1) Компиляция, запуск, структура проекта

- Минимальная структура программы на C++: где main(), какие заголовки нужны для ввода/вывода/математики?
- Почему важны предупреждения компилятора (-Wall -Wextra)?

2) Ввод/вывод и форматирование

- Как прочитать «число оператор число», если оператор — символ? Почему >> пропускает пробелы/переводы строки?
- Чем отличаются std::fixed и std::setprecision(3) и когда их использовать?
- Как печатать «дружелюбные» сообщения об ошибках и зачем они нужны?
- Почему «лишний пробел/перенос» может «уронить» автопроверку?

3) Типы, преобразования и переполнения

- Что такое переполнение целых и как его «увидеть» на примере периметра/площади?
- Почему замена только переменных результата на long long не спасает, если a и b остались int?
- Как «поднять» всё выражение в широкий тип (cast/литерал 1LL) и где это делать?
- Где проходит граница «безопасных» сумм относительно INT_MAX?
- Чем double удобен/опасен для калькулятора?

4) Ветвления и обработка ошибок

- Как различать «неизвестная операция» и «деление на ноль» в калькуляторе?
- Что считать «ошибкой ввода» и как на неё реагировать?

5) Функции и разделение ответственности

- Почему «чистые» функции не должны делать cin/cout?
- Как оформить try_div(a,b,q) так, чтобы было понятно, успешен ли расчёт?
- Какое имя выбрать функции, чтобы из названия было ясно её назначение?

6) Численные функции и тригонометрия

- Почему sin, cos, sqrt работают в радианах и как конвертировать градусы↔радианы? (deg↔rad)
- Что выведет гипотенуза для (3,4) и почему это «обязательный» тест?
- Как корректно сравнивать два double (эпсилон-сравнение) и откуда взять допуск?

7) Тестирование и самопроверка

- Какие кейсы считать «нормальными», «пограничными», «ошибочными»?
- Чем полезны «противопримеры», найденные тестами?

8) Приоритеты и порядок вычислений

- Где чаще всего ошибаются с приоритетами (*, /, %, +, -)?
- Почему выражение a + b * c не равно (a + b) * c и как скобками гасить двусмысленность?

11) Стиль кода и оформление

- Какие правила именования переменных/функций возьмём за стандарт курса?
- Как писать «говорящие» сообщения об ошибках (калькулятор)?
- Когда уместны комментарии, а когда лучше улучшить код/имена?

12) Мини-алгоритмическое мышление (на базе Ч.1)

- Чем линейный алгоритм отличается от ветвления и цикла на простых задачах из автопроверки?
- Как оценивать «достаточно ли быстро» на бытовом уровне без формальной сложности?

13) Частые грабли новичка

- «Всё работает, но автопроверка ругается»: что проверить в первую очередь (формат, пробелы, перевод строки)?
- «На моих тестах хорошо, на чужих плохо»: как расширить тест-набор?
- «Падает на больших числах»: вспомним про переполнения и типы.