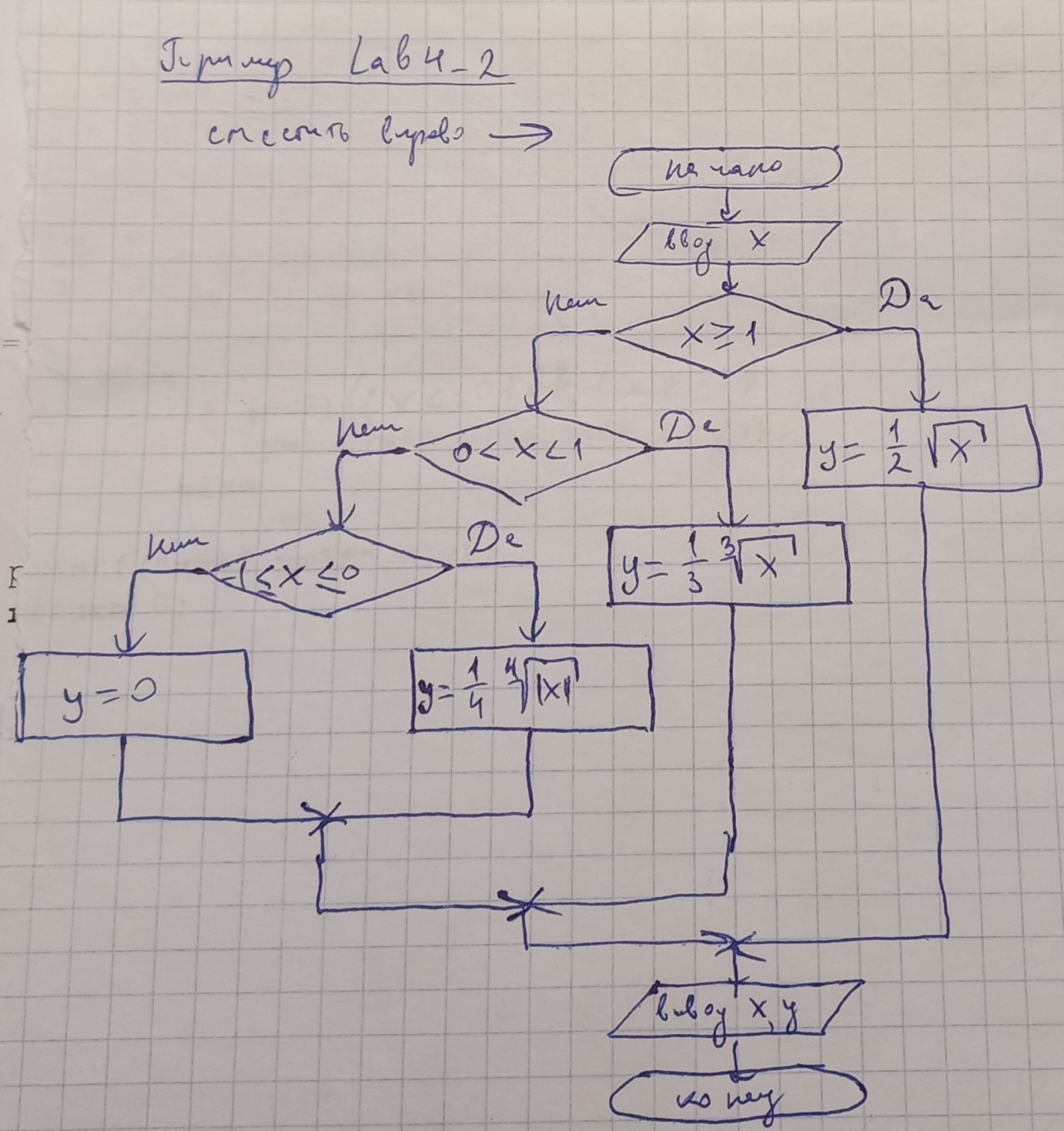
**Начать со схемы к примеру 4\_2 (программа была рассмотрена в конце предыдущей лекции)**

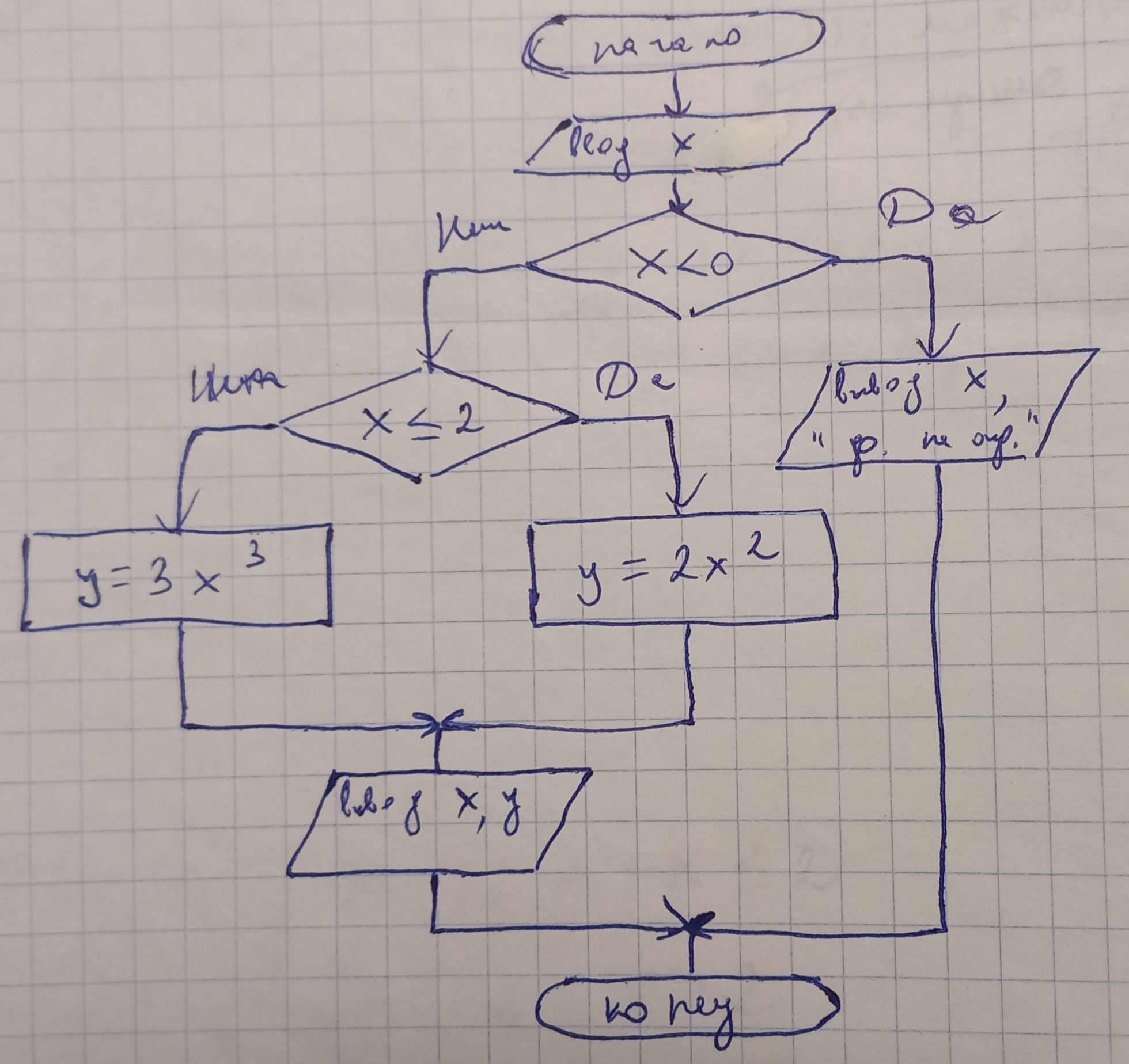
Обратить внимание и показать, что каждый условный оператор закрывается отдельно, начиная с самого вложенного. Частая ошибка, когда на схеме все условные операторы закрыты в один (в одну линию).

****

**Пример 4\_3**

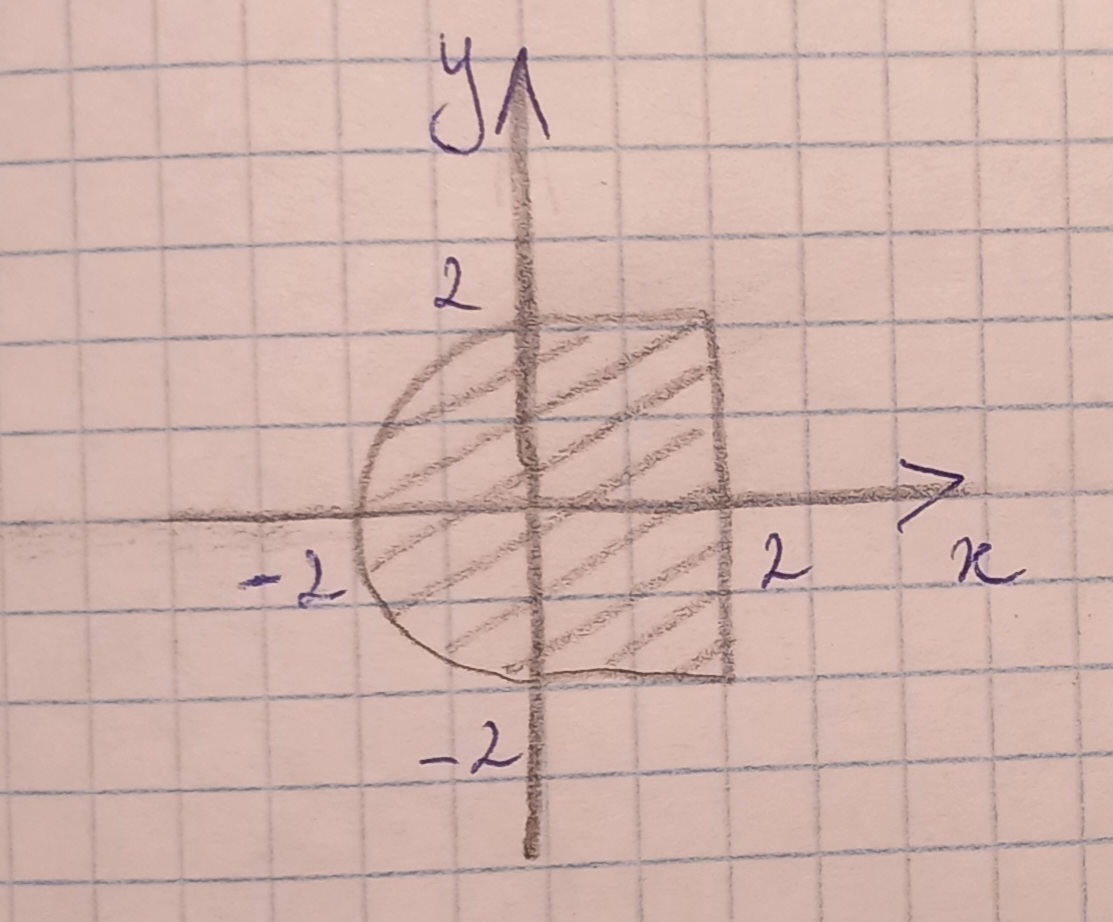
в остальных случаях функция не определена

Объяснить, что когда функция не существует, то не должно быть такого, что в одной из веток мы написали, что функция не существует, а потом пытаемся в конце программы печатать какой-то несуществующий y. Правильно в такой программе сразу проверить диапазон, на котором функция не существует, в ветке да вывести x и фразу «функция не определена», а во второй ветке вычислить y по оставшимся формулам и уже в конце этой ветки вывести x=… y=…



**Пример 4\_4**

Попадание точки в область



Показать как получена область (особенно про уравнение окружности, если равенство – то это точки на окружности, а если знак ≤, то это будут точки внутри окружности тоже, y выражать не нужно). Показать, что две области – левая ❶ и правая ❷. Записать каждую область в отдельности, потом спросить, какая логическая операция между ними (объяснить, почему правильно ИЛИ)

❶x<0 и x2+y2≤R2

или

❷x≥0 и x≤2 и y≤2 и y≥-2

**Тема 5 Многоальтернативное ветвление**

Ранее было рассмотрено, что проверка некоторого количества условий по очереди (обособленно) в большинстве случаев является неэффективным и неудобным решением. Для решения проблем, связанных со сложным выбором в зависимости от нескольких условий, используют вложенное ветвление (см. примеры 4\_2 и 4\_3).

В Питоне if, вложенный в ветку else другого if, можно заменить на конструкцию, называемую elif.

Синтаксис имеет вид:

if <выражение 1>:

<набор команд 1>

elif <выражение 2>:

<набор команд 2>

...

elif <выражение N>:  
 <набор команд N>

else:

<набор команд N+1>

**Пример 5\_1**

Разработать программу, которая выводит на экран дисплея соотношение между старыми русскими мерами длины и современными.

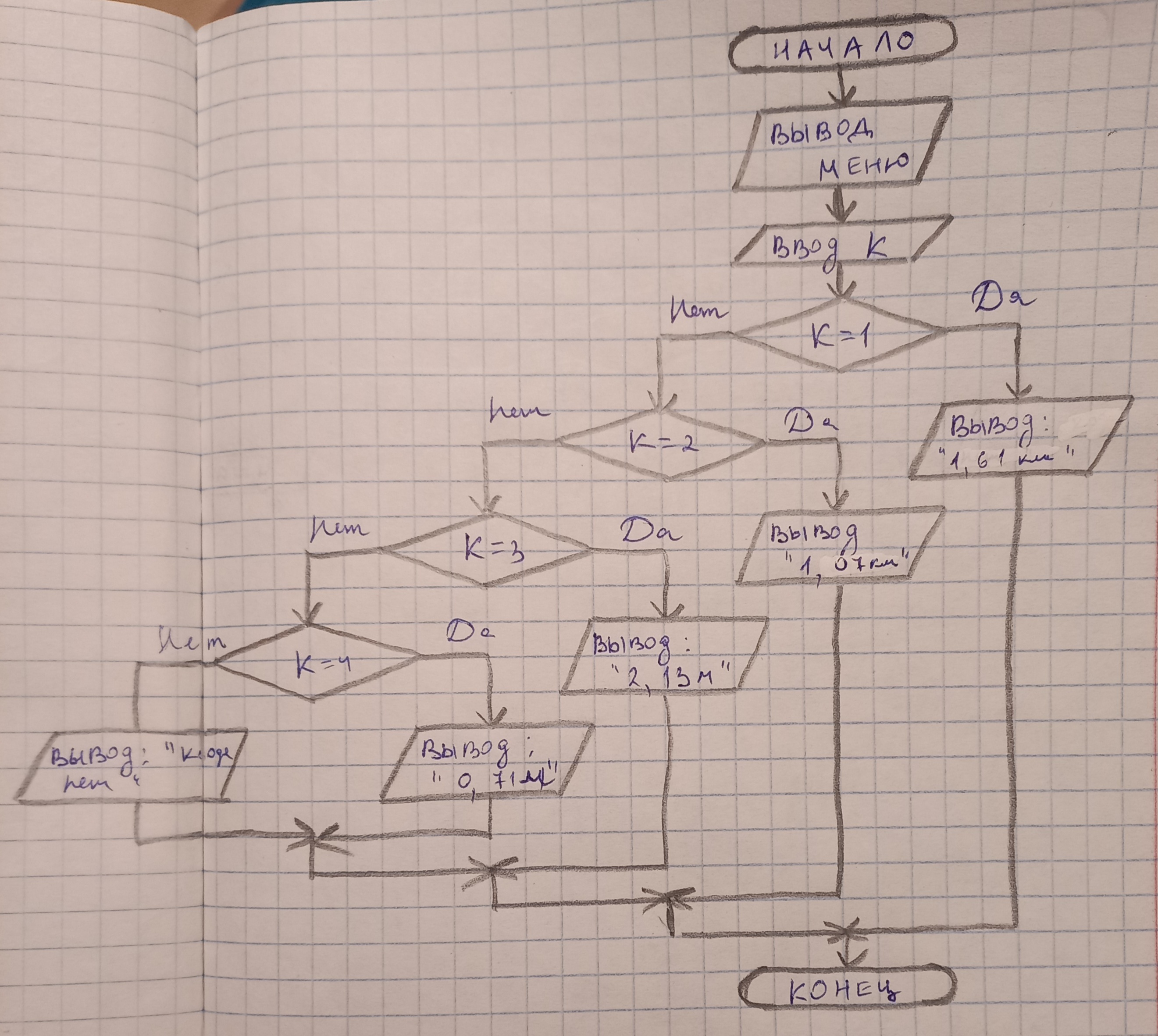
Вид простейшего меню:

1 - миля

2 - верста

3 - сажень

4 – аршин



**Оператор выбора**

Осенью 2021 года в Питоне версии 3.10 появился аналог case в Паскале и switch в С, оператор называется match.

Общий синтаксис конструкции match-case:

**match** <выражение>:

**case** <шаблон 1>:

<команды 1>

**case** <шаблон \_2>:

<команды 2>

...

**case** <шаблон N>:

<команды N>

**case** **\_**:

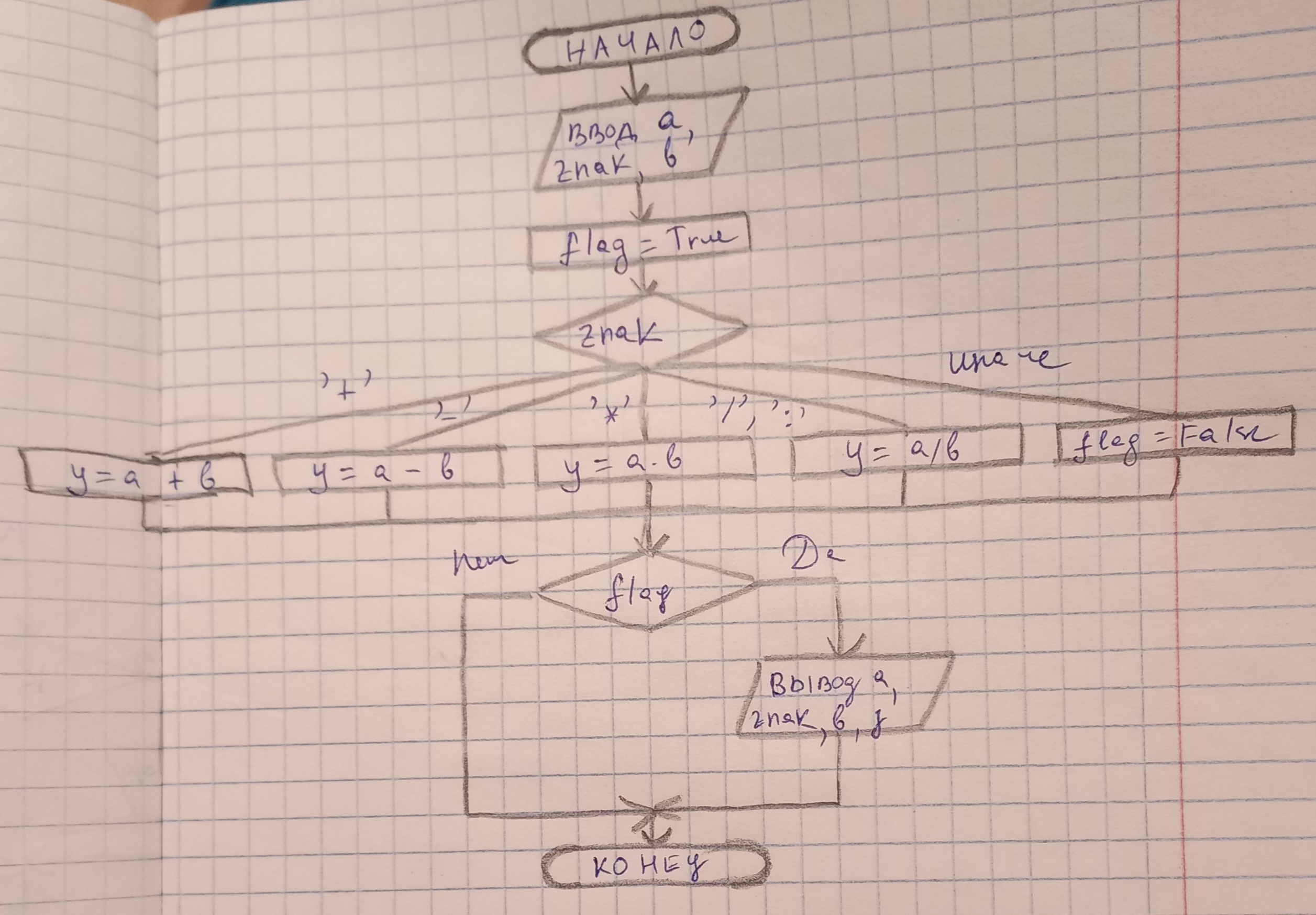
<команды N+1>

Оператор match принимает выражение  и сравнивает его значение с последовательными шаблонами, заданными как один или несколько блоков case. Тип значения для шаблона должен совпадать с типом значения выражения. Если значение шаблона совпало со значением выражения, выполняется блок команд, вложенных в секцию case данного шаблона.

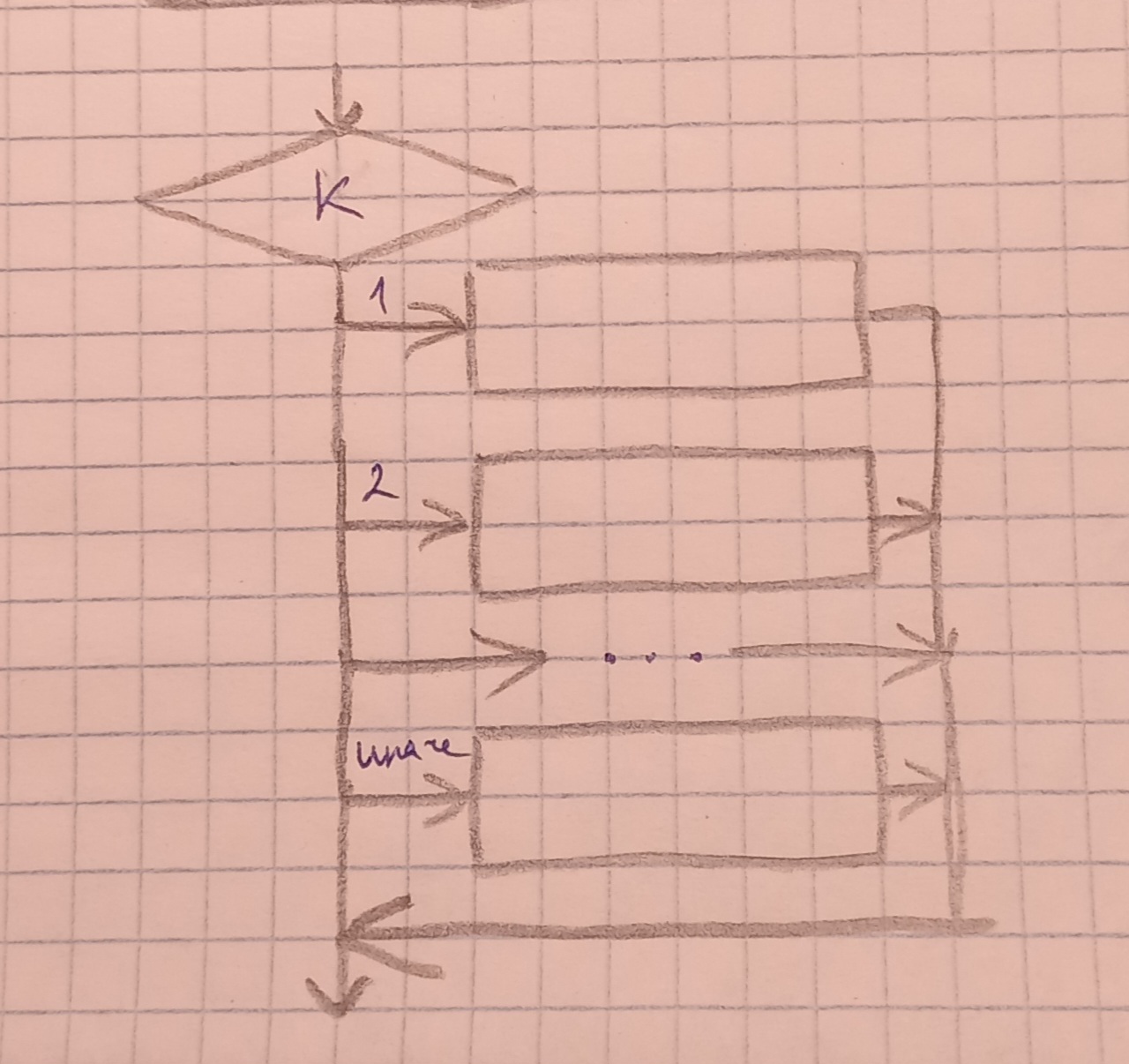
Если точного совпадения не произошло, то будет использоваться последний case c знаком '\_', если он указан. Если точное совпадение не подтверждено и case \_: - не существует, то весь блок match не выполняется.

**Пример 5\_2**

Простейший калькулятор



На схеме алгоритма оператор варианта можно еще изображать вот так.



Первый вариант экономит место в высоту, второй – в ширину.