План решения задачи
«Генерация программного кода по блок-схеме алгоритма»

Вначале опишем, какие функции необходимо реализовать в разрабатываемой модели для решения задачи

1. Распознавание составных элементов блок-схемы
   - Замкнутых геометрических фигур (прямоугольник, пятиугольник, шестиугольник, овал, он же четырехугольник с закругленными краями)
   - Незамкнутых геометрических фигур (отрезок прямой линии, стрелка)
   - Письменных знаков, или элементов текста (буквы, цифры, пунктуационные знаки, знаки математических и логических операций)

2. Определение взаимного расположения элементов и трактование их смысла в зависимости от позиции, характеризуемой координатам
   - Замкнутые геометрические фигуры (овал вверху схемы, «выше» всех остальных элементов – начало алгоритма, овал внизу схемы – конец алгоритма; прямоугольник справа от верхнего овала – список входных переменных)
   - Текст (внутри замкнутой фигуры – команда; вне фигуры – подтверждение Да, Yes, либо отрицание выполнимости условия Нет, No)
   - Незамкнутые геометрические фигуры (наличие ветвлений, по сути наличие общей точки для трех отрезков, что соотвествует началу/концу выполнения блока кода для условной конструкции, может понадобиться координата этой общей точки)

3. Классификация элементов блок-схемы, как элементы схемы соотносятся со структурными элементами алгоритма
   - Геометрические фигуры (овал, или прямоугольник с закругленными краями – начало/конец алгоритма; шестиугольник – условие цикла while или условие if; пятиугольник – начало/конец цикла for; прямоугольник – любая команда, как правило вычислительная, либо вывод на экран)
   - Смысл текста внутри одной фигуры (в шестиугольниках – вопросительный знак, указывающий на наличие условия, которое может быть выполнено или не выполнено, и конкретная операция сравнения, на которую указывает соотвествующий знак; в прямоугольниках и пятиугольниках содержится либо конкретная арифметическая операция, операция присваивания, либо список переменных или название функции)

4. Определение уровня вложенности конструкций, связей между объектами
    - Начало/конец алгоритма
    - Проверка на наличие более «крупных» циклов и ветвлений, содержащих вложенные циклы и ветвления (до тех пор, пока набор таких конструкций внутри алгоритма не будет исчерпан)
    - Отдельные циклы и ветвления без вложенных условий, содержащие только команды и утверждения, не являющиеся циклами и ветвлениями
    - Последовательность расположения объектов (по схеме сверху вниз и относительно друг друга)

5. Построение программы, перевод распознанных элементов, их связей, их порядка следования и команд алгоритма на язык программирования Python
    - Соотнесение элементов схемы с ключевыми словами и основными конструкциями языка программирования (название исполняющей алгоритм функции, имена и значения входных параметров, циклы for или while, оператор условия if, функция печати print, логические операции and или or, функция вывода на экран print)
    - Расположение конструкций в нужном порядке с учетом их уровня вложенности

Для обучения модели могут быть использованы датасеты из открытых источников. В первую очередь набор изображений с необходимыми геометрическими фигурами. При необходимости данные можно сгенерировать и нарисовать доступными средствами рисования Matplotlib. Для распознавания текста можно использовать предобученные модели Detecron2. Дополнительно можно использовать набор данных из открытых источников для обучения модели распознаванию отрезков.

Датасет для тестирования модели представляет собой набор изображений с блок-схемами алгоритмов. Схемы могут быть разного качества, они не обязательно состоят из ровные и четких фигур и текстов. Изображения могут содержать потертости, перспективные искажения, потертости, особенности фона в фроме клетки, линейки и так далее. Схемы могут быть как печатные, так и нарисованные от руки. Дополнительные данные могут быть сгенерированы при помощи библиотек OpenCV и пакета Skimage, например, с использованием аффинных преобразований с различными параметрами, размытия и так далее.
Посмотреть датасет можно по ссылке:
https://drive.google.com/drive/folders/1P9_cTfhcmz4IA0D2IqUWvfJURAXG7IUZ?usp=sharing

Построение модели можно разделить на два больших этапа, опишем их подробнее:

1. Обучение модели распознаванию ключевых элементов схемы (геометрических фигур, в том числе отрезков, текста). Для этой цели подойдет Detectron2 – библиотека модульных моделей компьютерного зрения на основе фреймворка PyTorch, предлагается использовать модель Faster R-CNN для обработки изображений. Вначале можно идентифицировать и классифицировать замкнутые фигуры и распознать текст, также определить координаты фигур и текста (например, угловые точки баундбоксов). Затем для более корректного распознавания отрезков и стрелок может потребоваться сегментация – «вырезка» замкнутых гелметрических фигур из всего изображения и рассмотрения оставшихся отрезков. Ветвление представляет собой соединение трех отрезков, и может оказаться полезным знать координату точки их пересечения с целью идентификации начала/конца ветвления и выявления элементов (команд), которые входят в данное ветвление. По умолчанию, если ветвление с простыми отрезками/ломаными – значит, перед нами цикл if, если один из концов отрезка является стрелкой – то цикл while. Вся получаемая информация об элементах схемы и их связях записывается в формате json или csv.

2. Построение кода программы. Программный код можно представить как ориентированный граф, в котором могут содержаться ориентированные петли. Узлами такого графа являются замкнутые геометрические фигуры, то есть строковые команды. Ребра по умолчанию соединяют узлы последовательно друг за другом и направлены сверху вниз (если смотреть на схему). В случае ветвлений направление может уточняться, для этого определяется координаты точек пересечения трех отрезков и положение команд относительно них (внутри/вне ветвления). Также на основе собранной информации об элементах можно определить уровень вложенности команд, в зависимости от этого в программный код добавляется нужное количество отступов (по правилам языка Python).

Точность модели будет определяться несколькими критериями. Во-первых, это точность распознавания геометрических фигур и текста на изображении. Во-вторых, это количество правильно интерпретированных блок-схем алгоритмов.