

Федеральное агентство по образованию
Государственное образовательное учреждение высшего
профессионального образования

**ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)**

Кафедра компьютерных систем в управлении и проектировании (КСУП)

М. А. Песков,

**Объектно-ориентированное программирование.
Методические указания к выполнению курсовых работ**

Учебно-методическое пособие

Томск – 2007

Песков М.А.

Объектно-ориентированное программирование. Методические указания к выполнению курсовых работ : учеб. пособие / Под общей редакцией М.А. Пескова. – Томск : Томск. гос. ун-т систем упр. и радиоэлектроники, 2007. – 38 с.

© Песков М. А. 2007

© Том. гос. ун-т систем упр. и
радиоэлектроники, 2007

Содержание

Введение	5
1 Задача курсовой работы.....	6
2 Содержание пояснительной записки.....	7
3 Темы на курсовой проект	9
3.1 Игра “Нажми Клавишу”	9
3.2 Игра “Поймай мышкой”	9
3.3 Тест по национальным сказкам	9
3.4 Практикум по национальным сказкам.....	10
3.5 Пианино	10
3.6 Часы.....	10
3.7 Моделирование игры “ЖИЗНЬ”	10
3.8 Моделирование “болезни”	11
3.9 Генератор любовного романа	11
3.10 Моделирование работы лифтов.....	11
3.11 Графическое программируемое табло	12
3.12 Интерактивная карта	12
3.13 Игра “Морской бой”	12
3.14 Логическая игра «Реверси».....	12
3.15 Экранный калькулятор.....	13
3.16 Вычислитель арифметических выражений.....	13
3.17 Музыкальный редактор.....	13
3.18 Карта Саха-Якутии	13
3.19 Тестирующая система	14
3.20 Логическая игра “Пятнашки”	14
3.21 База данных “классный журнал”.....	14
3.22 База данных “поручения”.....	15
3.23 База данных “Дни рождения”	15
3.24 Персонаж	15

3.25	Логическая игра “Шарики”	15
3.26	Логическая игра “Сокобан”	16
3.27	Логическая игра “Тетрис”	16
3.28	Логическая игра “Рикошет”	17
3.29	Векторный графический редактор	17
3.30	Рисование круговых диаграмм	17
3.31	Построение графика	17
3.32	Построитель графиков функций	18
3.33	Электронный органайзер	18
3.34	Программируемый калькулятор	18
3.35	Эмулятор Total Commander	18
3.36	Говорилка по тексту	18
3.37	Просмотрщик трёхмерных моделей	19
3.38	Игра монополия	19
3.39	Логическая игра “Крестики-нолики” на неограниченном поле	19
3.40	Логическая игра “Цепь”	20
3.41	Логическая игра “Точки”	20
3.42	Логическая игра “Обратный тетрис”	21
3.43	Логическая игра “Шахматы”	21
4	Список литературы	22
	Приложение А. Пример пояснительной записки	23

Введение

Данное пособие является набором указаний для выполнения курсовой работы по курсу «Объектно-ориентированное программирование». Целью курсового проектирования является закрепление навыков программирования на языке C++ на практике, а также получение навыков разработки программных проектов.

1 **Задача курсовой работы**

Задача курсовой работы заключается в следующем:

- сформулировать техническое задание для проекта;
- произвести анализ сформулированного технического задания, спроектировать интерфейс пользователя, диаграммы взаимодействия модулей и классов;
- разработать систему тестов для будущей программной системы;
- на основании сформулированных требований и их анализа реализовать программу на языке C++;
- на основании разработанной системы тестов протестировать программу, добиться корректного прохождения всех тестов, зафиксировать результаты тестирования;
- по проделанной работе оформить пояснительную записку.

2 Содержание пояснительной записки

Пояснительная записка (ПЗ) к курсовому проекту (работе) должна включать в себя нижеследующие разделы.

1. *Титульный лист* – пример оформления титульного листа представлен в приложении А.

2. *Реферат* – содержит краткое описание (реферат) выполненной работы (2-3 предложения). Перечисляются ключевые слова, указывается количество страниц и приложений. Реферат размещают на отдельной странице. Заголовком служит слово "Реферат", написанное прописными буквами по центру страницы.

3. *Задание на проектирование* (техническое задание) – содержит постановку задачи для курсового проекта. В данном разделе необходимо привести требования к разрабатываемой системе, полностью описать функциональность будущей программы.

4. *Содержание ПЗ* – нумерованный по страницам список разделов ПЗ. Нумерация страниц ПЗ – сквозная: титульный лист имеет первый номер.

5. *Введение* – описание предметной области, актуальность решаемой задачи, обзор используемой литературы, обзор существующих аналогов.

5. *Анализ задания*. В данном разделе необходимо описать входные и выходные данные системы, спроектированный интерфейс пользователя, структуру хранимых данных, структуру программной системы, привести блок-схемы основных алгоритмов программы, диаграмму использования модулей, диаграмму взаимодействия классов программы.

6. *Реализация* – перечень модулей и классов реализованной программы с описанием их интерфейсных функций.

7. *Тестирование* – набор тестовых данных как для отдельных модулей программной системы, так и для всей системы в целом. Описание результатов тестирования.

8. *Заключение* – в данном разделе необходимо сформулировать основные итоги работы: сопоставление желаемых и полученных результатов, возможные пути развития проекта.

9. *Список литературы* - список источников, используемых при работе над проектом.

3 Темы на курсовой проект

3.1 Игра “Нажми Клавишу”

Случайно выводится изображение клавиши и необходимо как можно быстрее нажать соответствующую клавишу на клавиатуре. При этом программа измеряет время между выводом изображения и соответствующим нажатием. Перед началом игры вводится имя игрока и определяется количество попыток. Результаты игры вносятся в специальную таблицу

<имя игрока> <результат>

3.2 Игра “Поймай мышкой”

Правила игры: на экране случайно сверху-вниз или еще как ни будь начинает двигаться кружок. Цель игры поймать кружок с помощью курсора мышки. Скорость движения кружка сделать изменяющимся в зависимости от номера попытки (сначала медленно, а затем ускорять). Кроме этого необходимо измерять время между запуском и моментом попадания. Результаты заносить в специальную таблицу.

3.3 Тест по национальным сказкам

Взять небольшую сказку. По этой сказке сформулировать 10 вопросов. На каждый вопрос необходимо записать несколько вариантов ответов. Один из вариантов должен быть правильным, остальные неправильные. Каждый вопрос должен быть организован как меню. Если вариант ответа в меню был выбран правильно, то ответ оценивается 1, если вариант был выбран неверный, то ответ оценивается 0. Общая оценка ставится на в виде суммы ответов на каждый вопрос. Если на 9 и более вопросов было отвечено правильно то ставится оценка 5. Если на 6-8, то ставится оценка 4. Если правильно отвечено на 4-5 вопросов ставится оценка 3, в других случаях ставится оценка 2.

3.4 Практикум по национальным сказкам

Практикум очень похож на тест из задания 3.3. Отличие заключается в том что, если ответ на вопрос был неверный, то обучаемому представляется правильный ответ и далее вновь задается этот вопрос. Оценка в практикуме не ставится.

3.5 Пианино

“Пианино” - программа, которая по нажатию некоторой клавиши выдает звучание некоторой ноты и отображение этой нота на экране терминала. Следует учесть, что частота звучания ноты «Ля» первой октавы равна 480 Гц, частота звучания одной и той же ноты в соседних октавах различается в 2 раза, а соседних нот одной октавы – в корень из 12-ти раз.

3.6 Часы

Аналогово-цифровые с несколькими будильниками, секундомером, таймером обратного отсчёта

3.7 Моделирование игры “ЖИЗНЬ”

Игра “Жизнь” является одной из первых попыток создания клеточных автоматов. Правила игры следующие:

1. Весь экран разбивается на клетки одинаковых размеров (2 на 2 точки).
2. Клетка с белым цветом считается неживой
3. Клетка с другим цветом живая.
4. Задается начальная конфигурация “живых” клеток
5. Производится процесс жизни по следующим законам:
 - если рядом с пустой клеткой ровно три живых, то в пустой клетке рождается жизнь
 - если рядом с живой клеткой меньше двух живых клеток, то она умирает от одиночества
 - если рядом с живой клеткой больше трех живых клеток, то она умирает от тесноты.

3.8 Моделирование “болезни”

Болезнь моделируется с помощью клеточного автомата. Правила игры следующие:

1. Весь экран разбивается на клетки одинаковых размеров (2 на 2 точки).
2. Клетка с белым цветом считается здоровой
3. Клетка с другим цветом больной.
4. Больные клетки могут заражать здоровые соседние клетки с некоторой заданной вероятностью.
5. Клетки болеют несколько тактов, затем они приобретают иммунитет и несколько тактов не заражаются, затем они теряют иммунитет и снова могут заразиться.

Цель игры: установить начальную конфигурацию больных клеток, задать вероятность заражения соседних клеток и посмотреть процесс протекания заболевания на экране.

3.9 Генератор любовного романа

Сгенерировать роман по заданным пользователем параметрам: задаются герои, их внешний вид и имена, место встречи, и так далее. После запроса всех необходимых данных, формируется текст романа с иллюстрациями.

3.10 Моделирование работы лифтов

Требуется написать программу моделирующую работу некоторого количества лифтов в многоэтажном офисном здании (как вариант можно написать программу моделирующую работу эскалатора в супермаркете). Имеется несколько этажей, который обслуживаются лифтами, на каждом этаже есть дверь из которой появляются люди, которым требуется воспользоваться лифтом. Каждый человек едет на определённый этаж. В офисе также могут иметься лестницы. Лифты могут находиться на расстоянии друг от друга. Пользователь может варьировать количество лифтов, их скорость и скорость появления людей. Можно написать игру на эту тему.

3.11 Графическое программируемое табло

На вход программы подаются изображения заданные в некотором формате, которые затем выводятся в окне программы, которое разбито на ячейки определённого размера, каждая ячейка это один элемент табло, который может быть закрашен определённым цветом. Возможна реализация анимации (бегущая строка), и вывод различной информации (погода, время и т.д.).

3.12 Интерактивная карта

Реализовать на карте способ отображения названий избегая пересечений.

3.13 Игра “Морской бой”

Разрабатываемое приложение представляет собой программную реализацию известной логической игры.

Приложение должно обеспечивать:

- Расстановку кораблей на игровом поле 10x10
- Выбор противника (человек, компьютер).
- Изменение интерфейса в зависимости от выбора противника.
- Фиксацию имен противников и число побед.

3.14 Логическая игра «Реверси»

Разрабатываемое приложение представляет собой программную реализацию логической игры “Реверси”. Игра идет на поле произвольного размера. Два игрока по очереди устанавливают фишки своего цвета на поле. Фишку можно ставить только на те клетки, рядом с которыми уже стоят фишки. Если между установленной фишкой и какой либо другой фишкой того же цвета находятся фишки другого цвета, все они меняют свой цвет.

3.15 Экранный калькулятор

Экранный калькулятор - это программа, моделирующая работу обычного калькулятора. Реализовать простейший калькулятор, с шестью элементарными действиями (сложение, вычитание, умножение, деление, занесение в стек и удаление из стека). Этот калькулятор состоит из корпуса, экрана для отображения чисел и клавиш на которых нанесены цифры и арифметические действия.

3.16 Вычислитель арифметических выражений

Вычислитель позволяет произвести вычисления арифметических выражений. Арифметическое выражение записывается из рациональных чисел (констант), арифметических операций и скобок “(,)”. Например,

$$(100.6 - 3.54) / (2.01 + 4.78)$$

Необходимо использовать программы, использующие метод рекурсивного спуска.

3.17 Музыкальный редактор

На экране компьютера нарисована нотная линейка. Используя некоторый набор клавиш, можно записать простейшую мелодию. После записи этой мелодии ее можно неоднократно проигрывать, нажимая некоторую специальную клавишу “Играй” (Например клавиша F2). Частоты нот взять из задания 7.

3.18 Карта Саха-Якутии

Программа выводит на экран карту Якутии с указанием на ней всех улусов и наименования населенных пунктов. Далее используя мышку можно получить информацию о населенном пункте (численность населения, индустрия, промыслы и т.д.)

3.19 Тестирующая система

Задаются файлы входных данных с вопросами, справочной информацией, вариантами ответов и номерами правильных ответов. Пользователю выводятся вопросы и он выбирает ответы. Если ответ не правильный, предлагается справочная информация и задаётся тот же самый вопрос. Успешно ответив на вопрос пользователь продвигается к следующему. Для разных пользователей варианты ответов перемешиваются. По результатам теста выставляется оценка. Предусмотреть возможность шифрования входных данных.

3.20 Логическая игра “Пятнашки”

Разрабатываемое приложение представляет собой программную реализацию известной игры “Пятнашки”, цель которой заключается в упорядочивании игроком фишек путем их последовательного перемещения на соседнее пустое место. Приложение должно обеспечивать правильное перемешивание фишек.

Следует учесть, что при произвольной расстановке фишек на поле в 50% случаев становится невозможным их упорядочивание, поэтому перемешивание фишек должно осуществляться по тем же правилам, что и их перемещение игроком. Кроме того, после перемешивания фишек в начале игры не более 10% от их количества могут находиться на своих местах.

3.21 База данных “классный журнал”

Создается программа для ведения базы данных по учету успеваемости студентов. Эта программа должна выполнять следующие функции:

1. создает базу данных;
2. производит корректировку записей;
3. вывод базы данных на печать и экран.

Структура записи содержит:

- фамилию;
- имя;
- отчество

- год рождения
- оценка.

Не допускается использование стандартных СУБД (например, BDE).

3.22 База данных “поручения”

Необходимо построить программу аналогичную программе в задании

3.21. Структура записи следующая:

1. фамилия, имя, отчество;
2. поручение (что нужно сделать)
3. дата выдачи
4. дата исполнения.

Не допускается использование стандартных СУБД.

3.23 База данных “Дни рождения”

Необходимо построить программу аналогичную программе в задании

3.21. Структура записи следующая:

1. фамилия, имя, отчество;
2. дата рождения;
3. время рождения.

Не допускается использование стандартных СУБД.

3.24 Персонаж

Создать персонажа, аналогичного персонажу “Скрепыш” из пакета MS Office. Задать ему свойство постоянно находиться на экране компьютера. Обеспечить смену действий. Каждое действие описывается как несколько фаз, для каждой фазы имеется картинка, звуковой файл (?) и время выполнения данной фазы. Все действия и их фазы описаны во внешнем инициализационном файле. Персонаж реализовать в виде DLL.

3.25 Логическая игра “Шарики”

Разрабатываемое приложение представляет собой программную реализацию известной логической игры. Цель игры состоит в том чтобы

набрать максимальное количество очков. Суть игры состоит в следующем: на игровом поле отображаются круги разного цвета, игроку предлагается создать линию из кругов одинакового цвета, расположенных по горизонтали или по вертикали, созданная линия «сгорает». Причем линия может «сгореть», только в том случае если количество элементов в ней равно 3. После этой процедуры пустые места на игровом поле заполняются новыми элементами. Линию можно создать путем перестановки по горизонтали или по вертикали соседних элементов. Игра заканчивается тогда когда не возможно создать ни одной линии.

Приложение должно обеспечивать возможность задания количества цветов элементов (кругов).

Следует учесть что цвета кругов выбираются произвольным образом исходя из заданного количества. Кроме того, линии (3 и более элементов) получаемые при произвольной расстановки элементов «сгорают».

3.26 Логическая игра “Сокобан”

Играет один игрок. Игрок последовательно переходит от одного уровня к другому, по мере выполнения заданий. Уровень представляет собой лабиринт, в котором в беспорядке расставлены ящики. Игрок должен толкая ящики человечком расположить их на своих местах (места отмечены на лабиринте). Человечек может толкать только один ящик. Если ящик упирается в стену, то дальше его толкать нельзя.

3.27 Логическая игра “Тетрис”

Программа представляет собой тетрис на поле произвольного размера.

Програма должна предоставлять возможность выбирать размер фигур (4, 5, 6, 7 клеток). Скорость падения управляется автоматически, в зависимости от времени игры.

3.28 Логическая игра “Рикошет”

Игровое поле содержит источник лазера, набор целей, набор зеркал. Зеркала и цели выставляются на поле случайным образом в начале игры. Каждое зеркало может проворачиваться в любом направлении с шагом 30гр. Игрок должен поворачивая зеркала по очереди сжечь все цели.

3.29 Векторный графический редактор

Графический редактор – программа, предназначенная для создания векторных рисунков. Данный редактор представляет простейшую программу. Основные функции: рисование линий, прямоугольников (контурных и сплошных) с различным цветом, окружностей, эллипсов, дуг и так далее. Рисование линий осуществляются с помощью мышки. Обеспечить возможность чтения/ записи файлов в формате метафайлов или в собственном формате. Обязательно обеспечить возможность редактирования нарисованных объектов.

3.30 Рисование круговых диаграмм

Написать программу, которая по заданным ей данным рисует круговую диаграмму. Данные хранить в файле или обеспечить взаимодействие через DDE или OLE. Программу оформить в виде динамической библиотеки.

3.31 Построение графика

Аналогично предыдущему заданию, только выводится обычный линейный график. Предусмотреть возможность отображения логарифмических шкал, настройка из файла и вручную, автоматическое и ручное масштабирование.

3.32 Построитель графиков функций

Программа читает из файла ф-ию заданную аналитически или в виде списка значений и строит её график в определённом диапазоне. Дополнительные возможности: скользящий маркер (позволяет узнать значение функции в любой указанной точке), несколько видов аппроксимации, настройки цвета и типа линий, несколько функций на одном графике.

3.33 Электронный органайзер

Телефонная книжка, планирование событий и т.д.

3.34 Программируемый калькулятор

Реализовать экранный калькулятор, работающий в 3-х режимах: режим калькулятора, режим ввода программы, режим выполнения программы. В калькуляторе реализовать шесть элементарных действий (сложение, вычитание, умножение, деление, занесение в стек и удаление из стека).

3.35 Эмулятор Total Commander

Реализовать эмуляцию перемещения по некоторой файловой системе. Реализовать эмуляцию большинства функциональных клавиш Total Commander, включая сочетание с клавишами «Ctrl», «Alt» и «Shift».

3.36 Говорилка по тексту

Программа проговаривает голосом заданный текст. Программа реализуется в виде DLL. Информация берется из файла или передается посредством DDE. Дополнительно написать маленькую программу демонстрирующую работу DLL, где текст заносится в текстовый редактор.

3.37 Просмотрщик трёхмерных моделей

Реализовать просмотр трехмерных моделей, описание которых хранится в текстовых файлах.

3.38 Игра монополия

Реализовать компьютерную версию популярной игры. Обеспечить режим работы с двумя (и более) игроками, создать компьютерных игроков. Обеспечить возможность игры по сети.

3.39 Логическая игра “Крестики-нолики” на неограниченном поле

Приложение является реализацией известной логической игры “Крестики-нолики”. В данной реализации предусматривается игра двух игроков на неограниченном поле. Цель игры . построить непрерывную линию из пяти или более фишек (крестиков или ноликов) по горизонтали, вертикали или диагонали.

Возможны два режима игры, когда выигрывает тот, кто первым построит линию, или кто больше наберет очков за определенное количество времени. Во втором случае количество построенных линий не ограничено, каждая фишка в линии приносит игроку одно очко. Приложение должно обеспечивать начало новой игры на чистом поле, а также проверку соответствия действий игроков правилам игры и условия окончания игры. Роль одного из игроков (по выбору пользователя) может выполнять компьютер.

Для исключения возможности образования изолированных игр на одном поле и неоправданного увеличения размера поля следует установить максимально допустимое расстояние (не более пяти) от уже существующих фишек до новой, размещаемой игроком.

3.40 Логическая игра “Цепь”

Данное приложение является реализацией известной логической игры. В данной реализации предусматривается игра двух игроков на квадратном поле фиксированного размера. Цель игры - построить непрерывную линию, соединяющую горизонтальные или вертикальные (для каждого из игроков соответственно) стороны игрового поля, причем линия считается непрерывной, если фишки граничат друг с другом по горизонтали, вертикали или диагонали. Выигрывает тот игрок, который первым построит свою линию.

Приложение должно обеспечивать начало новой игры на чистом поле, а также проверку соответствия действий игроков правилам игры и условия окончания игры. Роль одного из игроков (по выбору пользователя) может выполнять компьютер.

3.41 Логическая игра “Точки”

Данное приложение является реализацией известной логической игры. В данной реализации предусматривается игра двух игроков на квадратном поле фиксированного размера. Цель игры . заполнить максимальное количество клеток игрового поля своими фишками. Игроки выполняют ходы по очереди. Ход игрока заключается в произвольной установке линии на границе двух ячеек, причем, если какая-либо ячейка оказывается со всех четырех сторон обрамлена линиями, то она отмечается фишкой данного игрока, а самому игроку предоставляется дополнительный ход, и т.д. Игра заканчивается, когда все игровое поле оказывается заполнено фишками. Выигрывает тот игрок, чьих фишек на поле больше на момент окончания игры.

Приложение должно обеспечивать начало новой игры на чистом поле, а также проверку соответствия действий игроков правилам игры и условия окончания игры. Роль одного из игроков (по выбору пользователя) может выполнять компьютер.

3.42 Логическая игра “Обратный тетрис”

Тетрис наоборот. Игрок выбирает фигуру и бросает, компьютер пытается установить ее в стакан. Цель игры - завалить компьютер.

3.43 Логическая игра “Шахматы”

Известная игра. Учить игре в шахматы компьютер не нужно. Достаточно реализовать игру двух человек за одним компьютером.

4 Список литературы

1. Подбельский. В.В. Язык C++: учебное пособие для вузов. М.: Финансы и статистика, 6-е изд-ие, 2002 г. 560с.
2. Тригуб С.Н. Стандарты программирования на C++. // Пер. с англ. — М.: Издательский дом "Вильямс" ", 2005.— 224 с.
3. Скотт Мейерс. Наиболее эффективное использование C++. 35 новых рекомендаций по улучшению ваших программ и проектов // Пер. с англ. — СПб.: "ДМК Пресс" · 2000 . – 300 с.
4. Ален И. Голуб. C & C++. Правила программирования. // М.: БИНОМ 1996г. — 272с.
5. Буч Гради. Объектно-ориентированное проектирование и анализ с примерами на C++. 2-е изд.

Приложение А. Пример пояснительной записки.

Агентство по образованию Российской Федерации

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

ЭКРАННЫЙ КАЛЬКУЛЯТОР

Пояснительная записка к курсовой работе по дисциплине
"Объектно-ориентированное программирование"

Выполнил:

студент гр.580-1

Песков М.А.

17.11.2007 г.

Руководитель проекта

доцент каф КСУП

Коцюбинский В.П.

17.11.2007 г.

РЕФЕРАТ

Курсовой проект 18 с., 3 табл., 2 прил.

Экранный калькулятор, Microsoft Visual C++, модуль, класс, интерфейс пользователя, метод рекурсивного спуска.

Пояснительная записка содержит проектную документацию программной системы «Экранный калькулятор». Программная система спроектирована в программе Microsoft Visio 2003. Программа реализована в интегрированной среде разработки «Microsoft Visual Studio 2005» на языке Visual C++. Пояснительная записка составлена в текстовом редакторе Microsoft Word 2003.

Агентство по образованию Российской Федерации

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ
(ТУСУР)

Утверждаю

Зав. кафедры КСУП

_____ Ю.А. Шурыгин

« ____ » _____ г.

ЗАДАНИЕ

на курсовую работу по предмету

"Объектно-ориентированное программирование"

Студенту Пескову Михаилу Андреевичу группа 580-1

1. Тема работы: Экранный калькулятор.

2. Срок сдачи студентом работы: 20.12.2007

3. Исходные данные.

Реализовать калькулятор, имеющий графический интерфейс пользователя, позволяющий вычислять арифметические выражения, заданные в командной строке.

4. Требования к программе.

Программа должна обеспечивать следующую функциональность:

- вычислять арифметические выражения в инфиксной записи, состоящие из операторов «+», «-», «*», «/», а также круглых скобок;
- учитывать приоритет операций в задаваемых выражениях;
- выводить пользователю сообщения о некорректной записи выражения с указанием вида ошибки и номера символа, где ошибка произошла;

- обеспечивать графический интерфейс пользователя с кнопками для ввода информации при помощи мышки;
- программу реализовать на языке C++.

4. Содержание пояснительной записки:

- анализ задания;
- система тестов;
- реализация;
- тестирование;
- заключение;
- список литературы.

5. Дата выдачи задания: 06.09.2007

Руководитель доцент каф КСУП

Коцубинский В.П. _____

Задание принял к исполнению

Песков М.А. _____

Содержание

1 Введение

2 Анализ задания

2.1 Входные данные системы

2.2 Выходные данные системы

2.3 Проектирование интерфейса пользователя

2.4 Проектирование модулей

2.5 Проектирование классов

3 Реализация

3.1 Интерфейс IErrorOuter

3.2 Класс CFloatStack

3.3 Класс CVirtualMashine

3.4 Класс CParser

3.5 Класс CMainDlg

4 Тестирование

4.1 Тестирование пользовательского интерфейса

4.2 Тестирование разборщика командной строки

4.3 Тестирование виртуальной машины

4.4 Тестирование стека вещественных чисел

5 Заключение

6 Список литературы

1 Введение

В ходе обучения студенту часто приходится сталкиваться с задачей вычисления сложных математических выражений. Для решения этой проблемы незаменимым является экранный калькулятор, который предоставляет возможность удобного ввода арифметического выражения и вычисления его результата.

Был произведён обзор нижеследующих программ, реализованных для решения этой задачи, имеющие указанные достоинства и недостатки.

- Стандартный калькулятор Microsoft Windows. Достоинства: данная программа является бесплатной, занимает мало место на жестком диске, в режиме инженерного калькулятора предоставляет обширный набор математических операций. Недостатки: нет возможности арифметических выражений в командной строке.

- MathCAD 11.A. Достоинства: наличие удобного текстового редактора, обширный набор арифметических операций, возможность вывода результатов в виде таблиц и графиков, возможность написания скриптов. Недостатки: большая стоимость, занимает много места на жестком диске после инсталляции.

Ни одна из вышеперечисленных программ не удовлетворила нашим требованиям, поэтому было решено разработать свой экранный калькулятор.

2 Анализ задания

2.1 Входные данные системы

Разрабатываемой системе должны поступать следующие данные от пользователя:

- арифметическое выражение в виде строки;
- нажатия на кнопки графического пользовательского интерфейса, формирующие командную строку.

2.2 Выходные данные системы

Разрабатываемая система должна выдавать пользователю следующие данные:

- значение вычисленного выражения;
- описание ошибки в арифметическом выражении;
- позиция символа, где произошла ошибка в арифметическом выражении.

2.3 Проектирование интерфейса пользователя

Был спроектирован пользовательский интерфейс, представленный на рисунке 2.1.

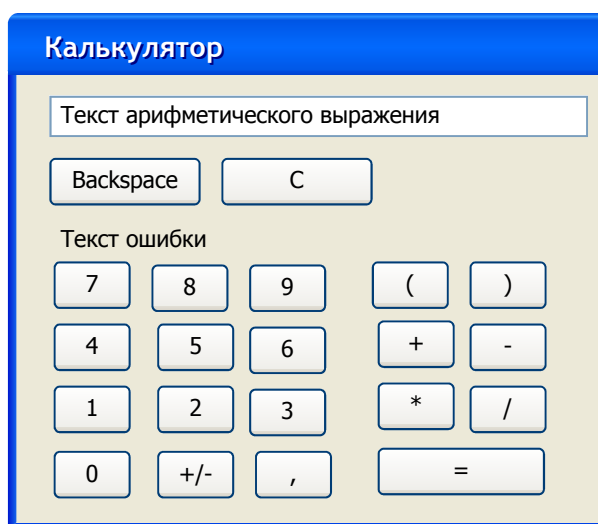


Рисунок 2.1 – спроектированный пользовательский интерфейс

Для подсчёта арифметического выражения, пользователь должен ввести текст выражения в поле ввода, которое находится наверху окна. Ввод выражения может осуществляться при помощи соответствующих кнопок на поверхности формы. При нажатии на кнопку «Backspace» должен затираться символ, стоящий слева от каретки в поле ввода. При нажатии на кнопку «C»

текст арифметического выражения должен очищаться. При нажатии на кнопку «=» арифметическое выражение должно посчитаться, а результат вывести в поле ввода арифметического выражения. Если в выражении произошла ошибка, на форму должен быть выведен текст ошибки, а каретка в поле ввода – перемещена на то место, где ошибка была обнаружена.

2.4 Проектирование модулей

В процессе проектирования модулей была построена схема взаимодействия модулей, представленная на рисунке 2.2.

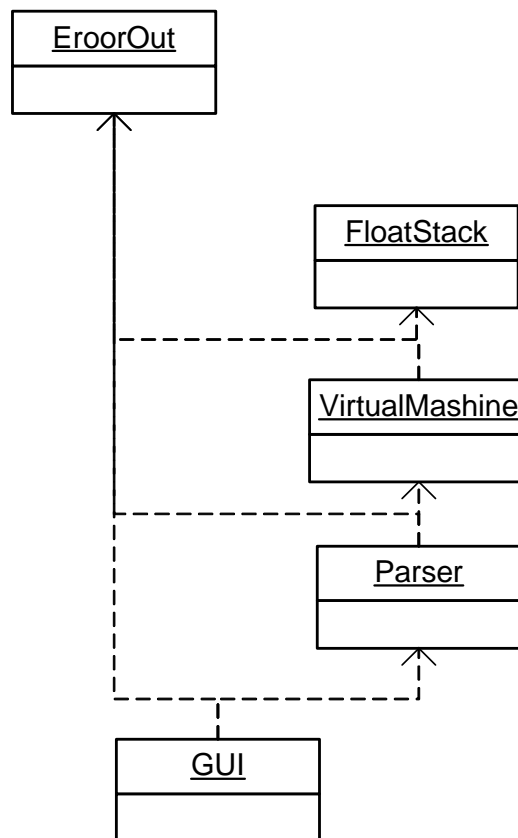


Рисунок 2.2 – Схема взаимодействия модулей

2.5 Проектирование классов

На основе спроектированной схемы взаимодействия модулей, используя идеологию «Один модуль – один класс», была получена диаграмма взаимодействия классов, представленная на рисунке 2.3.

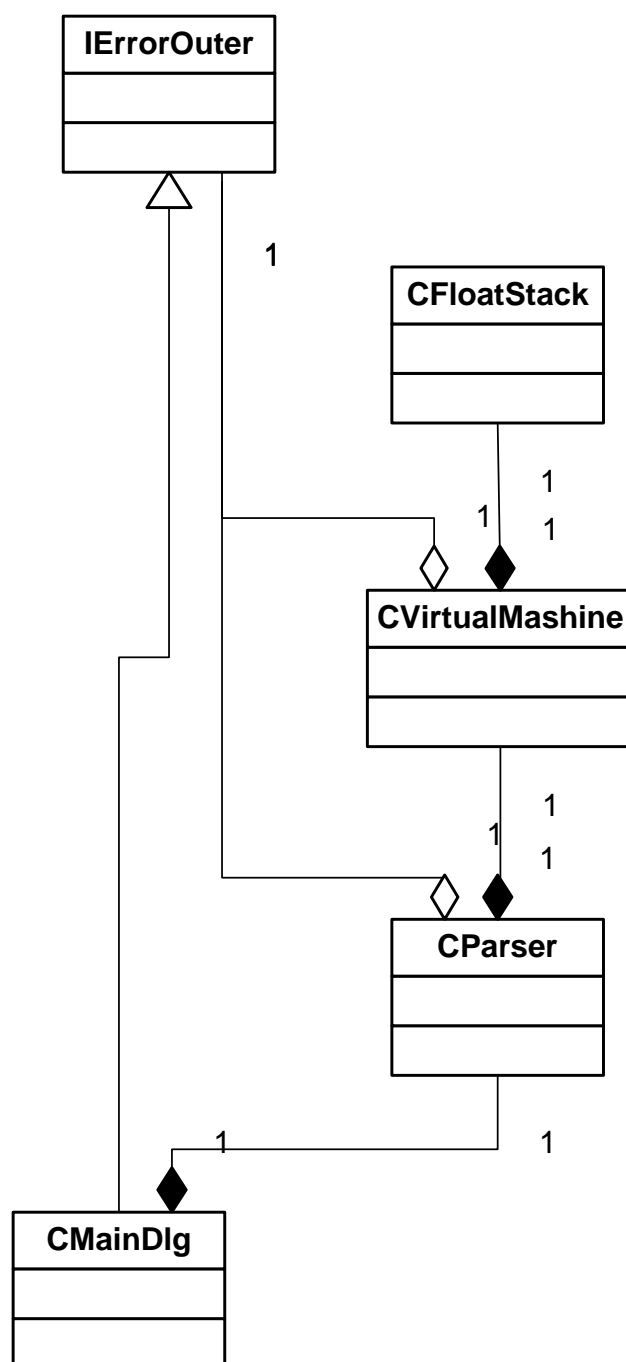


Рисунок 2.3 – Диаграмма взаимодействия классов

На данной схеме отображены нижеследующие классы.

IErrorOuter – абстрактный базовый класс «Вывод ошибок». Служит для представления интерфейса вывода сообщений об ошибках. В конечном итоге вывод сообщений об ошибках будет осуществляться в форме калькулятора.

CFloatStack – стек вещественных чисел. Планируется разработать на основе класса `vector` библиотеки STL.

CVirtualMashine – виртуальная машина. Выполняет команды калькулятора. Команды подаются в польской нотации.

CParser – разборщик командной строки. Разбирает командную строку методом рекурсивного спуска [1]. Разобранные команды переводит в польскую нотацию и передаёт на вход виртуальной машине.

3 Реализация

Согласно спроектированной схемы использования модулей (рисунок 2.2) и диаграммы взаимодействия классов (рисунок 2.3) был разработан набор классов с нижеследующими входными и выходными параметрами.

3.1 Интерфейс IErrorOuter

В данном интерфейсе реализована функция

```
virtual void ShowError(const CString& strErrorText , int
iSymbNum) = 0;
```

которая выводит сообщение об ошибке.

Параметры этой функции:

CString strErrorText – текст выводимой ошибки;

int iSymbNum - номер символа, содержащего ошибку.

3.2 Класс CFloatStack

Методы данного класса описаны в таблице 3.1.

Таблица 3.1 – Методы класса CFloatStack

Метод	Описание
CFloatStack();	Конструктор.
~CFloatStack();	Деструктор.
void PushNumber(double dNum);	Положить в стек число. Параметры: <i>double dNum</i> - вещественное число, закладываемое в стек.
bool PopNumber(double& dNum);	Извлечь вещественное число из стека. Возвращает <i>true</i> , если число было извлечено успешно и <i>false</i> – если стек пустой. Параметры: <i>double& dNum</i> – ссылка на переменную, куда будет извлечено число.

3.3 Класс CVirtualMashine

Методы данного класса описаны в таблице 3.2.

Таблица 3.2 – Методы класса CVirtualMashine

Метод	Описание
CVirtualMashine(IErrorOuter ErrorOuter);	Конструктор. Параметры: <i>IErrorOuter ErrorOuter</i> – интерфейс вывода ошибок.
~CVirtualMashine();	Деструктор.
bool PushNumber(double dNum, int iSymbNum);	Положить в стек вещественное число. Возвращает, корректно ли завершилась данная операция. Параметры: <i>double dNum</i> – - вещественное число, закладываемое в стек; <i>int iSymbNum</i> – номер разбираемого символа.
bool PopNumber(double& dNum, int iSymbNum);	Извлечь вещественное число из стека. Возвращает, корректно ли завершилась данная операция. Параметры: <i>double& dNum</i> – ссылка на переменную, куда будет извлечёно число; <i>int iSymbNum</i> – номер разбираемого символа.
bool DoOperation(const CString& strOperationName, int iSymbNum);	Выполнить арифметическую операцию. Возвращает, корректно ли завершилась данная операция. Параметры: <i>const CString& strOperationName</i> – имя

	операции; <i>int iSymbNum</i> – номер разбираемого символа.
--	--

3.4 Класс CParser

Методы данного класса описаны в таблице 3.2.

Таблица 3.2 – Методы класса CParser

Метод	Описание
CParser(IErrorOuter ErrorOuter);	Конструктор. Параметры: <i>IErrorOuter ErrorOuter</i> – интерфейс вывода ошибок.
~CParser();	Деструктор.
bool ParseString(const CString& strParsedString, double& dResValue);	Разобрать строку. Возвращает, корректно ли завершилась данная операция. Параметры: <i>const CString& strParsedString</i> – разбираемая строка; <i>double& dNum</i> – ссылка на переменную, куда будет записано вычисленное значение.

3.5 Класс CMainDlg

Класс CMainDlg был реализован при помощи диалогового окна, созданного при помощи мастера MFC. Наследован от интерфейса IErrorOuter, публичных методов не содержит.

4 Тестирование

Для каждого из модулей, описанных в разделе 3, были разработаны тестовые модули. Ниже приведены данные, на которых эти модули были протестированы. После процесса отладки программа стала реагировать адекватно на все нижеперечисленные тесты.

4.1 Тестирование пользовательского интерфейса

Данные для тестирования пользовательского интерфейса приведены в таблице 4.1 и должны задаваться как с клавиатуры, так и путём нажатия на кнопки экранной формы.

Таблица 4.1 – данные для тестирования пользовательского интерфейса

Входные данные	Выходные данные
2+3	5
2,0+3,0	5
32*22	704
5,5/3,5	1,5714
2+3*5	17
(2+3)*5	25
(2,5+4,6)/4,1-3,1	-1,368292
(2,5*(3,6-2,2)+4,6)/4,1-3,1	-2,441634
55/0	Ошибка «деление на ноль», символ 3
5a6-32	Ошибка «неопознанный символ», символ 2
2++4	Ошибка «ошибка синтаксиса», символ 3
5*(43+1	Ошибка «пропущена закрывающая фигурная скобка», символ 7
3*)32-54)	Ошибка «ошибка синтаксиса», символ 3

4.2 Тестирование разборщика командной строки

Данные разборщика командной строки совпадают с данными для тестирования пользовательского интерфейса и приведены в таблице 4.1.

4.3 Тестирование виртуальной машины

Тестирование виртуальной должно быть проведено при помощи последовательности команд, представленных в таблице 4.2.

Таблица 4.2 – команды для тестирования виртуальной машины

Входные команды	Выходные данные
положить в стек 1; положить в стек 2,5; положить в стек 3,2; вынуть число из стека; вынуть число из стека; вынуть число из стека.	3,2; 2,5; 1.
положить в стек 1,1; положить в стек 2,5; произвести операцию «+»; вынуть число из стека.	3,6.
положить в стек 32; положить в стек 22; произвести операцию «*»; вынуть число из стека.	704.
положить в стек 5,5; положить в стек 3,5; произвести операцию «/»; вынуть число из стека.	1,5714.
положить в стек 3,6; положить в стек 2,2; произвести операцию «-»; положить в стек 2,5; произвести операцию «*»; положить в стек 4,6; произвести операцию «+»; положить в стек 4,1; положить в стек 3,1; произвести операцию «-»; вынуть число из стека.	-2,441634.

положить в стек 55; положить в стек 0; произвести операцию «/».	ошибка «деление на ноль».
положить в стек 2; положить в стек 4; произвести операцию «+»;произвести операцию «+».	ошибка «стек пустой».
положить в стек 2; вынуть число из стека; вынуть число из стека.	2; ошибка «стек пустой».

4.4 Тестирование стека вещественных чисел

Тестирование виртуальной машины должно быть проведено при помощи последовательности команд, представленных в таблице 4.3.

Таблица 4.3 – команды для стека вещественных чисел

Входные команды	Выходные данные
положить в стек 1,1; положить в стек 2,2; положить в стек 3,3; вынуть число из стека; вынуть число из стека; вынуть число из стека.	3,3; 2,2; 1,1.
положить в стек 1,1; вынуть число из стека; вынуть число из стека.	1,1; ошибка «стек пустой».

5 Заключение

В процессе проделанной работы был разработан калькулятор, имеющий графический интерфейс пользователя, позволяющий вычислять арифметические выражения, заданные в командной строке. Техническое задание было выполнено полностью. К разработанной программе была составлена система автоматических тестов и техническая документация.

Дальнейшее развитие данной программы может состоять в наращении функциональности по следующим направлениям:

- добавление новых арифметических операций (тригонометрические функции, возведение в степень, извлечение корня, логарифм и т.д.);
- добавление кнопок запоминания текущего значения в памяти;
- добавление возможности составления программ.

6 Список использованных источников

1. Альфред В. Ахо, Рави Сети, Джеффри Д. Ульман. Компиляторы: принципы, технологии и инструментарий. – СПб.: Вильямс, 2003. – 768с. , с ил.