Начну с изменений в FormulaAST: тут появился новый класс CellExpr : Expr для него нужно реализовать Evaluate() - то есть имея имя ячейки нужно понять что в ней хранится. Я планирую опираться на то, что таблица находится в согласованном состоянии на момент вызова и буду вызывать для переданной таблицы(константная ссылка на std::unordered_map<int, std::unordered_map<int, unique_ptr<Cell>>> передаётся в метод Evaluate()) метод GetCell() - так как таблица согласована, там будет лежать либо string, либо double, а если GetCell() вернёт nullptr, значит ячейка не существует, для такой ячейки значение по условию задачи - 0. Если GetCell() вернёт string, то попытаемся его интерпретировать как double методом std::stod(возможно такое не сработает, так как метод желает получить чётко отформатированную строку, в таком случае либо поищу другие варианты, либо напишу свой конвертер string в double), если сконвертировать в double не удастся, то выкину исключение.

Решение проблемы с циклами в ячейках: воспользуюсь проходом в глубину по используя при этом переданный std::unordered_map<int, std::unordered_map<int, unique_ptr<Cell>>> и новый std::forward_list<Position> cells_. Чёткой идеи, как реализовать поиск в глубину пока в голове нет, так как не делал никогда подобного, но по ходу написания кода думаю разберусь. Проверять на цикличность буду в самом начале метода CellExpr::Evaluate(), в случае нахождения цикличности буду бросать соответсвующее исключение, в случае, если были невалидные ячейки - буду бросать соответсвующее исключение.

Реализация кэша: внутри Cell добавлю ещё два приватных поля, которые будут хранить “предков” и “потомков” конкретной ячейки vector<std::unique_ptr<Cell>*>, пример реализации: есть ячейки:
A1 = “=A2”

A2 = “=A3 + A4”

A3 = “=5”

A4 = “=3”

Идея такая: в момент A2 = A3 + A4 я в A3 и A4 добавлю, что у них появился предок A2, а в A2 что у него есть потомки A3 и A4; A1 = A2 здесь у A2 появляется предок A1, а у A1 потомок A2. Допустим теперь я захочу в A2 записать “=3”. В этом случае я сначала пройдусь по потомкам A2 и так как теперь A2 не зависит ни от A3 ни от A4 удалю в A3 и A4 упоминание о том, что от них зависит A2. После этого посмотрю на предков A2, в данном

случае A1 - и пересчитаю его. Далее посмотрю на предков A1, если таковые есть, пересчитаю их, и тд. Кэш как таковой в моём понимании уже есть, ведь в момент добавления новой ячейки или изменения существующей, все остальные ячейки корректны и их значения посчитаны, то есть я не буду добавлять какую либо структуру данных, для хранения “кэша”. Кэш = значения ячеек до того, как было вызвано добавление новой/редактирование существующей ячейки, хотя может такое интерпретирование не верно и решение не будет работать, когда начну реализовывать будет яснее.