

Задание 1 для первого семестра 2 курса

Требуется разработать и реализовать несколько простейших классов на C++ и написать тесты для проверки работы этих классов.

Требования к реализации

0. Если кто еще не умеет, то должен научиться писать реализацию в нескольких файлах, использовать собственные заголовочные файлы, уметь составлять простейшие make-файлы и использовать их для сборки, а также уметь собирать и отлаживать программу из командной строки (в том числе с использованием gdb).

1. Все реализации должны удовлетворять согласованным неформальным интерфейсам и иметь заголовочный .h файл с прокомментированным описанием методов класса. Разработка формального описания классов предоставляется студенту, при этом должны быть предусмотрены способы реакции и оповещения о некорректных или ошибочных ситуациях.

2. К заданию должны быть написаны тесты, т.е. отдельные программы которые вызывают требуемые функции с разнообразными наборами данных и (если возможно, то автоматически) проверяют правильность результата.

3. Поскольку детальная реализация предлагаемых задач по векторам и геометрии может оказаться весьма объемной, то не нужно пытаться реализовать все возможные операции. Но при этом реализованная часть все же должна поддерживать некоторые вполне содержательные задачи (тесты) и должно быть ясно как при необходимости можно расширить реализацию до необходимого уровня доопределением дополнительных функций и операций.

Возможные варианты задания Данные формулировки задают только направление работы. Конкретизация постановок, уточнение и т.д. проводятся в рабочем порядке с преподавателем. Каждый должен выполнить некую реализацию в рамках задачи **АII**. и одно из заданий **А–Н**. Задачи стоит делать одновременно, чтобы по мере знакомства с конструкциями C++ постепенно расширять свои “проекты”.

АII. Евклидова геометрия в \mathbf{R}^2 . Векторы, точки, отрезки, линии, фигуры. Построения “циркулем и линейкой”.

Задание подразумевает создание системы “геометрических” классов, в которых координаты объектов спрятаны в private секциях, а на уровне интерфейса предоставляются естественные геометрические соотношения и операции по типу геометрии Евклида.

Построения “циркулем и линейкой”. Сравнения объектов, пересечения, принадлежность. Преобразования (подобие, гомотетия, повороты, симметрии, инверсии).

Предполагается, что каждый студент выберет себе некоторое количество “школьных” задач на построение или проверку теорем, реализует необходимые классы геометрических объектов и напишет тесты для решения поставленных задач.

Кроме этого каждый реализованный геометрический класс должен иметь метод, позволяющий его отрисовать, например, в смысле использования утилиты gnuplot. В этом случае, можно будет визуализировать последовательность и результат построений “циркулем и линейкой” требуемых геометрических объектов.

Например, пусть поставлена задача построения треугольника по трем сторонам. Т.е. заданы три объекта типа “отрезок”. При построении “циркулем и линейкой” у нас нет возможности узнать длину отрезков, мы можем только откладывать отрезок, равный данному. Поэтому можно предусмотреть конструктор объекта “окружность” с радиусом, равным данному отрезку. После этого мы создаем две окружности с радиусами равными двум данным отрезкам и центрами на концах третьего отрезка, получаем точки пересечения этих окружностей как применение отдельной операции пересечения окружностей, и создаем объект “треугольник” из найденных трех точек. Еще раз обращаю внимание, что циркуль и линейка не предполагают использование числовых значений длины, только манипуляции с отрезками, прямыми, точками, окружностями и т.д. В качестве задач можно взять:

- построение треугольника по трем элементам (не только простейшим, но и более сложным, скажем по двум углам и отрезку с длиной равной периметру и т.д.);
- построение касательных с разным фигурам;
- построение преобразованных фигур (растяжения, повороты и пр.)
- “проверка” разных утверждений (например, вписанный угол, опирающийся на диаметр прямой, реализация сравнения фигур на равенство как это понимается у Евклида, т.е. все сводится к проверке совпадению точек или принадлежности точек прямым или отрезкам);
- любые другие аналогичные задачи.

Не возбраняется коллективное использование геометрических классов, но не программ решающих конкретные задачи на построение. Эти программы каждый должен выбрать для себя сам и уметь также потом выполнять аналогичные построения сразу на семинаре.

Пример: Проверить, что вписанный угол, опирающийся на диаметр, является прямым. Решение: Строим произвольную окружность. Проводим прямую через ее центр. Находим отрезок пересечения этой прямой с окружностью — это диаметр. Проводим прямую через одну из крайних точек диаметра — получаем хорду. Строим отрезок другой хорды. Получаем вписанный угол. Строим прямой угол известным школьным построением. Сравниваем эти два угла (не используя понятие градусной меры). В этом смысле в школьной геометрии под равенством фигур понимается их совпадение при наложении. В нашем случае мы можем определить "равенство точек" (как совпадение их координат в рамках заданной точности), далее определить равенство отрезков как равенство их длин в рамках заданной точности (при этом понятие длины присутствует только внутри нашей процедуры сравнения, на внешнем уровне длина как число не появляется)

Арифметика. Задача — построить класс, представляющий собой специальное число, и потом проверить некоторые вычислительные алгоритмы с такими числами. Для начала — простейшие арифметические операции, потом что-то более сложное — суммирование рядов, вычисление интегралов, решение уравнений и т.п. Функция печати этого числа в соответствующей наглядной форме.

Задачи этого типа могут оказаться разной сложности. В случае сложных задач надо вовремя остановиться. То есть, построить реализацию, которая может решать некоторые содержательные частные задачи, но не обязательно обладает полной функциональностью. В списке ниже более сложные задачи отмечены звездочкой.

А. Комплексная арифметика. Обычное комплексное число — действительная и мнимая части. Реализация операций, элементарных функций. Тест, например, решение квадратного уравнения, вычисление элементарных функций через ряды и т.п.

В. Рациональная арифметика. Число представляется как целочисленная пара — числитель и знаменатель. Арифметические операции, сокращение общего множителя, контроль переполнения знаменателя. Взаимодействие с вещественными числами (приближение).

***С.** Длинная целая арифметика. Число представляется массивом его цифр (можно в десятичном или двоичном виде, как удобнее и понятнее). Четыре арифметических операции (деление "столбиком"). Тест, например, вычисление факториала, биномиальных коэффициентов, чисел Фибоначчи или каких других аналогичных значений. В перспективе интересно объединение с задачей **В.** для устранения переполнения знаменателя.

***D1.** Вещественная арифметика с фиксированной точкой. Число — это пара целых чисел (целая и дробная части). Арифметические операции и элементарные функции. Тесты — любые вычислительные процедуры и сравнение с результатами в `double` или `float`.

***D2.** Вещественная арифметика с фиксированной точкой. Число — это целое число, в котором часть разрядов отведена под целую часть, а часть под дробную.

Вариант: в разных числах это разделение может быть разным (т.е. это тоже переменная класса), в этом случае при операциях необходимо делать выравнивание.

Вариант: это разделение определяется в конструкторе и числа с разным разделением не могут участвовать в одной операции.

Арифметические операции и элементарные функции. Тесты — любые вычислительные процедуры и сравнение с результатами в `double` или `float`.

Е. Интервальная арифметика. Число — это интервал, которому может принадлежать рабочее значение (`min`, `max`). Арифметические операции и элементарные функции. Соответственно результат любой операции — это тоже интервал для максимального и минимального возможных значений. Тесты — любые вычислительные процедуры и сравнение с результатами в `double` или `float`.

Ф. Арифметика с оценкой абсолютной погрешности. Число есть значение и модуль его абсолютной погрешности. Арифметические операции и элементарные функции. Тесты — любые вычислительные процедуры и сравнение с результатами в `double` или `float`.

Г. Арифметика с оценкой относительной погрешности. Число есть значение и его относительная погрешность. Арифметические операции и элементарные функции. Тесты — любые вычислительные процедуры и сравнение с результатами в `double` или `float`.

***Н.** Арифметика с производной. Число есть пара чисел, которые интерпретируются как значение некоторой функции и ее производной. Соответственно результат операций для производной вычисляется по формулам производной суммы, разности, произведения, частного. Выполняя некоторый алгоритм с такими числами (например, суммируя ряд для синуса), мы должны одновременно получить и значение производной (т.е. значение косинуса). Есть некоторые тонкости при взаимодействии с обычными числами.

I. Алгебра квадратных матриц и векторов фиксированной размерности. Размерность задается в конструкторе. Объекты разной размерности не могут участвовать в совместных операциях.

J. Алгебра матриц и векторов разной размерности. Т.е. доступны прямоугольные матрицы. Размерности задается в конструкторе. Объекты разной размерности могут участвовать в допустимых совместных операциях (умножение матриц и векторов с матрицами при согласованных соответствующих размерностях).

График выполнения задания

неделя 1: разработка и согласование описания (интерфейса) классов All и A-J.
(фактически — поставить себе задачу что делать)

неделя 2: реализация и первичный тест работоспособности.

неделя 3–4: доработка более сложных методов и тестов с реакцией на некорректные ситуации.

неделя 5: окончательная сдача и завершение задания.

При реализации нужно использовать перегрузку операторов, разнообразные конструкторы, в частности, операторы присваивания и сравнения и конструкторы копирования.

Следует предусмотреть возможность проверки и обработки ситуаций, когда операция с объектами не может быть выполнена или не дает корректного результата (например, деление на 0, пересечение параллельных линий и т.п.)