

ФБГОУ ВО АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ
УНИВЕРСИТЕТ

Физико-технический факультет
Кафедра вычислительной техники и электроники

Современные технологии программирования

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ВЫПОЛНЕНИЮ
ЛАБОРАТОРНЫХ РАБОТ

Барнаул 2021

Лабораторная работа 1

При написании ПО слишком часто возникает ситуация, когда хочется посмотреть, что же было некоторое время назад. Плюс, иногда хочется без слома основной разработки доработать разные части ПО по отдельности. Это позволит система Git.

Для примера предлагается разработать простую программу для выполнения математических преобразований над скалярными, векторными и тензорными величинами: скалярами, векторами и матрицами. Над матрицами может быть выполнено множество различных преобразований — от транспонирования и поэлементного сложения до перевода в скалярный вид при вычислении определителя. Вот несколько операций, которые могут быть выполнены по отношению к матрицам:

1. Умножение матрицы на скаляр $C[i][j] = bA[i][j]$;

2. Поэлементное сложение $C[i][j] = A[i][j] + B[i][j]$

3. Поэлементное произведение $C[i][j] = A[i][j]B[i][j]$

4. Умножение вектора на матрицу $C[j] = \sum_{l=0}^{L-1} A[l]B[l][j]$

5. Матричное произведение $C[i][j] = \sum_{l=0}^{L-1} A[i][l]B[l][j]$

6. Вычисление следа и определителя $TrM = \sum_{i=0}^{N-1} M[i][i]$,
<https://ru.wikipedia.org/wiki/Определитель>

7. Вычисление обратной матрицы <https://ru.wikipedia.org/wiki/O>

8. Транспонирование $B[i][j] = A[j][i]$

По отношению к векторам:

1. Умножение вектора на скаляр $C[i] = bA[i]$;

2. Поэлементное сложение $C[i] = A[i] + B[i]$
3. Поэлементное умножение $C[i] = A[i]B[i]$
4. Умножение вектора на матрицу $C[j] = \sum_{l=0}^{L-1} A[l]B[l][j]$
5. Скалярное произведение $= \sum_{l=0}^{L-1} A[l]B[l]$
6. Векторное произведение (только для трехмерных векторов) https://ru.wikipedia.org/wiki/Векторное_произведение
7. Вычисление длины вектора $\|A\| = \sqrt{\sum_{l=0}^{L-1} A[l]^2}$
8. Проверка сонаправленности векторов
9. Проверка векторов на ортогональность

По отношению к скалярам:

1. Сумма $= a + b$;
2. Инверсия $b = -a$
3. Произведение $c = ab$
4. Возведение в степень $c = a^b$
5. Вычисление корня $= \sqrt[b]{a}$
6. Расчет тригонометрических функций

UML (англ. Unified Modeling Language — унифицированный язык моделирования) — язык графического описания для объектного моделирования в области разработки программного обеспечения, моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

UML является языком широкого профиля, это — открытый стандарт, использующий графические обозначения для создания абстрактной модели системы, называемой UML-моделью. UML был создан для определения, визуализации, проектирования и документирования, в основном, программных систем. UML не является языком программирования, но на основании UML-моделей возможна генерация кода.

Требования к ПО Требуется реализация в объектном подходе. Выделяемые объекты — скаляр, вектор, матрица; поля объектов должны быть скрыты, все операции выполняются через вызов методов написанных классов. Для оценки полученных результатов должны быть методы, выдающие содержание полей объекта.

Интерфейс графический или команднотрочный. В интерфейсе должна быть возможность указания авторства ПО и полноценная помощь по пользованию программой. Интерфейс должен позволять в процессе выполнения программы задать последовательность выполнения действий, как то: задать величины, рассчитать значения, выдать значения в интерфейс (или в текстовые файлы, или в табличные файлы, на выбор). По сути, пишем короткий аналог Matlab'a.

Во время разработки должен использоваться Git. Работы без указания истории отметок в Git не принимаются.

Описание Git Для начала работы следует установить пакет Git. Затем в директории проекта инициализировать репозиторий Git. Добавить файлы проекта в хранилище, содержащие исходный код и сделать первую запись редакции в хранилище. Создавать боковые ветви разработки не требуется, но и не запрещается. Финальная версия проекта должна находиться на главной ветви. Создавать удаленный репозиторий Git не требуется. Инициализация делается командой `git init`, добавление — `git add`, запись редакции в хранилище — `git commit -a`, создание и переключение ветвей — `git branch`, описание их смотреть

в документации к Git: <https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control>.

Задание

- Изучить основы управления версиями в пакете Git. Для начала работы советуется использовать документацию на официальном сайте проекта. Присутствуют версии на разных языках.
 1. После создания проекта (или пустых файлов исходного кода программы) сделать инициализацию репозитория Git в директории проекта.
 2. Сделать начальную отметку, содержащую данные о создателе и дате создания.
- Написать указанный программный продукт с использованием Git в процессе разработки.
- Загрузить отчет по работе и сопутствующие файлы к 2021-03-02.

Отчет должен содержать:

- Титульный лист с указанием автора, проверяющего преподавателя, названия работы, даты сдачи, ...
- Содержание
- Краткую теорию: описание терминов «Скаляр», «Вектор», «Матрица» с указанием источника, описание языка реализации матричного калькулятора, описание каждой матричной операции
- UML-диаграмму, доказывающую реализацию калькулятора в указанном подходе
- Историю отметок Git