған:

```
k=0
sum=0
while k!=50:
   k=k+1
   sum=sum+k
print("Сумма чисел от 1 до 50 sum = ", sum)
```




Сурет 58 – 6.1 есептің шешу алгоритмының блок-схемасы

Функциялардың мәндерін немесе кез-келген сандық тізбекті (мысалы, арифметикалық прогрессия) есептеу кезінде олар көбінесе, арнайы жиынтықтар түрінде жазылады **қатарлар** деп аталатын. Көптеген сандарды, функцияларды, сандық әдістердің алгоритмдерін қатарлар немесе итерациялық алгоритмдер арқылы жазуға болады, бұл олардың жуық мәндерін берілген дәлдікпен есептеуге мүмкіндік береді. Мұндай тапсырмаларды бағдарламалау кезінде циклды ұйымдастыру үшін итерациялық әдістер қолданылады, онда кейбір **рекурентті формула** есептеледі.

**Рекурренттік формула** (лат. recurrens, recurrentis — қайта оралатын), келтіру формуласы — тізбектің бастапқы **n** мүшесі белгілі болған кезде оның кез келген мүшесін есептеуге мүмкіндік беретін an+p= F(n, an, an+1, , an+p-1) түріндегі қатысты айтады, яғни қандай да болмасын тізбектің (көбінесе, сан тізбегінің) n-мүшесін алдыңғы мүшелері арқылы табатын формула.

Жалпы жағдайда бұл формула келесідей:

$$S_n = S_{n-1} + U_n,$$

мұнда $S_n$ алғашқы **n** қатарының қосындысы , ол қосындының алғашқы қадамында есептеледі $S_{n-1}$; $U_n$ - ағымдағы қадамда алынған қосынды.

### 6.2 While циклдерін қолдануға арналған тапсырмалар мысалдарын түсіндіру

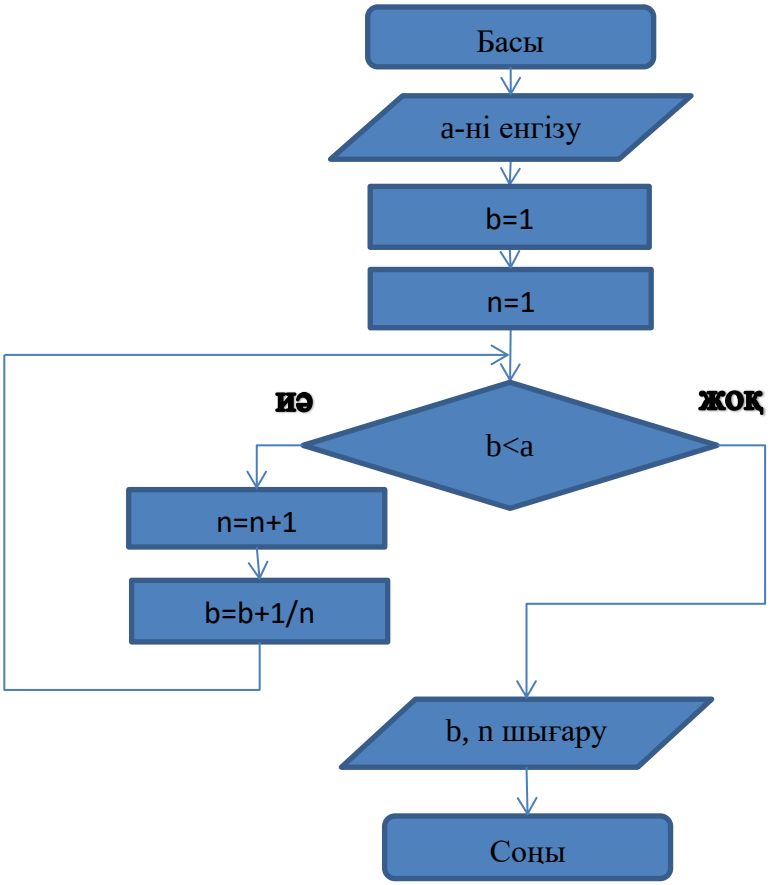
Итеративті циклдар мен рекуренттік(қайталанатын) қатынастарды қолданатын бірқатар тапсырмаларды қарастырамыз.

**Тапсырма 6.1.** Келесі сандар арасында$1, 1+\frac{1}{2}, 1+\frac{1}{2}+\frac{1}{3}, \cdots$ енгізілген **a** айнымалысынан үлкен санды табыңыз.

**Шешімі .** Бұл тапсырманы шешу алгоритмі шексіз тізбектің мүшелерін есептеу алгоритмдеріне жатады. Біз шексіз тізбектің келесі мүшесін **b** деп белгілейміз. Оның нөмірі бөлшектің бөліміне сәйкес келсе, тізбектің келесі мүшесінің мәнін алу үшін алдыңғы мүшеге қосып **n** деп белгілеңіз.Сонда тізбектің келесі мүшесін есептеуге арналған итерациялық формула келесі түрде болады:

$$b_n = b_{n-1} + \frac{1}{n}.$$

Есептің шешу алгоритмының блок-схемасы 59-шы суретте көрсетілген.

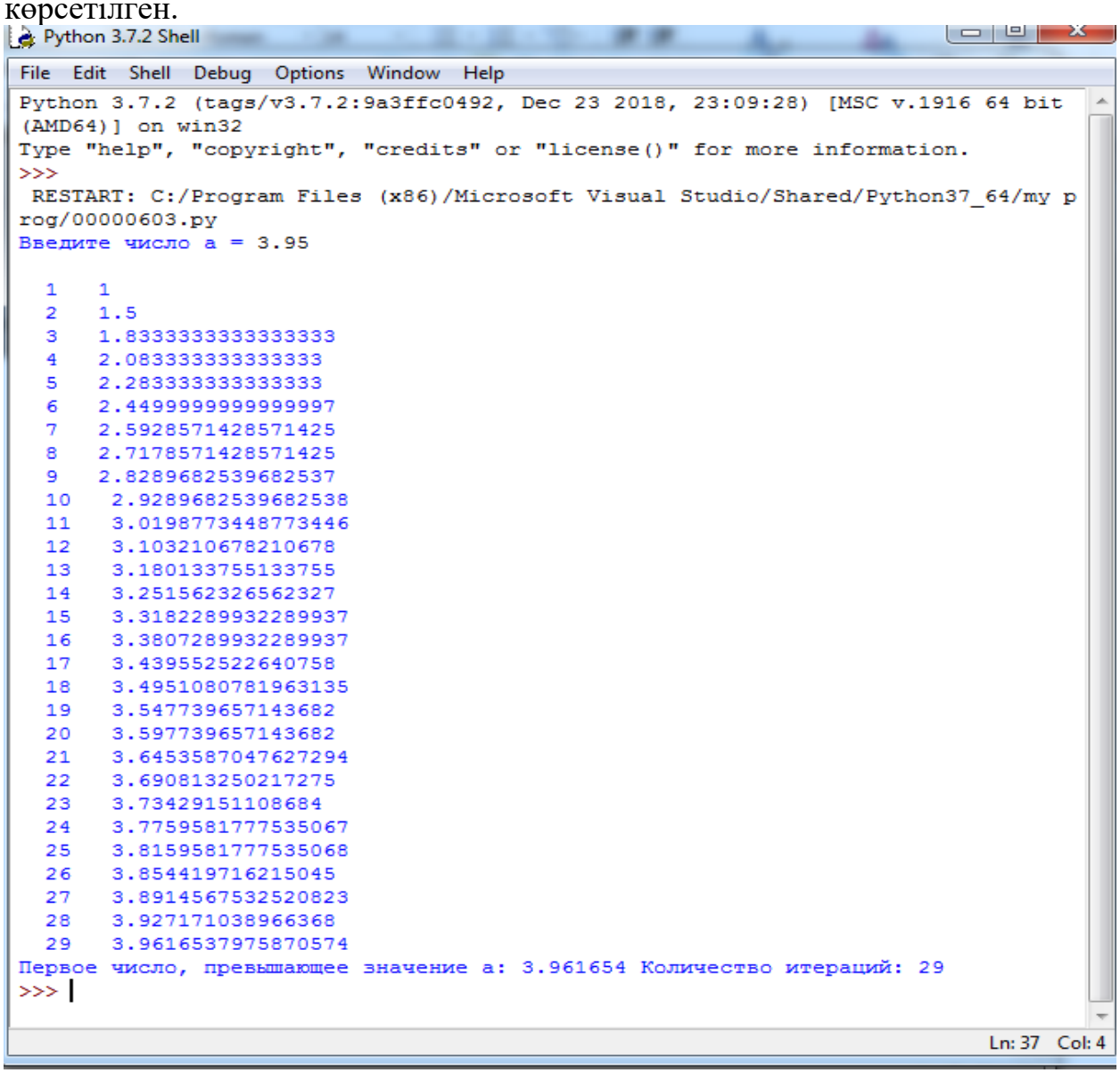



Сурет 59 – 6.1 тапсырманың шешу алгоритмының блок-схемасы

Төменде есептің шешуіне арналған программа коды берілген:

```
a=float(input("Введите число a = "))
b=1
n=1
print("\n ", n, " ", b)
while b<a:
    n+=1
    b=b+1/n
    print(" ", n, " ", b)
print("Первое число, превышающее значение a:", '{0:4f}'.format(b), "Количество итераций:", n)
```

60-шы суретте **a** мәні **3.95**-ке тең болған кездегі бағдарламаның нәтижесі көрсетілген.



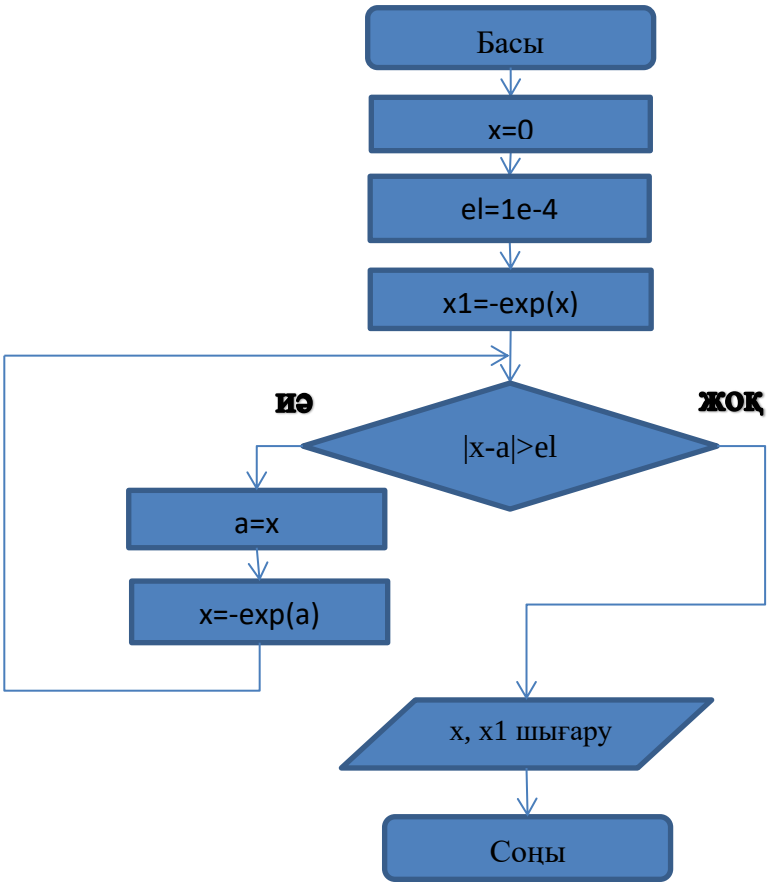

Сурет 60 – **a=3.95 тең** болған жағдайдағы бағларламма нәтижесі

**Тапсырма 6.2.** Итерациялық формуланы $x_{i+1} = -e^{x_i}$ қолдана отырып,

$e^x + x = 0$ теңдеуінің түбірін ε=$10^{-4}$ дәлдікпен есептеңіз, мұндағы $i = 0, 1, 2, \cdots;\ x_0 = 0$ Итеративті процесті келесі шарт орындалған $|x_{i+1} - x_i| < \varepsilon$ кезде аяқтаңыз.

**Шешімі.** Бұл тапсырманы шешу үшін $x_{i+1}$ есептелген түбірдің келесі мәнінен $x_i$ түбірінің алдыңғы мәнін алып тастау керек. Ол үшін циклдің әр қайталануында келесі $x$ түбірді есептемес бұрын, ағымдағы $x$ мәнін **a** айнымалысында сақтаймыз(сол кезде ол алдыңғы мән болады). Цикл $x$ -тің айырмасы $\varepsilon = 10^{-4}$ кіші болған кезде аяқталады. Бағдарламада ε-ді e1 деп белгілейміз.

Тапсырманы шешу алгоритымның блок-схемасы 61-ші суретте көрсетілген.



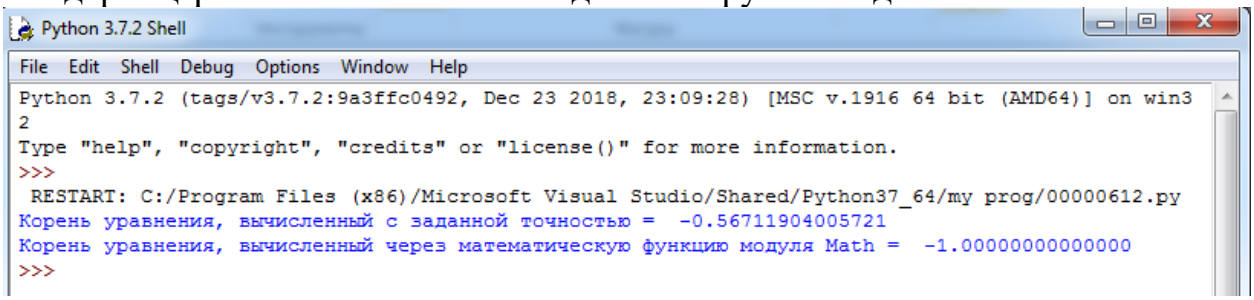

Сурет 61 – 6.2 тапсырманың шешу алгоритмның блок-схемасы

Есептің бағдарламмалық коды төменде көрсетілген:

```
from math import *
a=1 #Инициализация значения a
x=0
x1=-exp(x)
el=0.0001
while abs(x-a)>el:
    a=x
```

```
    x=-exp(a)
print("Корень уравнения, вычисленный с заданной точностью = ", '{0:.14f}'.format(x))
print("Корень уравнения, вычисленный через математическую функцию модуля Math = ", '{0:.14f}'.format(x1))
```

Бағдарлама жұмысының нәтижесі 62-ші суретте көрсетілген. Бұдан Math модулінің математикалық функциясы арқылы есептелген теңдеу түбірінің мәндерінің үлкен мәнге ие болатындығын көруге болады.




Сурет 62 –6.2 тапсырманың бағдарламма нәтижесі

**Тапсырма 6.3.** Берілген $\varepsilon$ дәлдікпен $e^x$ функциясының шамаланған мәнін есептейтін бағдарламаны жазыңыз.

**Решение.** $e^x$ функциясын шексіз дәрежелі қатармен көрсетуге болады:

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!} + \cdots$$

**EXP(x)** функциясының мәнін ε дәлдігімен есептеу үшін, берілген қатардың барлық мүшелерінің қосындысынды модулі бойынша дәлдіктен асатын болу керек яғни ε-нан. Сондықтан қосындыны есептеу үшін циклдік процесті ұйымдастырып, шарт орындалғанша оны қайталау қажет

$$\left|\frac{x^n}{n!}\right| > \varepsilon.$$

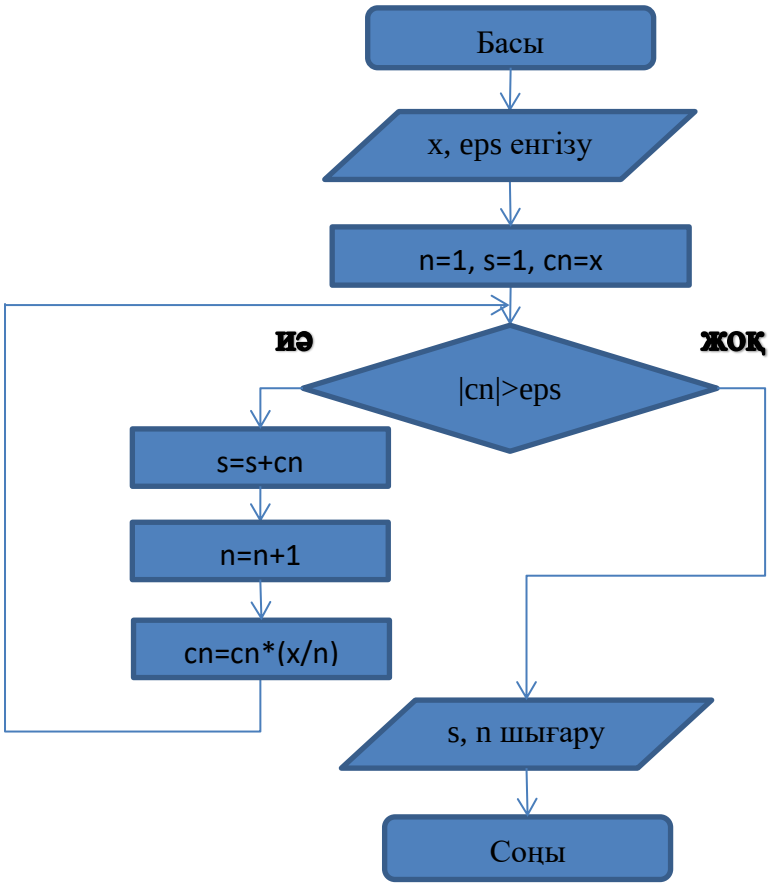
Жоғарыда келтірілген формуланы ескере отырып, қатардың көрші мүшелерін(оларды $C_n$ және $C_{n+1}$ деп белгілейміз) өзара қатынастың байланысты екенін көруге болады

$$C_{n+1} = C_n \cdot \frac{x}{n+1}, \qquad \text{где } n = 0, 1, 2, \cdots \text{причем } C_0 = 1.$$

Осы қатынасты қолдана отырып, рет ретімен қатарларды есептеуге болады. Бұл жағдайда факториалды және дәрежені есептеу операциялары қажет емес. Қарастырылған дәрежелі қатардың қасиеті олардың жартылай қосындыларын есептеу үшін өте қарапайым және өте тиімді (есептеу жылдамдығын тиімділеу мағынасында) алгоритмдерді ұйымдастыруға мүмкіндік береді.

Тағы бір айта кететін жайт, алдыңғы мән негізінде кейінгі мәндерін есептеуге мүмкіндік беретін формулалар **реккуренттік**(қайталанатын)

**қатынастар** деп аталады. Тапсырманы шешу алгоритмінің блок-схемасы 63-ші суретте көрсетілген.



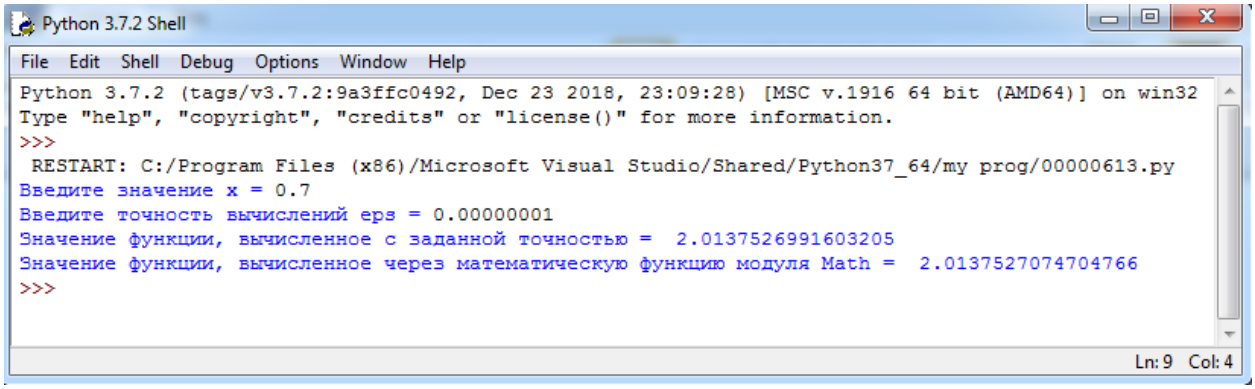

Сурет 63 – 6.3 тапсырманың шешу алгоритмның блок-схемасы

Төменде есепті шешуге арналған бағдарламмалық коды берілген

```
from math import *
x=float(input("Введите значение x = "))
eps=float(input("Введите точность вычислений eps = "))
s=1
n=1
cn=x
while abs(cn)>eps:
    s=s+cn
    n=n+1
    cn=cn*(x/n)
y=exp(x)
print("Значение функции, вычисленное с заданной точностью = ", s)
print("Значение функции, вычисленное через математическую функцию модуля Math = ", y)
```

64-ші суретте көрсетілген нәтижелерден функцияның қатардың қосындысы арқылы есептелген мәні жеткілікті дәлдікті беретінін көруге

болады. Ал ε мәні әлдеқайда аз болса, нәтиже EXP(x) операторы арқылы есептелген мәндерге қарағанда дәлірек болады.




Сурет 64 – 6.3 тапсырманың бағдарламмасының нәтижесі

**Тапсырма 6.4.** Кемімелі тізбекпен айнымалы таңбалы қатардың қосындысын берілген дәлдікпен **ε** есептеңіз.

$$\frac{(x-1)}{1!}-\frac{(x-1)^2}{2!}+\frac{(x-1)^3}{3!}-\cdots+(-1)^n\frac{(x-1)^{n+1}}{(n+1)!}+\cdots$$

**Шешімі.** Берілген ε дәлдікпен есептеу дегеніміз, қатардың келесі есептелген мүшесі ε санының абсолютті шамасынан аз болғанша, қатардың мүшелерін жинақтауды жалғастыру керек.

Көптеген тапсырмаларда келесі мүшені тікелей есептеу есептеу қиындықтарымен байланысты екенін ескеріңіз. Бұл жағдайда айнымалы мәнді ағымдағы қадамда $a_{n+1}=a_n\cdot q$ мәнін пайдаланып келесі қадамда есептеуге мүмкіндік беретін қайталанатын формуланы қолданған жөн - **q** үшін өрнекті $a_{n+1}/a_n$ бөлу арқылы алуға болады.

Тапсырмада берілген қатарлар үшін қайталанатын формуланы шығарайық. Формуланың $a_n$ мүшесі

$$a_n=(-1)^n\frac{(x-1)^{n+1}}{(n+1)!},$$

сонда $a_{n+1}$ мүшесінің формуласы

$$a_{n+1}=(-1)^{n+1}\frac{(x-1)^{n+1+1}}{(n+1+1)!}=(-1)^{n+1}\frac{(x-1)^{n+2}}{(n+2)!}.$$

$a_{n+1}$ мүшесін $a_n$ – ге бөліп **q** үшін келесі өрнекті аламыз

$$q=\frac{a_{n+1}}{a_n}=\frac{(-1)^{n+1}\frac{(x-1)^{n+2}}{(n+2)!}}{(-1)^n\frac{(x-1)^{n+1}}{(n+1)!}}=\frac{(x-1)^{n+2}\cdot(-1)^{n+1}\cdot(n+1)!}{(-1)^n\cdot(x-1)^{n+1}\cdot(n+2)!}=-\frac{(x-1)}{n+2}.$$

Осылайша, берілген қатар үшін рекуренттік формуланың жалпы түрі

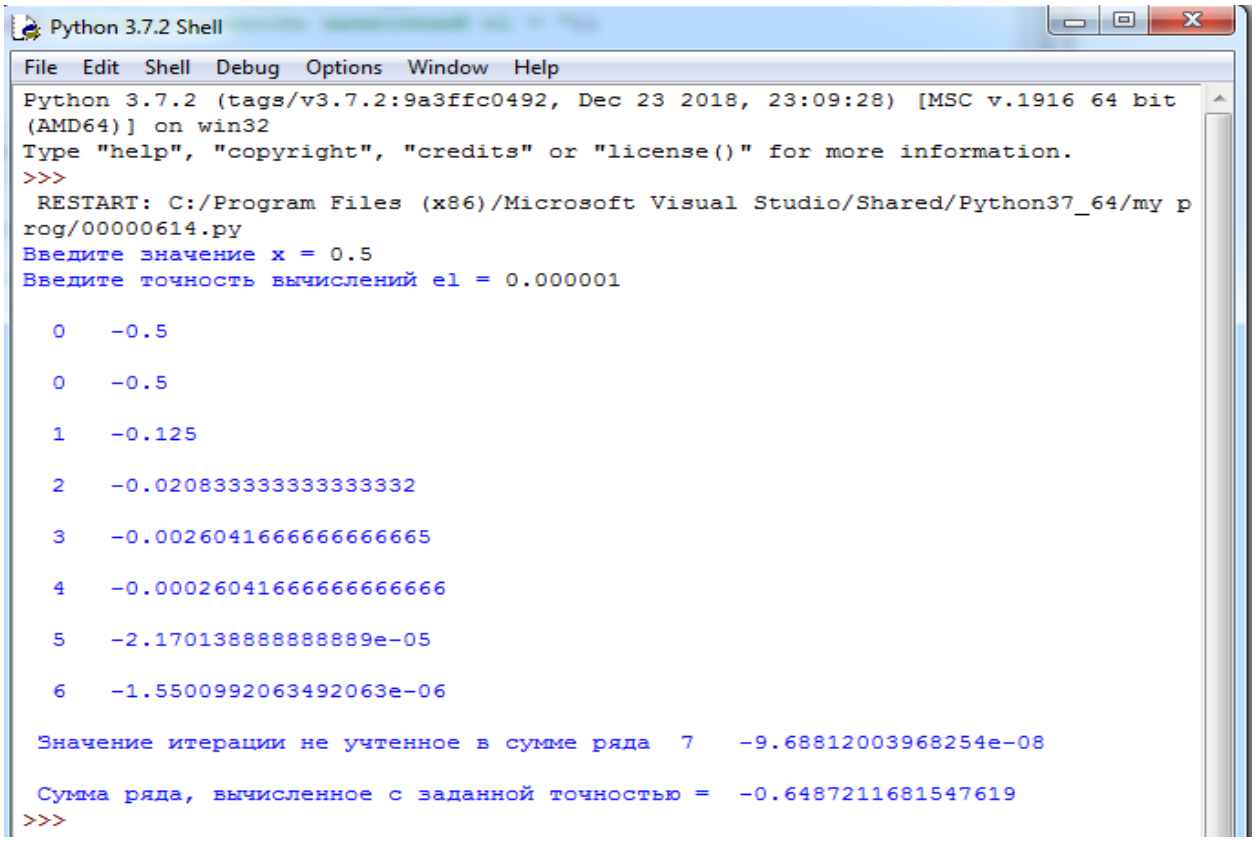
$$a_{n+1}=-a_n\frac{(x-1)}{n+2}.$$

Біздің жағдайымыз үшін **n** қатарының мүшесінің бастапқы мәні n=0 болады, өйткені бұл мәнді қатардың n мүшесінің формуласына ауыстырған кезде біз бірінші мүшенің мәнін аламыз **x-1** немесе **a=x-1** тең болатын.

$$a_n = (-1)^n \frac{(x-1)^{n+1}}{(n+1)!}$$

Программаның нәтижесі 65-ші суретте көрсетілген . Есептің шешімі төмендегі программа коды арқылы көрсетілген:

```
from math import *
x=float(input("Введите значение x = "))
el=float(input("Введите точность вычислений el = "))
n=0
a=x-1
s=0
print("\n ", n, " ", a)
while abs(a)>el:
    print("\n ", n, " ", a)
    s=s+a
    a=-a*(x-1)/(n+2)
    n=n+1
print("\n Значение итерации не учтенное в сумме ряда ", n, " ", a)
print("\n Сумма ряда, вычисленное с заданной точностью = ", s)
```




Сурет 65 –6.4 тапсырманың программа нәтижесі

**Тапсырма 6.5.** Итерациялық формуланы пайдаланып $x_{i+1} = x_i - \frac{f(x)}{f'(x)}$ мұнда $i = 0, 1, 2, \cdots$; $x_0 = 2{,}2$ теңдеу түбірін $\boldsymbol{f(x) = x^3 - 2x^2 + x - 3 = 0}$ берілген дәлдік арқылы $\varepsilon = 10^{-5}$ есептеу керек. Итерациялық процесс келесі шарт орындалғанда аяқталады $|x_{i+1} - x_i| < \varepsilon$.

**Шешімі.** Тапсырманы шешіп, табылған түбірді теңдеуге қойып шешімнің дұрыстығын тексереміз.

Ол үшін $f'(x)$ туындысын есептейміз

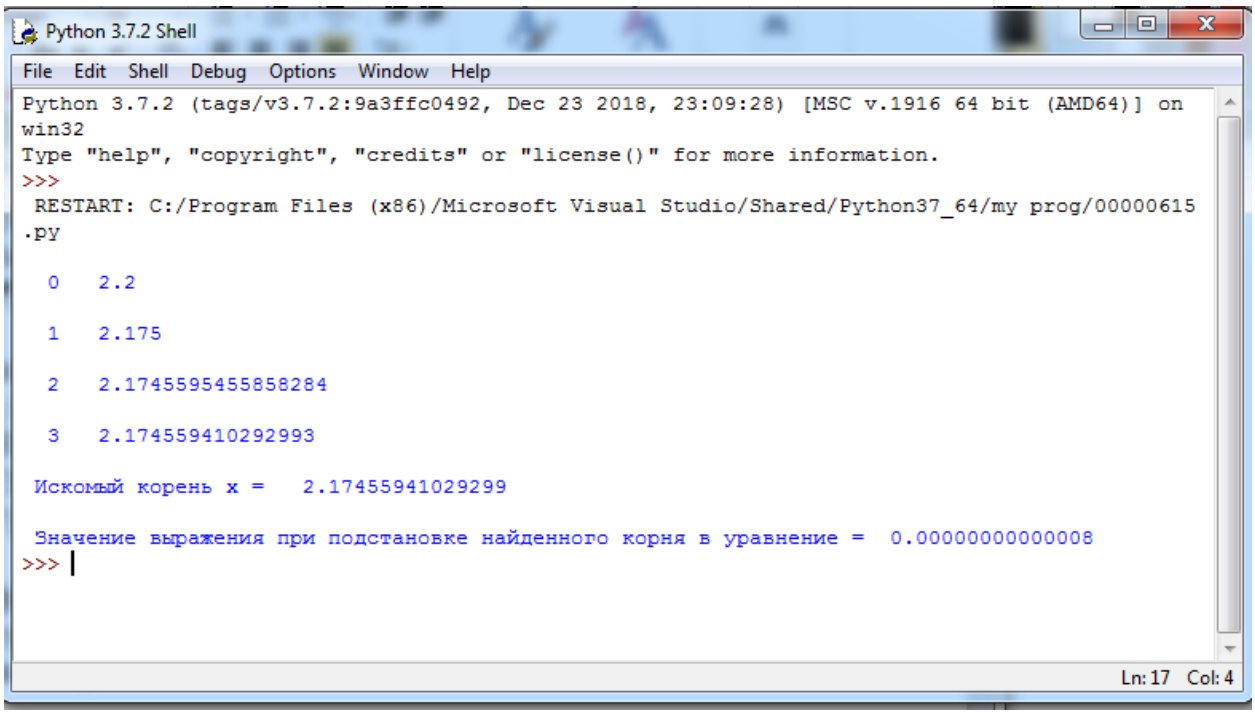
$$f'(x) = F(x) = 3x^2 - 4x + 1.$$

Келесі белгілеулерді пайдаланамыз, **x** - түбірге ағымдағы жуықтау, **a** - алдыңғы жуықтау, **f** - алдыңғы мән үшін f(x) функциясының мәні, **p** - алдыңғы мәні үшін f'(x) туындысы, **i** - теңдеудің түбіріне ағымдағы жуықтау нөмірімен сәйкес келетін итерация нөмірі, **y**-берілген дәлдікпен табылған теңдеудің түбірі үшін f(x) функциясының мәні.

Егер ағымдағы және алдыңғы түбір мәндерінің арасындағы модуль айырмасы **ε** дәлдігінен аз болса, яғни біздің жағдайымыз үшін |x-a|<ε орындалса, берілген дәлдік **ε** қамтамасыздандырылған, Тапсырманы шешуге жауап беретін бағдарлама коды төменде көрсетілген.

```
from math import *
x=2.2
el=0.00001
i=0
a=1
print("\n ", i, " ", x)
while abs(x-a)>el:
   a=x
   f=pow(x, 3)-2*a*a+a-3
   p=3*a*a-4*a+1
   x=a-f/p
   i=i+1
   print("\n ", i, " ", x)
y=pow(x, 3)-2*x*x+x-3
print("\n Искомый корень x =  ", '{0:.14f}'.format(x))
print("\n Значение выражения при подстановке найденного корня в уравнение = ", '{0:.14f}'.format(y))
```

Осылайша, 66-шы суретте келтірілген нәтижелерден **y** мәні табылған түбір теңдеуінде алмастыру ретінде табылғаны анық.




Сурет 66 – 6.5 тапсырманың программасының нәтижесі