4 РАЗВЕТВЛЯЮЩИЙСЯ АЛГОРИТМ

Ранее мы рассмотрели, как вводить данные и присваивать значения переменным. Теперь остановимся на вопросе организации различных потоков выполнения приложения в зависимости от ситуации, которая складывается в ходе работы программы. Для этого введем понятие разветвляющегося алгоритма.

Алгоритм называется разветвляющимся, если последовательность выполнения шагов алгоритма изменяется в зависимости от выполнения некоторых условий. Условие - это логическое выражение, которое может принимать одно из двух значений: «ДА» - если условие верно (истинно), и «НЕТ» - если условие неверно (ложно).

Разветвляющийся алгоритм можно реализовать в программах с помощью простого, сокращенного, составного операторов, а также конструкции многозначных ветвлений. Рассмотрим ниже более подробно эти варианты.

4.1 Простой условный оператор

Общий вид в алгоритме конструкции простого условного оператора представлен на рисунке 30.

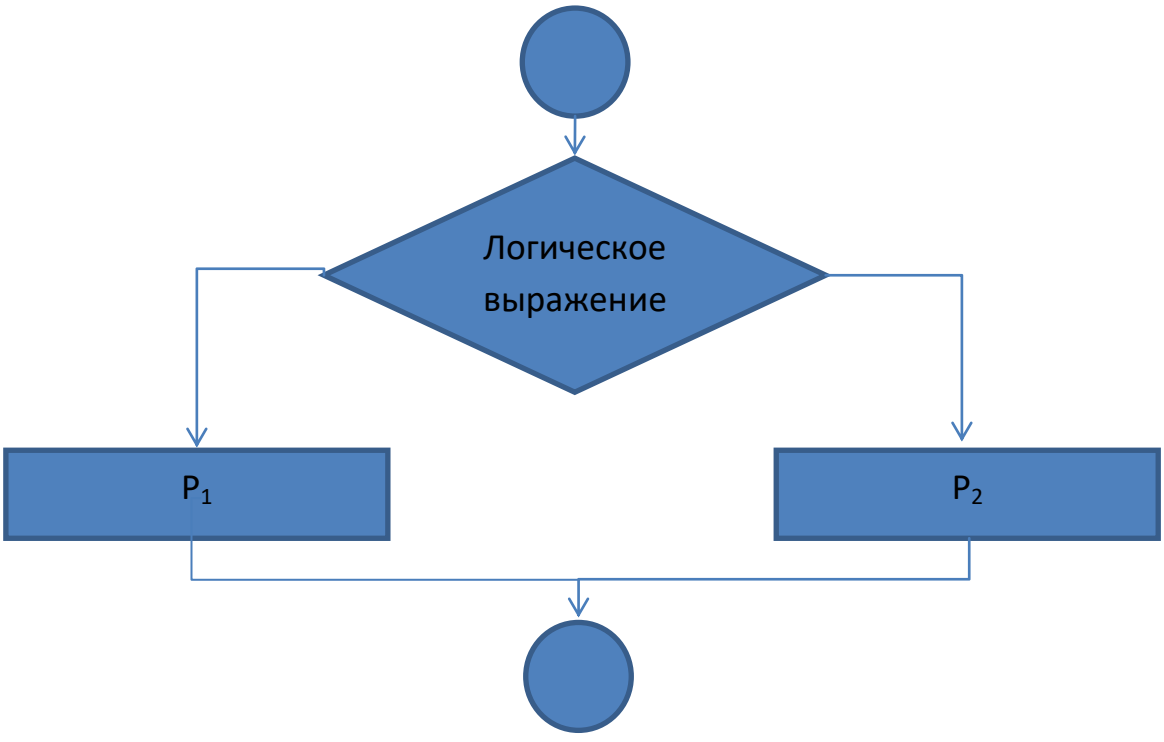

Рисунок 30 – Блок-схема условного оператора

Синтаксис простого условного оператора следующий:

if Логическое выражение:
    P_1

else:
   P_2

где **if** (если), **else** (иначе) - зарезервированные слова, а P_1, P_2 - операторы.

Простой условный оператор работает по следующему алгоритму: сначала вычисляется логическое выражение. Если результат есть **true** (истина), то выполняется оператор $\mathbf{P_1}$ а оператор $\mathbf{P_2}$ пропускается. Если результат есть **false** (ложь), то выполняется оператор $\mathbf{P_2}$, а оператор $\mathbf{P_1}$ пропускается.

Например, в следующем коде в зависимости от результата сравнения двух переменных выводится тот или иной ответ. Оператор **if** - это блочный оператор, поэтому отступы в коде обязательны. Вход в блок операторов осуществляется двоеточием:

```
a=int(input("Введите значение a="))
b=int(input("Введите значение b="))
if a<b:
   print("Тест 1 ветви")
else:
   print("Тест 2 ветви")
```

На рисунке 31 представлен скриншот примера выполнения простого условного оператора при значениях **a=3, b=2**.

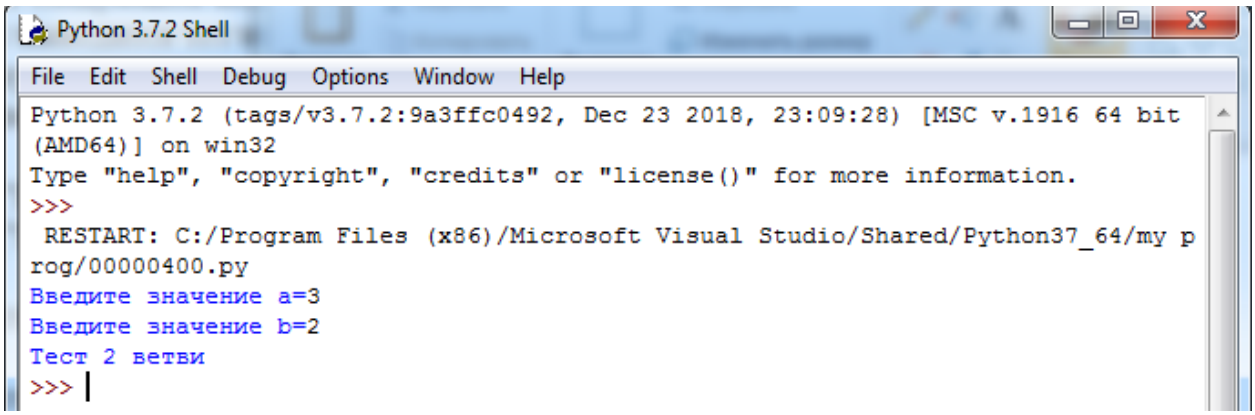

Рисунок 31 – Скриншот выполнения простого условного оператора при значениях a=3, b=2

4.2 Сокращенный условный оператор

Если необходимо выполнить некоторое действие только при истинности проверяемого условия, то в таком случае применяется сокращенный условный оператор. Общий вид в алгоритме конструкции

сокращенного условного оператора представлен на рисунке 32.

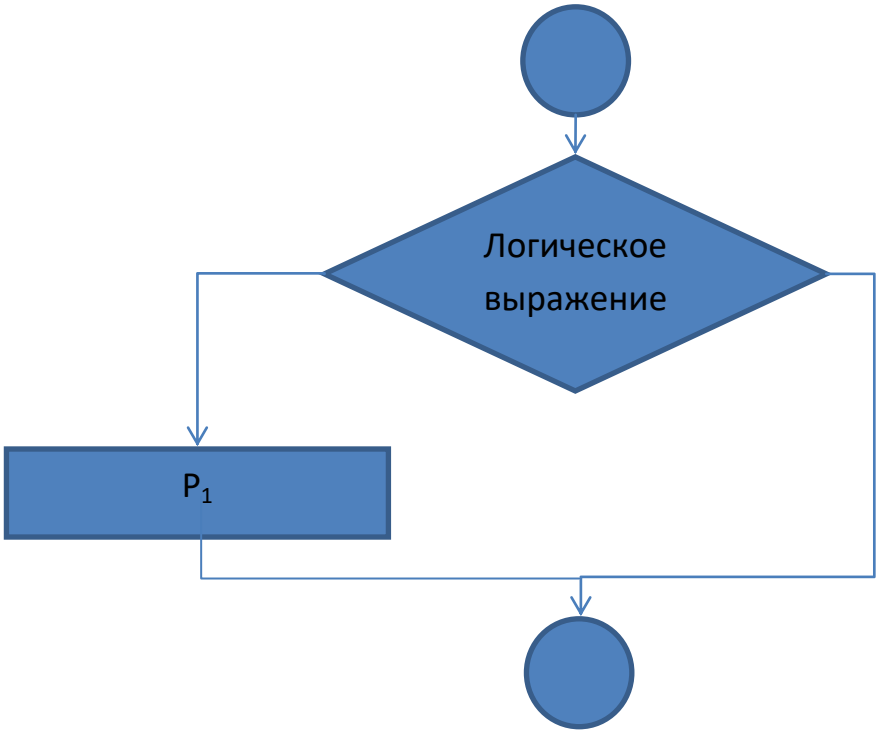

Рисунок 32 – Блок-схема сокращенного условного оператора

Синтаксис сокращенного условного оператора следующий:

if Логическое выражение:
    P1

где **if** (если) - зарезервированное слово, а P_1 - оператор.

Например, в листинге ниже при истинности логического выражения выводится соответствующее сообщение «Тест 1 ветви»:

```
a=int(input("Введите значение a="))
b=int(input("Введите значение b="))
if a<b:
    print("Тест 1 ветви")
```

4.3 Составной условный оператор

Если при некотором условии надо выполнить определенную последовательность операторов, то их объединяют в один составной оператор. Общий вид в алгоритме конструкции составного условного оператора представлен на рисунке 33.

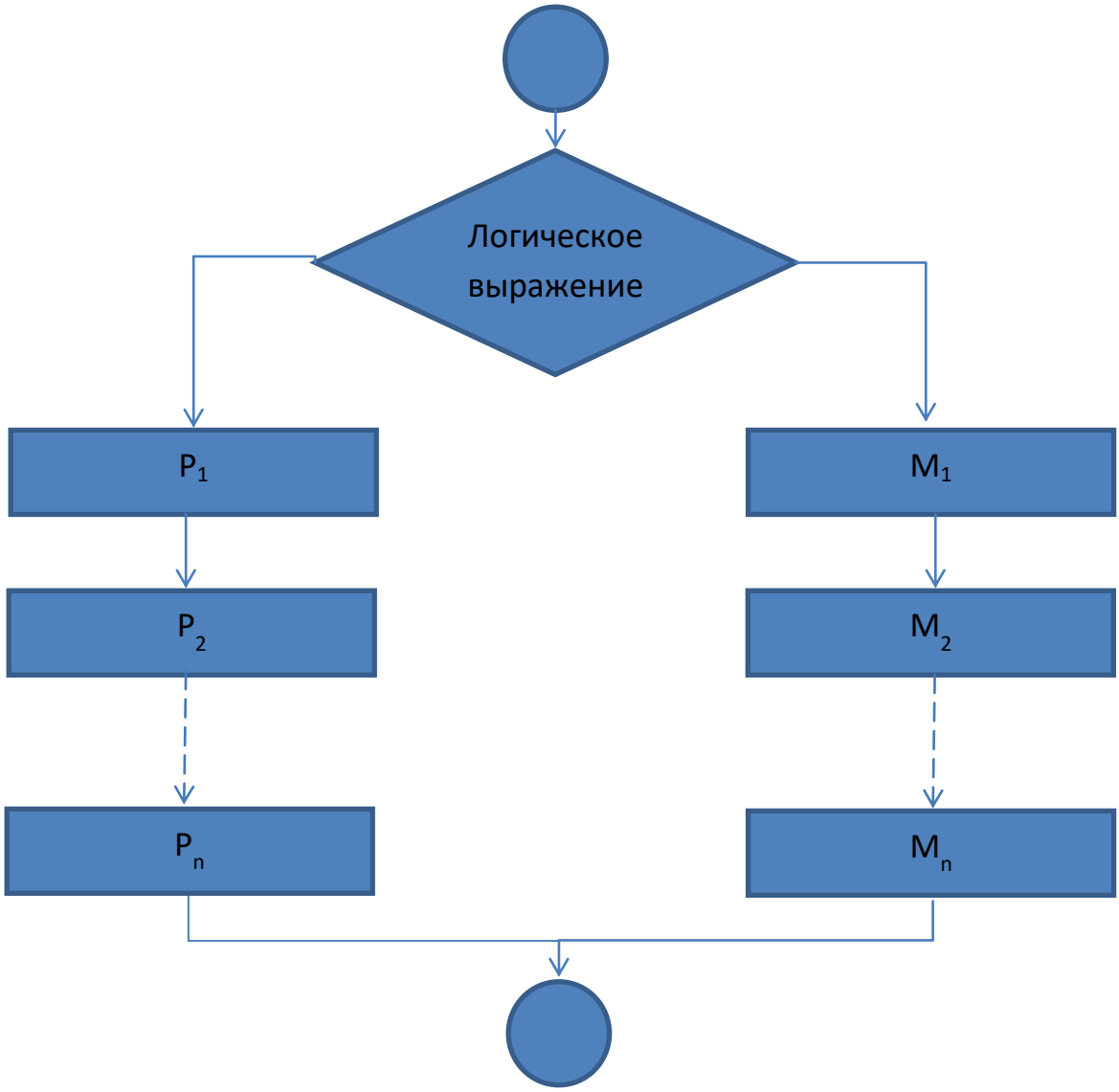

Рисунок 33 – Блок-схема составного условного оператора

Синтаксис составного условного оператора следующий:

if Логическое выражение:
    P_1
    P_2
    .
    .
    P_n
else:
    M_1
    M_2
    .
    .
    M_n

где **if**, **else** - зарезервированные слова, а P_1, P_2,…, P_n, M_1, M_2,…, M_n -

операторы.

Например, в листинге ниже в каждой ветви условного оператора мы хотим выполнить по два оператора. Тогда каждый из них мы должны сместить вправо ровно на четыре пробела от начала строки:

```
a=int(input("Введите значение a="))
b=int(input("Введите значение b="))
if a<b:
    a=a+b
    print("Сумма двух чисел a+b=", a)
else:
    a=a*b
    print("Произведение двух чисел a*b=", a)
```

4.4 Многозначные ветвления

Очень часто приходиться выбирать путь решения задачи не из двух, а из нескольких возможных. В программировании данный ход можно реализовать, используя несколько условных операторов. Общий вид в алгоритме конструкции многозначных ветвлений представлен на рисунке 34.

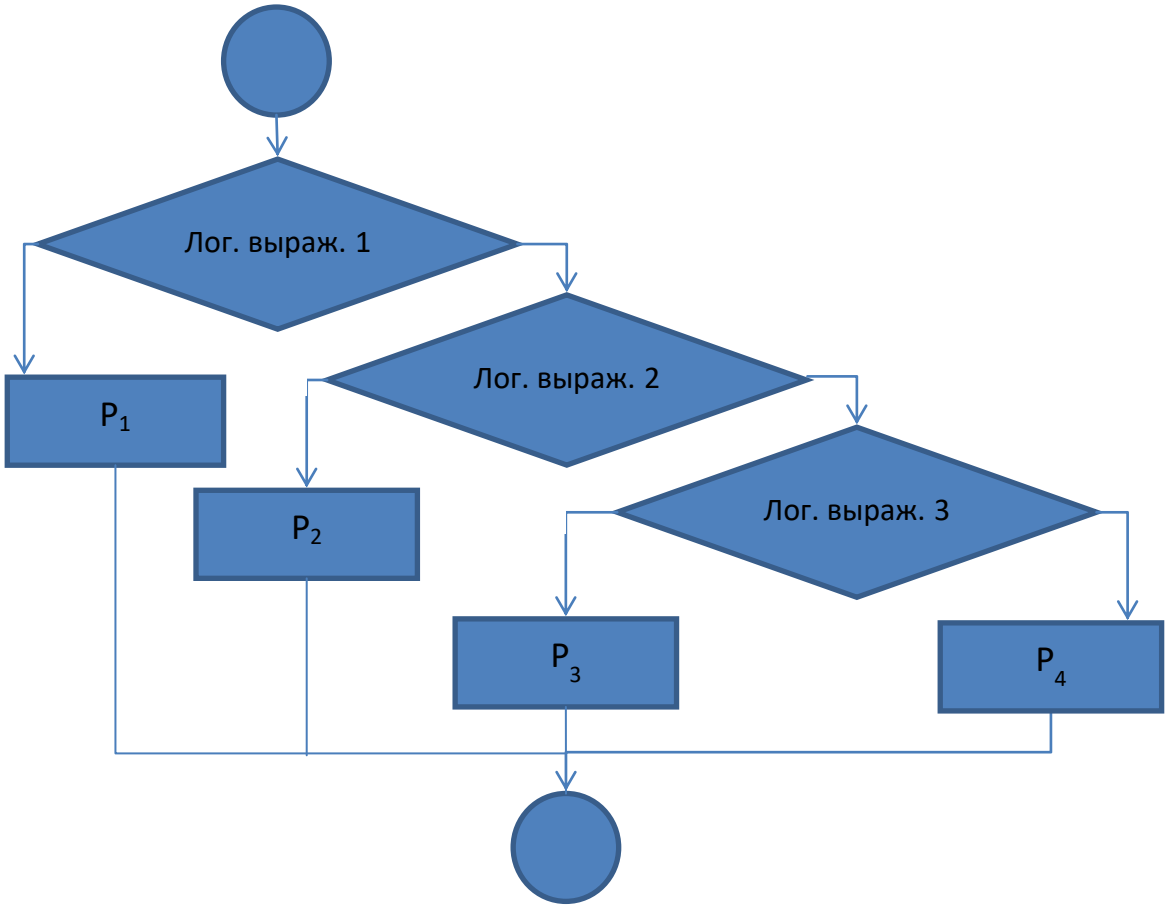

Рисунок 34 – Блок-схема конструкции многозначных ветвлений

Синтаксис условного оператора в конструкции многозначных ветвлений следующий:

if Логическое выражение 1:
   P_1
elif Логическое выражение 2:
   P_2
elif Логическое выражение 3:
   P_3
else:
   P_4

где **if**,**elif**, **else** - зарезервированные слова, а P_1, P_2, P_3, P_4 - операторы.

Алгоритм работы такой конструкции состоит в следующем. Если **Логическое выражение 1** истинно, то выполняется оператор или блок операторов, следующих в данной ветви, в противном случае этот оператор или блок пропускается. Если логическое выражение, следующее за оператором **if**, ложно, то анализируется **Логическое выражение 2**, следующее за оператором **elif**. Если оно истинно, то выполняется оператор или блок операторов, следующих в данной ветви, в противном случае этот оператор или блок пропускается. Операторы, следующие за последним **else**, выполняются лишь в том случае, если ложны все предыдущие логические выражения. Условные операторы **if** в такой конструкции называются **вложенными**.

Например, далее в листинге показан процесс тестирования трех ветвей программы. Пользователь может менять исходные данные, которые будут находиться в ячейках **a** и **b**, и всякий раз получать тот или иной результат.

```
a=int(input("Введите значение a="))
b=int(input("Введите значение b="))
if a<b:
   a=a+b
   print("Сумма двух чисел a+b=", a)
elif a==b:
   a=a*b
   print("Произведение двух чисел a*b=", a)
else:
   a=a-b
   print("Разность двух чисел a-b=", a)
```

4.5 Алгоритмы поиска максимального и минимального элементов

Рассмотрим алгоритмы нахождения максимального и минимального значений, которые будут востребованы при дальнейшем изучении языка

программирования Python.

Задача 4.1. Найдите максимальное из двух чисел.

Решение. Разработка алгоритма решения задачи представлена на рисунке 35.

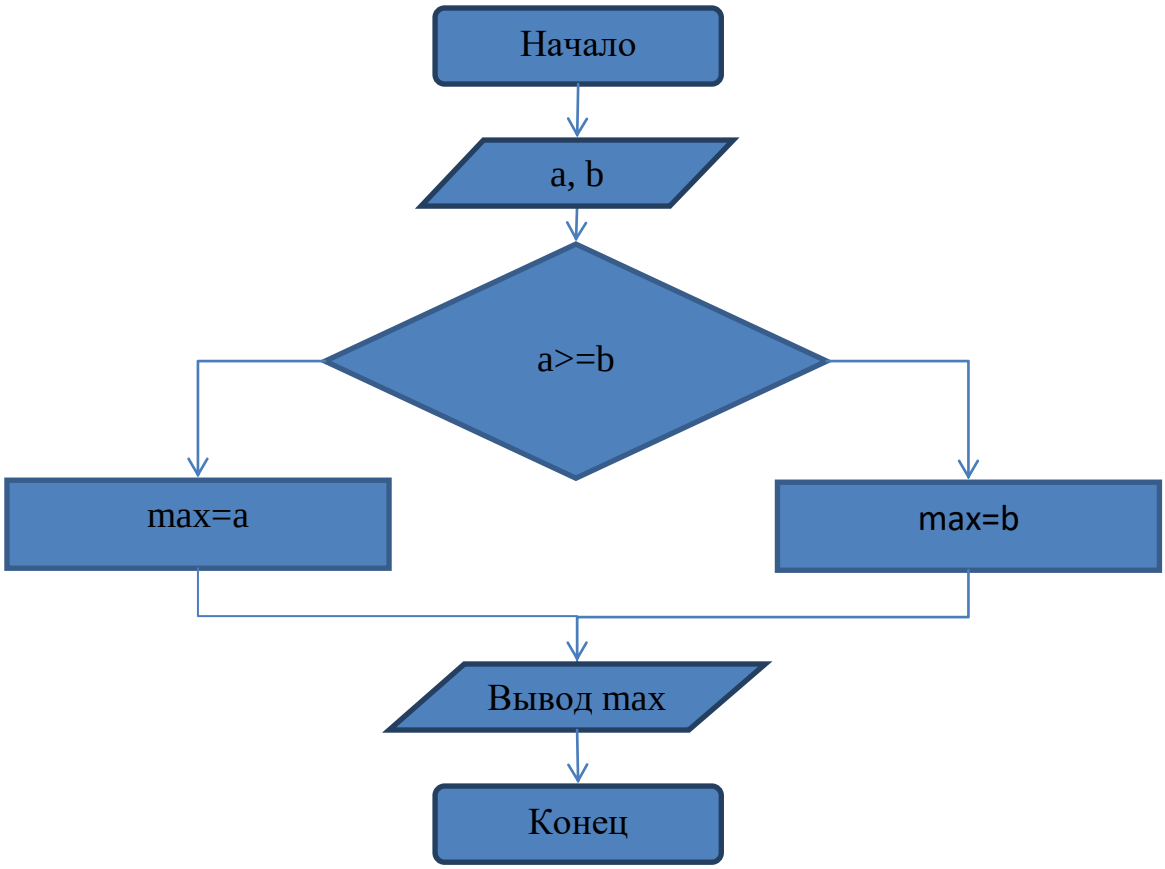

Рисунок 35 – Алгоритм нахождения максимального из двух чисел

Листинг приведем ниже:

```
a=int(input("Введите значение a="))
b=int(input("Введите значение b="))
if a>=b:
   max=a
else:
   max=b
print("Максимальное из двух чисел max=", max)
```

Задача 4.2. Найдите минимальное из трех чисел.

Решение. Алгоритм нахождения максимального или минимального элемента может быть запрограммирован несколькими способами. Фрагмент разработки алгоритма решения задачи (первый способ) представлен на рисунке 36. Очевидно, что при большом количестве чисел, из которых нужно осуществить выбор экстремального значения, алгоритм станет очень громоздким и запутанным.

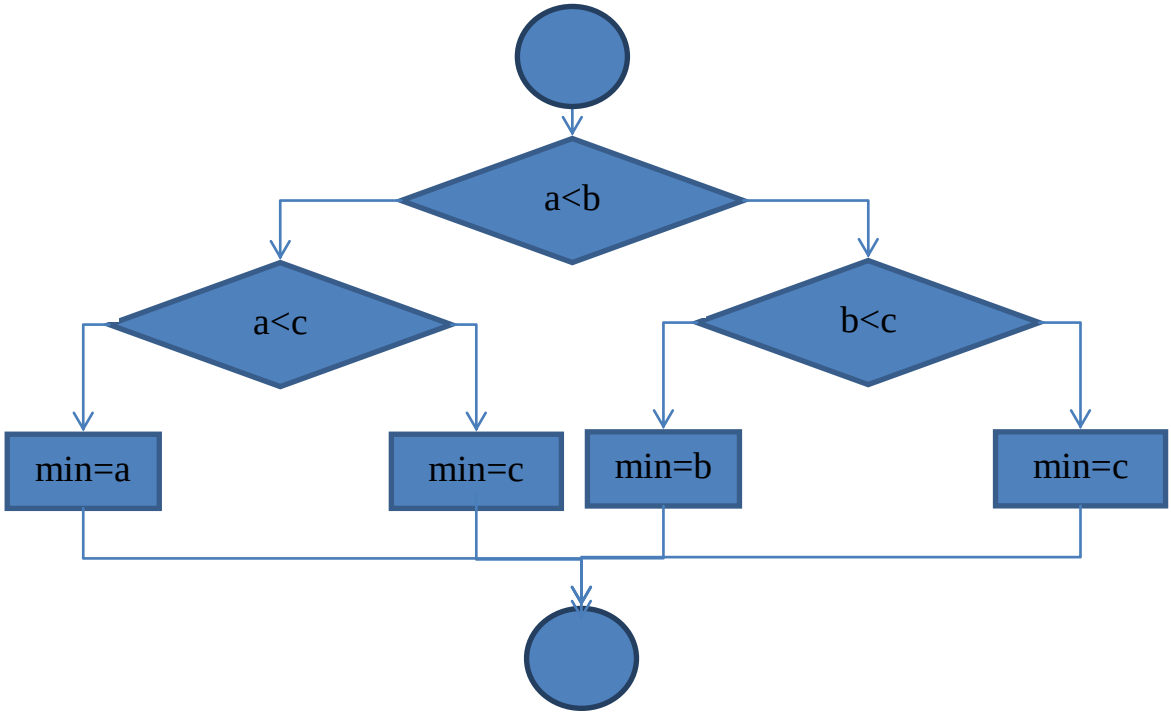

Рисунок 36 – Алгоритм нахождения минимального из трех чисел (первый способ)

Фрагмент второго способа, представленный на рисунке 37, более простой и наглядный. Сравнив два числа между собой и определив минимальное из них, каждое последующее число будем сравнивать с тем, которое уже находится в ячейке **min**, и осуществлять перезапись ячейки оператором присваивания.

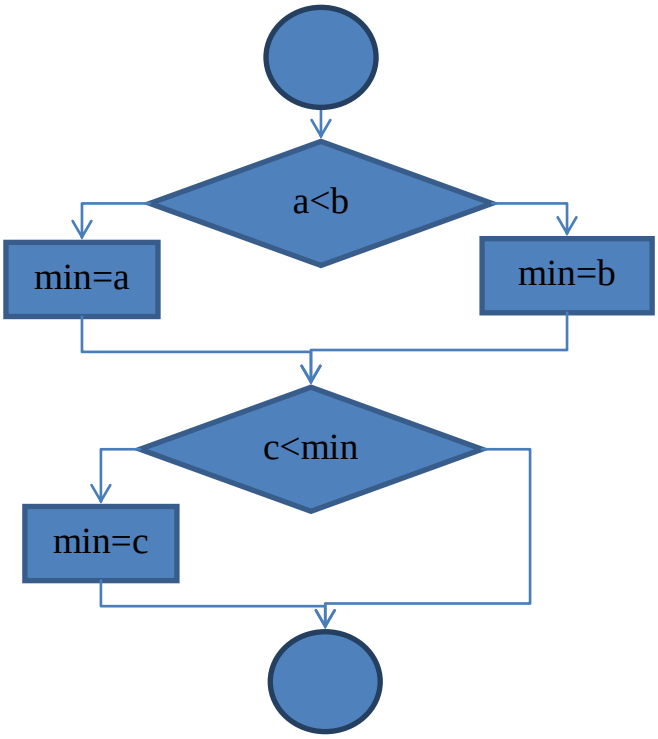

Рисунок 37 – Фрагмент алгоритма нахождения минимального из трех чисел (второй способ)

Фрагмент разработки алгоритма решения задачи (третий способ) представлен на рисунке 38.

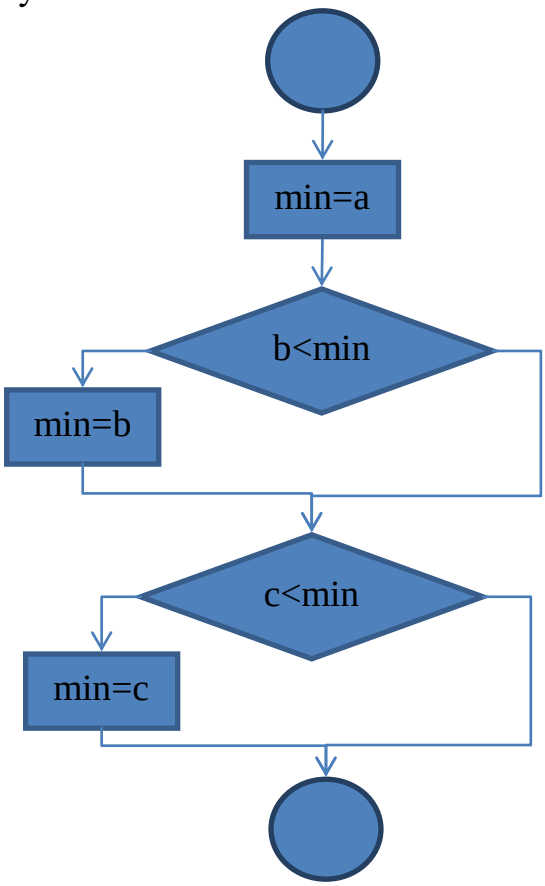

Рисунок 38 – Фрагмент алгоритма нахождения минимального из трех чисел (третий способ)

Листинг третьего способа нахождения минимального из трех чисел приведем ниже:

```
a=int(input("Введите значение a="))
b=int(input("Введите значение b="))
c=int(input("Введите значение c="))
min=a
if b<min:
    min=b
if c<min:
    min=c
print("Минимальное из трех чисел min=", min)
```

На основе рассмотренных алгоритмов решим следующую задачу.

Задача 4.3. Вычислите значение функции y, если дана следующая функция.

$$y = \begin{cases} min(a_1, a_2, a_3), & \text{если } -1 < x < 1 \\ max\left(b_1, b_2, min(c_1, c_2)\right), & \text{если } x \geq 1 \\ 1, & \text{если } x \leq -1 \end{cases}$$

Решение. Разработка алгоритма решения задачи представлена на рисунке 39.

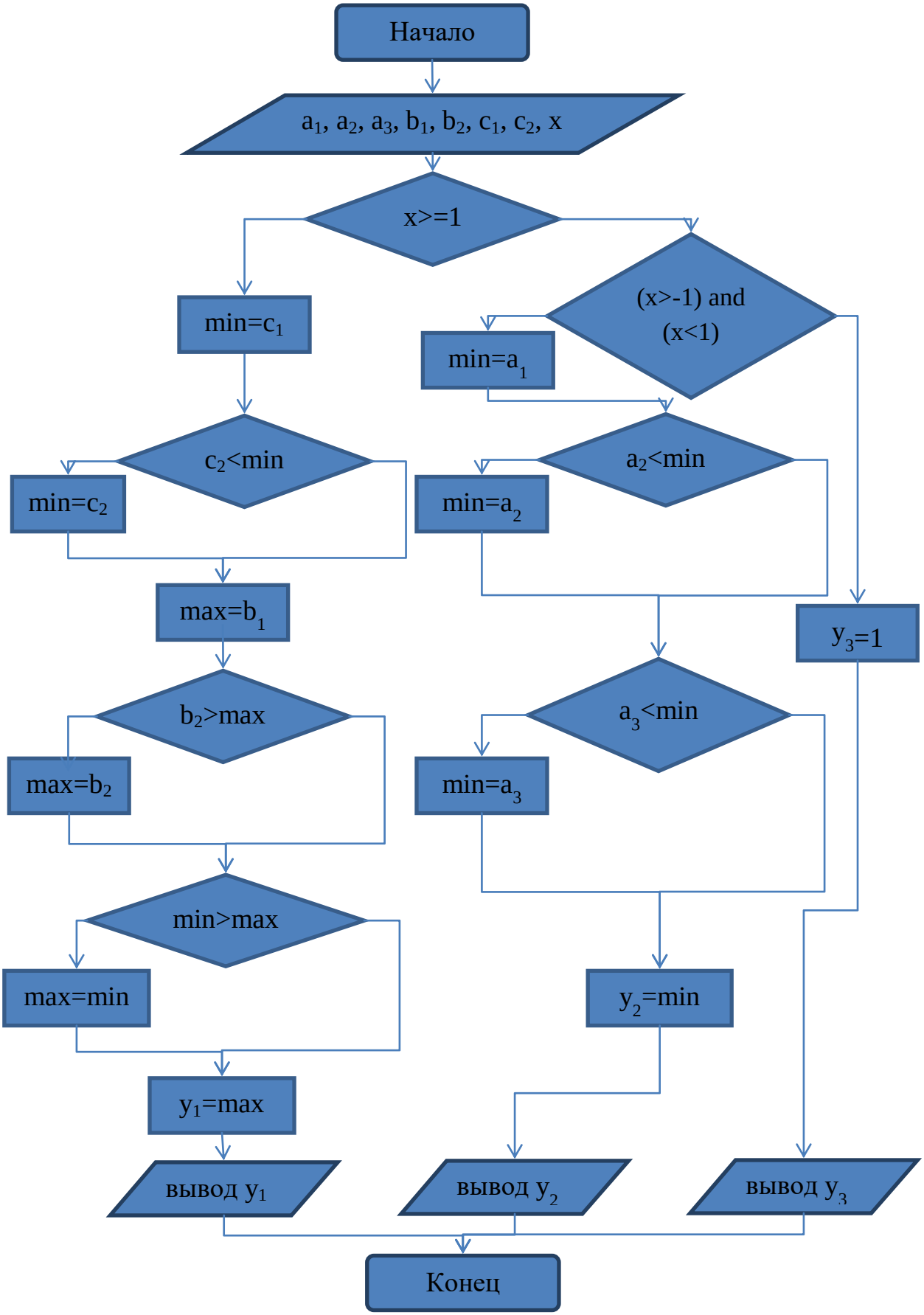

Рисунок 39 – Алгоритм решения задачи 4.3

Ниже приведен код программы, отвечающий за решение задачи. Еще раз стоит обратить внимание на отступы, сделанные в листинге. Каждый оператор **if** имеет свое вложение операторов согласно алгоритму решения задачи и, соответственно, отступ вправо. Как только условный оператор закончил свою работу, следующий оператор (операторы) пишется (пишутся) без отступа, как, например, в следующем фрагменте:

```
if min>max:
    max=min
y1=max
print("Тест 1 ветки")
print("y1=", y1)
```

После выполнения оператора **max=min** условный оператор **if** закончил свою работу, далее операторы (выделены курсивом) уже не относятся к оператору **if** и при наборе программы смещаются влево.

```
a1=int(input("Введите значение a1="))
a2=int(input("Введите значение a2="))
a3=int(input("Введите значение a3="))
b1=int(input("Введите значение b1="))
b2=int(input("Введите значение b2="))
c1=int(input("Введите значение c1="))
c2=int(input("Введите значение c2="))
x=float(input("Введите значение x="))
if x>=1:
    min=c1
    if c2<min:
        min=c2
    max=b1
    if b2>max:
        max=b2
    if min>max:
        max=min
    y1=max
    print("Тест 1 ветки")
    print("y1=", y1)
elif (x>-1) and (x<1):
    min=a1
    if a2<min:
        min=a2
    if a3<min:
        min=a3
    y2=min
    print("Тест 2 ветки")
```

```
    print("y2=", y2)
else:
    y3=1
    print("Тест 3 ветки")
    print("y3=", y3)
```

4.6 Пояснение примеров заданий на применение разветвляющегося алгоритма

Задание 4.1.. Каким будет значение переменной **s** после выполнения группы операторов?

```
n=2.5
f=0.5
d=True
s=0
if n<f:
    s=12
if f>=n:
    s=28
if d:
    s=39
print("s=", s)
```

Решение. В этом упражнении значение переменной **s** равно **39**. Рассуждать надо следующим образом. Первые два логических выражения ложны, поэтому проверяется третье логическое выражение, в ячейке **d** хранится значение **True** (Истина), поэтому выполняется оператор s=39.

Задание 4.2. Каким будет значение переменной j после выполнения группы операторов?

```
w=3
p=5
j=3.5
if (j<p) and (j>w):
    j=j+0.5
    j=j+10
else:
    j=11
print("j=", j)
```

Решение. Рассуждать нужно следующим образом. Подставив значения переменных **w**, **p**, **j** и проверив логическое выражение, получаем, что его

значение есть **True** (Истина). Следовательно, выполняется оператор: **j=j+0.5**, а затем оператор **j=j+10**. Ответ в упражнении: значение переменной **j=14.0**.

Задание 4.3. Каким будет значение переменной **j** после выполнения группы операторов?

```
j=3
k=15
m=20
if j<=k:
   if m>k:
     j=k%2
     j=j%3
   else:
      j=10
print("\n Значение j=", j)
```

Решение. Подставив значения переменных **k**, **m**, **j** и проверив оба логических выражения, получаем, что их значения есть **True** (Истина). Следовательно, выполняется оператор: **j=k%2**, а затем: **j=j%3**. Ответ в упражнении: значение переменной **j** равно **1**.

Задание 4.4. Каким будет значение переменной **j** после выполнения группы операторов?

```
j=6
k=6
if j>k:
   j=j+2
   j=j+3
else:
   j=k-3
   j=j+4
print("\n Значение j=", j)
```

Решение. Подставив значения переменных **k** и **j**, проверив логическое выражение, получаем, что его значение есть **False** (Ложь). Следовательно, выполняются операторы: **j=k-3** и **j=j+4**. Ответ в упражнении: значение переменной **j** равно **7**.

Задание 4.5. Каким будет значение переменной j после выполнения группы операторов?

```
w=3
p=5
```

```
j=3.5
if (j<p) and (j>w):
   j=j+0.5
   j=j+12
else:
   j=11
print("\n Значение j=", j)
```

Решение. Подставив значения переменных, получим, что два простых условия **(j<p)** и **(j>w)** оказываются истинными, следовательно, и все логическое выражение имеет значение **True**. Таким образом, выполняется оператор: **j=j+0.5**, а затем: **j=j+12**. После выполнения в ячейке **j** будет число **16.0**.

Задание 4.6. Каким будет значение переменной **j** после выполнения условного оператора?

```
j=7
k=7
f=10
if j>=k:
   if f<=k:
      k=30%5
      j=(j%2)*k
   else:
      j=1
else:
   j=0
print("\n Значение j=", j)
```

Решение. Подставив значения переменных в первое логическое выражение, получим результат: **True** (Истина). Подставив значения переменных во второе логическое выражение, получим, что результат: **False** (Ложь), следовательно, выполняется оператор в ветви **else** (отступ вправо у оператора соответствует ветви **else** для оператора **if** первого уровня, следовательно, **j=0** не выполнится, так как в первом логическом выражении был результат **True**). Таким образом, правильный ответ на вопрос, поставленный в упражнении, такой: в ячейке **j** будет находиться значение, равное **единице**.

Задание 4.7. Каким будет значение переменной **f** после выполнения группы операторов?

```
x=55
y=5e1
```

```
d=False
f=0
if d:
   f=x%2
if x<y:
   f=x
if x>y:
   f=int(2.9)
print("\n Значение f=", f)
```

Решение. Проверив первое логическое выражение, получим, что результат Ложь - в ячейке **d** хранится значение **False**. Поэтому оператор **f=x%2** не выполняется. Для того чтобы определить, истинно или ложно второе логическое выражение, надо знать, что число **5e1** записано в форме с плавающей точкой и равно **50**. Следовательно, второе логическое выражение ложно, и оператор **f=x** не выполняется. Проверив третье логическое выражение, убеждаемся в том, что оно истинно, так как **55>50**. В операторе **f=int(2.9)** используется преобразование к целому типу. Таким образом, после выполнения условного оператора в ячейке **f** будет число **2**.

Задание 4.8. Каким будет значение переменной **j** после выполнения группы операторов?

```
j=10
k=10
if j>k:
   j=k-3
else:
   k=k-3
   j=k-3
print("\n Значение j=", j)
```

Решение. Проверив логическое выражение, убеждаемся в том, что оно ложно. Следовательно, выполняются операторы в ветви **else**. После выполнения оператора **k=k-3** в ячейке **k** оказывается значение **7**, а после выполнения оператора **j=k-3** в ячейке **j** находится число **4**. Ответ в упражнении: **j** равно **4**.

4.7 Примеры решения задач

Задача 4.4. Вычислите значение функции **y**:

$$y = \begin{cases} sinx, & \text{если } x \geq 1 \\ cosx, & \text{если } x < 1 \end{cases}.$$

Решение. Блок-сема алгоритма решения задачи представлена на рисунке 40.

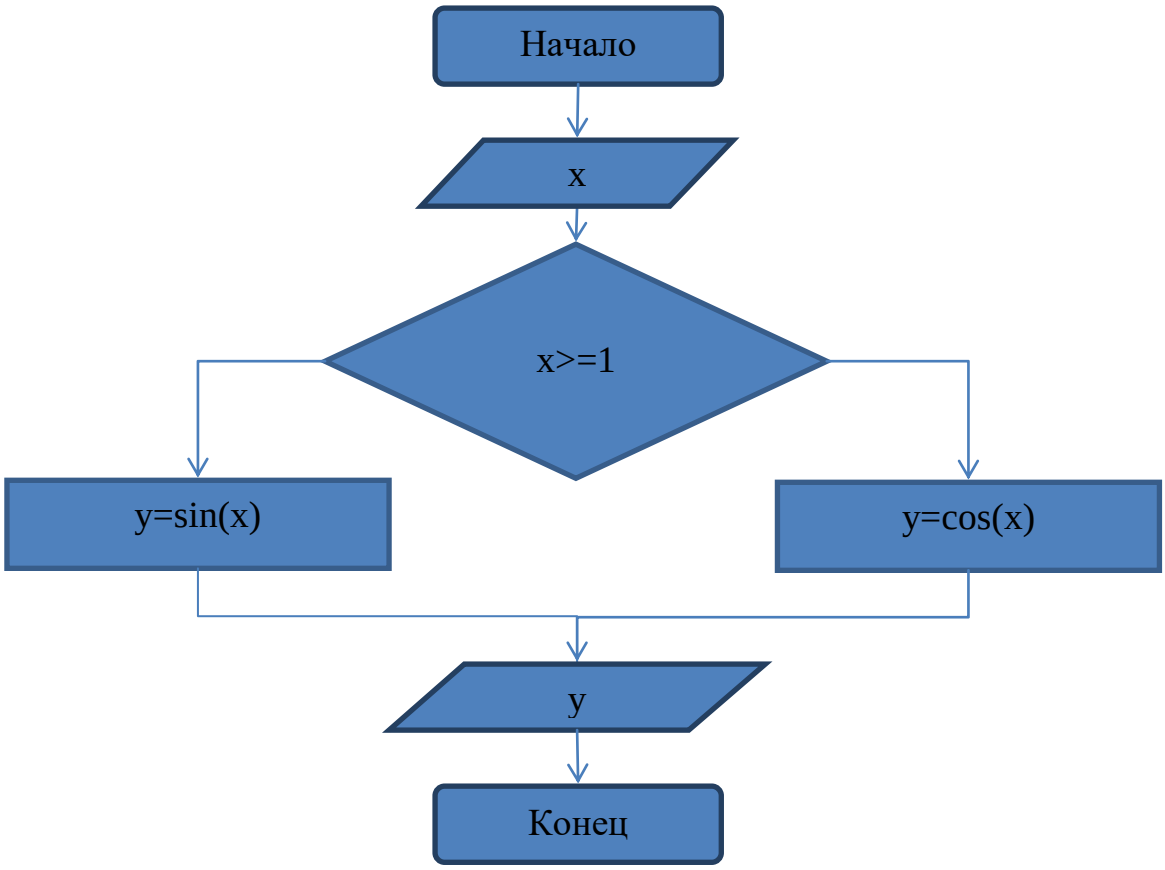

Рисунок 40 – Алгоритм решения задачи 4.4

В листинге приведен код программы, отвечающий за решение задачи:

```
from math import *
x=float(input("Введите значение x="))
if x>=1:
    y=sin(x)
else:
    y=cos(x)
print("\n Результат:", y)
```

Задача 4.5. Вычислите значение функции **y**:

$$y = \begin{cases} sinx, & \text{если } x < 0 \\ cosx, & \text{если } 0 \le x \le 1. \\ tgx, & \text{если } x > 1 \end{cases}$$

Решение. Блок-сема алгоритма решения задачи представлена на рисунке 41.

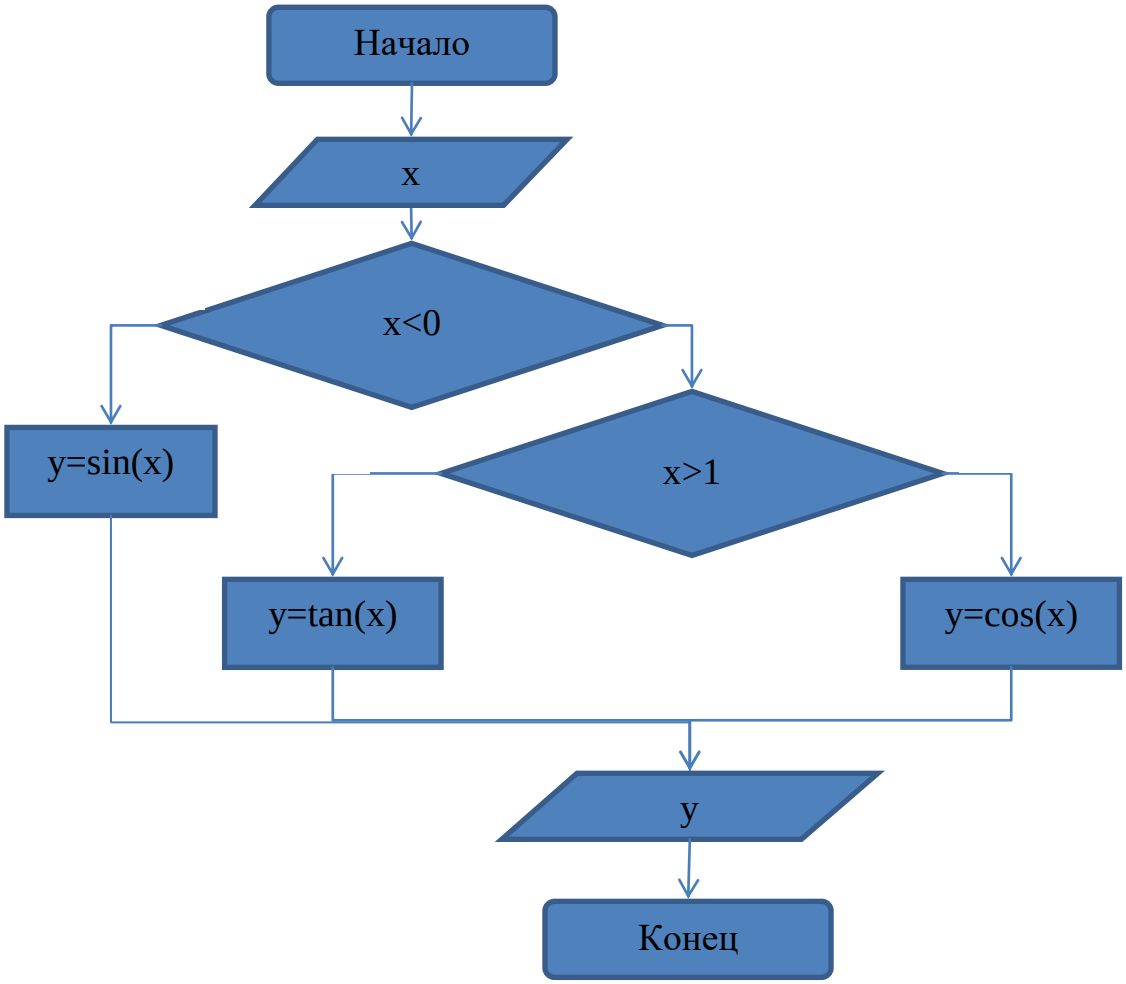

Рисунок 41 – Алгоритм решения задачи 4.5

В листинге приведен код программы, отвечающий за решение задачи:

```
from math import *
x=float(input("Введите значение x="))
if x<0:
    y=sin(x)
elif x>1:
    y=tan(x)
else:
    y=cos(x)
print("\n Результат:", y)
```

4.8 Контрольные вопросы

1. Что называется разветвляющимся алгоритмом?
2. Как записывается простой условный оператор в блок-схемах?
3. Как записывается простой условный оператор в программах?
4. Как работает простой условный оператор?
5. Как записывается сокращенный условный оператор в блок-схемах?

6. Как записывается сокращенный условный оператор в программах?

7. Как работает сокращенный условный оператор?

8. Как записывается составной условный оператор в блок-схемах?

9. Как записывается составной условный оператор в программах?

10. Как работает составной условный оператор?

11. Как записываются многозначные ветвления в блок-схемах?

12. Как записываются многозначные ветвления в программах?

13. Как работает условный оператор Дпри проверке нескольких условий?

4.9 Задачи для самостоятельного решения

1. Разработайте алгоритм и программу вычисления значения **y** по формуле

$$Y = \begin{cases} X, & \text{если } X \le 0 \\ 2X, & \text{если } X > 0 \end{cases}.$$

2. Разработайте алгоритм и программу, которая суммирует только положительные значения, введенные пользователем с клавиатуры.

3. Даны значения трех переменных. Напишите последовательность операторов, подсчитывающих количество значений, которые были равны нулю. Разработайте алгоритм и программу решения данной задачи.

4. Разработайте алгоритм и программу, которая выводит на экран три типа ответа: «Вы имеете удовлетворительную успеваемость», «Вы имеете хорошую успеваемость», «Вы имеете отличную успеваемость», в зависимости от введенного пользователем числа.

5. Разработайте алгоритм и программу решения следующей задачи: найти корни квадратного уравнения по формулам

$$D = b^2 - 4ac,$$

$$x_1 = \frac{-b+\sqrt{D}}{2a}, \quad x_2 = \frac{-b-\sqrt{D}}{2a}.$$

6. Разработайте алгоритм и программу решения следующей задачи. Определить стоимость железнодорожного билета «туда и обратно», если известны расстояние до пункта назначения и длительность пребывания в нем, учитывая, что если расстояние превышает 1000 км, а длительность пребывания превышает 7 дней, то железнодорожная компания дает скидку 30 %.

7. Разработайте алгоритм и программу решения следующей задачи. Даны три числа a, b, c. Проверить, образуют ли они строго возрастающую (a<b<c), строго убывающую (a>b>c) последовательность или не выполняется ни одно из этих двух условий.

8. Разработайте алгоритм и программу решения следующей задачи. Найти минимальное из трех чисел.

9. Разработайте алгоритм и программу решения следующей задачи. Вычислить значение функции **y**:

$$y = \begin{cases} 0, & \text{если } x < 0 \\ x, & \text{если } 0 \le x \le 1 \\ 1, & \text{если } x > 1 \end{cases}$$

10. Разработайте алгоритм и программу решения следующей задачи. Пользователь вводит два числа. Меньшее из введенных чисел заменяется числом 0, а в случае их равенства - числом 100.