ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

Государственное образовательное учреждение
высшего профессионального образования

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ

Управление качеством программного обеспечения

Методические указания к выполнению лабораторных работ

Санкт-Петербург

2017

Составитель: П.А.Степанов

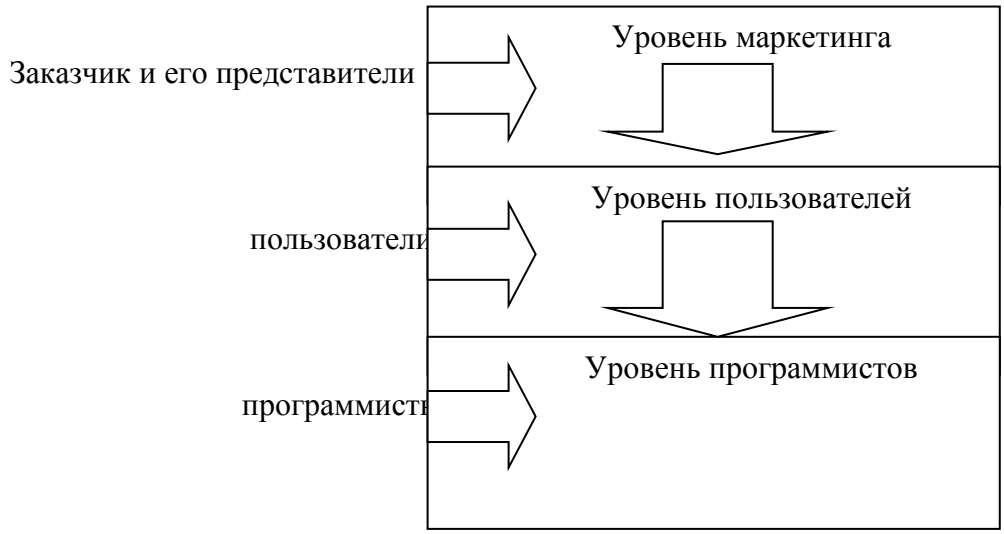
Рецензент:

Лабораторная работа номер 1- Тестирование требований

Тестирование программного обеспечения должно начинаться с самого раннего этапа – этапа формулировки требований. Основными проблемами требований являются:

- Некорректность
- Двусмысленность
- Неполнота
- Непроверяемость
- Несвязанность между требованиями разных уровней (нетрассируемость)
- Непонятность

Требования к программному обеспечению формулируются на трех уровнях. Самый верхний – уровень маркетинговых требований – содержит самые общие требования. Следующий уровень – уровень пользователей – содержит, обычно, описание бизнес-процессов. На самом низком уровне – уровне программистов – используются мелкие технические требования (функциональные требования).



В данной работе все требования считаются сформулированными на уровне маркетинга.

Задание на лабораторную работу.

Вы – системный аналитик. Ваша задача написать техническое задание на предмет, маркетинговые требования к которому Вам были переданы в соответствии с вариантом (в рамках лабораторной работы делать этого не надо, нужно только решить, достаточно ли для

этого информации). Для этого Вы должны раскрыть требования маркетинга на уровень пользователей. В процессе этого у Вас могут возникать различные проблемы.

В лабораторной работе необходимо проанализировать требования заказчика, указать на проблемы в требованиях (каждую ошибку отнести к соответствующей категории) и скорректировать требования таким образом, чтобы в результате получился предмет указанного наименования. Для каждого пункта требований описать, каким образом будет производиться его проверка.

Указания.

Типовой проблемой при выполнении лабораторной работы является попытка придраться к каждому слову спецификации. На самом деле, предполагается моделирование реальной ситуации – Вам прислали требования заказчика и Вы делаете по ним техническое задание, по мере выполнения которого обнаруживаете разнообразные проблемы, препятствующие этому.

Заказчик никогда сам не пишет техническое задание! Когда в ответ на требование вида “разработать двухкамерный холодильник” Вы пишете замечания вида «неизвестно какого он должен быть размера, цвета и формы» Вы, тем самым, говорите заказчику, что он не сделал Вашу работу, потому что определить размер, цвет и форму, если они не указаны закахчиком, – задача аналитика.

Еще одной важной задачей является произвести на заказчика впечатление грамотного специалиста, а не человека, который не умеет делать свою работу.

Пример выполнения.

Спецификация на разработку стула.

1. Стул должен иметь четыре ножки и горизонтальную поверхность для сидения.
2. Стул должен иметь возможность регулирования высоты
3. Стул должен быть удобным.
4. Стул должен иметь высоту 60 сантиметров.
5. Стул должен весить не более 500 грамм
6. Стул должен быть легко перемещаем по помещению.
7. Стул не должен царапать паркет при перемещении
8. Стул должен использовать только нетоксичные материалы.

Рассмотрим требования по очереди.

Требование номер один интересно тем, что в нем упоминаются только ножки и поверхность для сидения. Стул, у которого нет спинки, называется “табурет”. Имеет смысл

уточнить, имеется ли в самом деле в виду табурет или спинка была просто забыта при описании.

Требование номер два обычно используется для стульев, имеющих одну ножку, хотя принципиальных проблем с реализацией для четырехногого стула не имеется.

Требование номер три невозможно проверить, его можно скорректировать разнообразными способами, например, “дизайн стула утверждается заказчиком” в том смысле, что сперва будет утвержден дизайн и только после этого будет продолжена реализация.

Требование номер 4 явно противоречит требованию номер два, необходимо указать диапазон изменения высоты либо отказаться от требования номер два. Кроме того, возникает вопрос – является ли указанная высота высотой сиденья или спинки.

1 Спецификация на разработку холодильника

Необходимо разработать двухкамерный холодильник на базе системы андроид, отвечающий следующим требованиям:

- Холодильник двухкамерный
- При захлопывании дверцы она всегда обеспечивает плотное прижатие, вне зависимости от того, с какой силой было произведено это действие.
- Холодильник имеет интерфейс через сенсорный дисплей с локал изацией, поддерживающий следующие языки: Русский, Английский
- Управление температурой в холодильной и морозильной камерах осуществляется с дисплея.
- Когда дверца холодильника открыта, дисплей показывает предупреждающее сообщение и не разрешает управление температурой
- Когда дверца холодильника закрыта, дисплей отображает текущую температуру в холодильной и морозильной камерах.
- При изменении температуры на N градусов фактическая температура в камере должна измениться через N минут

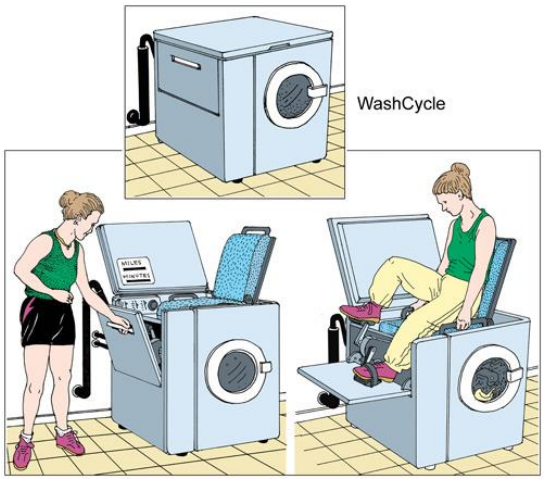
2 Спецификация на разработку пылесоса

Необходимо разработать пылесос на базе системы андроид, отвечающий следующим требованиям:

- Пылесос способен убирать пыль и мелкий мусор
- Пылесос обеспечивает всасывание воздуха с мощностью 1600 Ватт
- Масса пылесоса в процессе работы не должна превышать 5 килограмм
- Пылесос может быть использован для сбора пыти на любых поверхностях и под любыми предметами мебели
- На пылесосе должна быть предусмотрена ручка
- Заряда пылесос должно хватать на 1 час работы

3 Спецификация на разработку стиральной машины

Необходимо разработать стиральную машину, отвечающую следующим требованиям:



- Стиральная машина должна уметь стирать белье.
- Минимальная загрузка должна составлять пять килограмм
- Стирка должна осуществляться в двух режимах – быстрая и полная, а также машина должна уметь осуществлять полоскание
- Стиральная машина должна подключаться к водопроводной трубе, сама закачивать воду, нагревать ее до нужной температуры, по окончании стирки – сливать
- У машины должен быть дисплей, демонстрирующий пользователю полезную информацию.
- Машина должна подключаться к WiFi

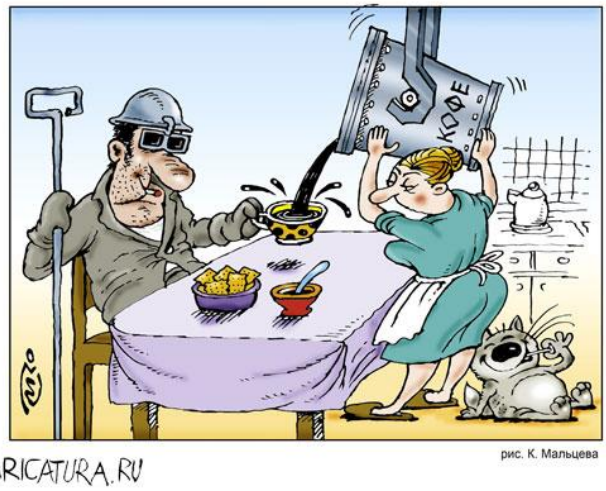
4 Спецификация на разработку микроволновой печи

Необходимо разработать микроволновую печь, отвечающую следующим требованиям:

- Микроволновая печь должна уметь нагревать продукты, а также осуществлять разморозку.
- Микроволновая печь должна иметь дверцу
- Микроволновая печь во включенном состоянии не должна производить опасное для человека излучение
- Нагревать продукты в микроволновой печи нужно при закрытой дверце
- Микроволновая печь должна позволять помещать вместе с продуктами металлические столовые приборы и посуду, при этом они не должны нагреваться или искрить
- Микроволновая печь должна иметь подсветку.
- Микроволновая печь не должна сушить продукты, которые в ней готовятся

5 Спецификация на разработку электрокофеварки

Необходимо разработать электрокофеварку, отвечающий следующим требованиям:



- Кофеварка должна уметь варить кофе
- Кофеварка должна иметь кнопку включения, ручку и крышку над емкостью для наливания воды
- Кофеварка должна работать от электричества
- Кнопка должна включаться, только если крышка закрыта
- Воду в кофеварку можно наливать, только если крышка закрыта
- Кофеварка должен поддерживать протокол HTCPCP (RFC 2324). Заказчик особенно настаивает на этом требовании и отказывается его убирать.

- Кофеварка должна быть красивой

6 Спецификация на разработку газовой плиты

Необходимо разработать газовую плиту, отвечающую следующим требованиям:

- Плита должна иметь четыре конфорки и духовку со стеклянной дверцей.
- Когда духовка открыта, газ подаваться не должен.
- На передней панели плиты должен быть расположен датчик температуры
- Духовка должна быть оборудована таймером, выключающим газ, и зуммером с настраиваемой мелодией, срабатывающим по завершении
- На плите можно готовить весь набор популярных блюд
- Плита должна исключать возможность возникновения пожара

7 Спецификация на разработку велосипеда

Необходимо разработать велосипед, отвечающий следующим требованиям:

- У велосипеда должно быть удобное, регулирующееся по высоте сиденье
- Велосипед должен поддерживать 16 скоростей
- Велосипед не должен сильно подпрыгивать на небольших бугорках
- Велосипед должен позволять перевозить двух пассажиров

8 Спецификация на разработку дырокола

Необходимо разработать дырокол, отвечающий следующим требованиям:

- Дырокол должен делать два отверстия в листе бумаги на стандартном для папок расстоянии
- Дырокол должен позволять вставить не менее 20 листов бумаги
- Габариты дырокола не должны превышать 20 сантиметров в ширину и 10 сантиметров в высоту
- Вес дырокола со вставленной бумагой не должен быть более 200 грамм
- Дырокол не должен иметь острых или выступающих краев, которыми можно пораниться или поранить другого
- Дырокол должен иметь информационное табло, отображающее актуальную информацию

9 Спецификация на разработку шкафа-купе

Необходимо разработать шкаф-купе, отвечающий следующим требованиям:

- Шкаф имеет три вертикальные секции и три двери
- Двери шкафа должны крепиться на горизонтальных направляющих
- Высота шкафа должна составлять 2 метра
- Двери шкафа должны быть созданы с таким расчетом, чтобы ими ничего нельзя было прищемить
- Глубина шкафа должна быть выбрана таким образом, чтобы в него помещался велосипед
-

А Задание повышенной сложности

Самостоятельно разработать спецификацию, содержащую осмысленные ошибки и описать их.

Лабораторная работа номер 2- Функциональное тестирование методом черного ящика

В рамках лабораторной работы необходимо произвести функциональное тестирование кода, оценить его покрытие и качество тестов

Задание на лабораторную работу.

1. Разработать функцию в соответствии со своим вариантом. **Внимание. Варианты на лабораторную работу раздаются заново**.
2. Разработать функциональные тесты для написанного кода методом черного ящика. Добиваться 100% прохождения тестов не нужно. Необходимо описать принципы выбора тестов.

К отчету должна быть приложена спецификация на тесты в следующем формате:

<начало примера>

Тестируемая функциональность

Имя теста	Описание сценария	Входные данные	Выходные данные

<конец примера>

Например:

Функция add(int,int)

Имя теста	Описание сценария	Входные данные	Выходные данные
addTwoPositive	сложение двух положительных чисел, проверка	Первый параметр 1 Второй параметр 2	Результат вызова 3

	результата		
addTwoNegative	сложение двух отрицательных чисел, проверка результата	Первый параметр -1 Второй параметр -2	Результат вызова -3
addTwoTimes	Сложение двух положительных чисел, затем сложение двух отрицательных чисел, затем сложение результатов	При первом вызове Первый параметр 1 Второй параметр 2 При втором вызове Первый параметр -1 Второй параметр -2	Результат первого вызова 3, Результат второго вызова -3, Результат третьего вызова 0

Тесты должны быть описаны достаточно недвусмысленно чтобы их содержание было понятно без заглядывания в код!

Решать задачу можно на языках C++ и Java. Желающим выбрать другой язык необходимо убедиться, что

А) для этого языка существуют необходимые инструменты

Б) он в состоянии объяснить преподавателю, что именно написано в его коде

Отчет должен содержать описание применения всех указанных шагов. Не следует добиваться, чтобы 100% покрытие и полная корректность тестов были достигнуты уже в пункте 2.Значительно более интересным является процесс исправления недостатков тестовой базы.

Вариант 1. Компилятор простых арифметических выражений, например 2+(-5)*(7-8). Вход и выход в виде строк

Вариант 2. Функция поиска пути в неориентированном графе методом поиска в ширину. На вход подается граф и две вершины. На выходе – путь между этими вершинами.

Вариант 3. Функция поиска пути в неориентированном графе методом поиска в глубину. На вход подается граф и две вершины. На выходе – путь между этими вершинами.

Вариант 4. Функция поиска пути в неориентированном графе методом A*. На вход подается карта (граф с географическими координатами вершин) и два угла. На выходе – путь между этими узлами.

Вариант 5. Функция балансировки двоичного дерева

Вариант 6. Функция, рассчитывающая контур пересечения двух треугольников

Вариант 7. Хеш-таблица, не перетирающая элементы при вводе значений с совпадающим ключом, а хранящая список таких элементов и, соответственно, возвращающая их методом get. Метод – двойное хеширование

Вариант 8. Функция, рассчитывающая следующий ход в игре крестики-нолики на доске заданного размера и для заданной длины выигрышной последовательности путем построения полного дерева решений(например, доска 5 на 5 и длина выигрышной последовательности 4)

Вариант 9 Функция, производящая поиск заданного набора строк в текстовом файле. Поиск должен уметь находить любую строку из набора, при этом должен правильно обрабатывать переносы текста. Использовать алгоритм Ахо-Корасик.

Вариант 10 Парсер, использующий простые регулярные выражения, вводимые с клавиатуры, содержащие управляющие конструкции . – любой символ, * - 0 и более символов, + - 1 и более символов (вводится регулярное выражение и строка, результатом является позиция, с которой это выражение встречается в тексте)

Вариант 11 Молекула ДНК состоит из последовательностей нуклеотидов А, Г, Ц и У. Несколько одинаковых молекул известной длины были нарезаны на фрагменты произвольной длины. Функция восстанавливает исходную молекулу в том случае, если это возможно сделать единственным образом

Пример: АГЦЦГГУААЦЦ нарезана на фрагменты АГЦЦ, ЦГГУ , ГГУАА и УААЦЦ.

Пример невосстанавливаемой последовательности: АГЦЦГГУААЦЦ нарезана на фрагменты АГЦЦ, ГГУАА и УААЦЦ.

Вариант алгоритма решения. В памяти строится ориентированный граф, в вершинах которого находятся фрагменты, а связи соединяют два фрагмента, если фрагмент-источник может быть слева от фрагмента-приемника. Далее в графе ищутся все пути и для каждого проверяется, что он содержит в себе все фрагменты. Если такой путь один, то задача считается решенной

Вариант 12 В матричной форме задается система линейных уравнений, необходимо ее решить (например, методом Гаусса).

Вариант 13. Реализовать структуру «Список с пропусками”, см. https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D1%81_%D0%BF%D1%80%D0%BE%D0%BF%D1%83%D1%81%D0%BA%D0%B0%D0%BC%D0%B8

Реализовать функции добавления, удаления и поиска.

Вариант 14. В заданном произвольном тексте найти все повторяющиеся фрагменты текста длиной не менее трех слов (без использования стемминга, т.е. слова в различных склонениях и падежах считаются разными, знаки препинания не учитываются). При этом для каждого повторяющегося фрагмента должна указываться максимальная длина, например, для данного текста

- «поиска пути в неориентированном графе методом поиска в” встречается дважды

- «поиска пути в неориентированном графе методом” встречается трижды
- Более короткие части отдельно не встречаются, поэтому не рассматриваются

Лабораторная работа номер 3- Функциональное тестирование методом белого ящика

В рамках лабораторной работы необходимо произвести функциональное тестирование кода методом белого ящика (всех ветвей)

Задание на лабораторную работу.

1. Разработать функцию в соответствии со своим вариантом. **Внимание. Варианты на лабораторную работу раздаются заново**.
2. Разработать функциональные тесты для написанного кода методом белого ящика. Добиваться 100% прохождения тестов не нужно. Необходимо описать принципы выбора тестов.

К отчету должна быть приложена спецификация на тесты в следующем формате:

<начало примера>

Тестируемая функциональность

Имя теста	Описание сценария	Входные данные	Выходные данные

<конец примера>

Например:

Имеется функция sign (x){

If (x>0) return 1;else

If (x<0) return -1;

Else return 0;

}

Функция sign(int)

Имя теста	Описание сценария	Входные данные	Выходные данные

positiveSign	x>0: true x<0 false	Входной параметр 10	Результат вызова 1
negativeSign	x>0:false x<0 true	Входной параметр -10	Результат вызова -1
zeroSign	x>0: false x<0 false	Входной параметр 0	Результат вызова 0

Тесты должны быть описаны достаточно недвусмысленно чтобы их содержание было понятно без заглядывания в код!

Решать задачу можно на языках C++ и Java. Желающим выбрать другой язык необходимо убедиться, что

А) для этого языка существуют необходимые инструменты

Б) он в состоянии объяснить преподавателю, что именно написано в его коде

Отчет должен содержать описание применения всех указанных шагов. Не следует добиваться, чтобы 100% покрытие и полная корректность тестов были достигнуты уже в пункте 2.Значительно более интересным является процесс исправления недостатков тестовой базы.

Вариант 1. Компилятор простых арифметических выражений, например 2+(-5)*(7-8). Вход и выход в виде строк

Вариант 2. Функция поиска пути в неориентированном графе методом поиска в ширину. На вход подается граф и две вершины. На выходе – путь между этими вершинами.

Вариант 3. Функция поиска пути в неориентированном графе методом поиска в глубину. На вход подается граф и две вершины. На выходе – путь между этими вершинами.

Вариант 4. Функция поиска пути в неориентированном графе методом A*. На вход подается карта (граф с географическими координатами вершин) и два угла. На выходе – путь между этими узлами.

Вариант 5. Функция балансировки двоичного дерева

Вариант 6. Функция, рассчитывающая контур пересечения двух треугольников

Вариант 7. Хеш-таблица, не перетирающая элементы при вводе значений с совпадающим ключом, а хранящая список таких элементов и, соответственно, возвращающая их методом get. Метод – двойное хеширование

Вариант 8. Функция, рассчитывающая следующий ход в игре крестики-нолики на доске заданного размера и для заданной длины выигрышной последовательности путем построения полного дерева решений(например, доска 5 на 5 и длина выигрышной последовательности 4)

Вариант 9 Функция, производящая поиск заданного набора строк в текстовом файле. Поиск должен уметь находить любую строку из набора, при этом должен правильно обрабатывать переносы текста. Использовать алгоритм Ахо-Корасик.

Вариант 10 Парсер, использующий простые регулярные выражения, вводимые с клавиатуры, содержащие управляющие конструкции . – любой символ, * - 0 и более символов, + - 1 и более символов (вводится регулярное выражение и строка, результатом является позиция, с которой это выражение встречается в тексте)

Вариант 11 Молекула ДНК состоит из последовательностей нуклеотидов А, Г, Ц и У. Несколько одинаковых молекул известной длины были нарезаны на фрагменты произвольной длины. Функция восстанавливает исходную молекулу в том случае, если это возможно сделать единственным образом

Пример: АГЦЦГГУААЦЦ нарезана на фрагменты АГЦЦ, ЦГГУ , ГГУАА и УААЦЦ.

Пример невосстанавливаемой последовательности: АГЦЦГГУААЦЦ нарезана на фрагменты АГЦЦ, ГГУАА и УААЦЦ.

Вариант алгоритма решения. В памяти строится ориентированный граф, в вершинах которого находятся фрагменты, а связи соединяют два фрагмента, если фрагмент-источник может быть слева от фрагмента-приемника. Далее в графе ищутся все пути и для каждого проверяется, что он содержит в себе все фрагменты. Если такой путь один, то задача считается решенной

Вариант 12 В матричной форме задается система линейных уравнений, необходимо ее решить (например, методом Гаусса).

Вариант 13. Реализовать структуру «Список с пропусками”, см. https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D1%81_%D0%BF%D1%80%D0%BE%D0%BF%D1%83%D1%81%D0%BA%D0%B0%D0%BC%D0%B8

Реализовать функции добавления, удаления и поиска.

Вариант 14. В заданном произвольном тексте найти все повторяющиеся фрагменты текста длиной не менее трех слов (без использования стемминга, т.е. слова в различных склонениях и падежах считаются разными, знаки препинания не учитываются). При этом для каждого повторяющегося фрагмента должна указываться максимальная длина, например, для данного текста

- «поиска пути в неориентированном графе методом поиска в” встречается дважды
- «поиска пути в неориентированном графе методом” встречается трижды
- Более короткие части отдельно не встречаются, поэтому не рассматриваются

Лабораторная работа номер 4- Оценка качества тестовой базы

В рамках лабораторной работы необходимо произвести функциональное тестирование кода методом белого ящика (всех ветвей)

Задание на лабораторную работу.

3. Оценить по отдельности и вместе покрытие тестами, разработанными в лабораторной работе номер 2 и 3.
4. Описать недостающие тесты
5. Выполнить инъекцию багов, оценить качество разработанных тестов.

К отчету должна быть приложена спецификация на тесты в следующем формате:

<начало примера>

Тестируемая функциональность

Имя теста	Описание сценария	Входные данные	Выходные данные

<конец примера>

Решать задачу можно на языках C++ и Java. Желающим выбрать другой язык необходимо убедиться, что

А) для этого языка существуют необходимые инструменты

Б) он в состоянии объяснить преподавателю, что именно написано в его коде

Отчет должен содержать описание применения всех указанных шагов. Не следует добиваться, чтобы 100% покрытие и полная корректность тестов были достигнуты уже в пункте 2.Значительно более интересным является процесс исправления недостатков тестовой базы.

Вариант 1. Компилятор простых арифметических выражений, например 2+(-5)*(7-8). Вход и выход в виде строк

Вариант 2. Функция поиска пути в неориентированном графе методом поиска в ширину. На вход подается граф и две вершины. На выходе – путь между этими вершинами.

Вариант 3. Функция поиска пути в неориентированном графе методом поиска в глубину. На вход подается граф и две вершины. На выходе – путь между этими вершинами.

Вариант 4. Функция поиска пути в неориентированном графе методом A*. На вход подается карта (граф с географическими координатами вершин) и два угла. На выходе – путь между этими узлами.

Вариант 5. Функция балансировки двоичного дерева

Вариант 6. Функция, рассчитывающая контур пересечения двух треугольников

Вариант 7. Хеш-таблица, не перетирающая элементы при вводе значений с совпадающим ключом, а хранящая список таких элементов и, соответственно, возвращающая их методом get. Метод – двойное хеширование

Вариант 8. Функция, рассчитывающая следующий ход в игре крестики-нолики на доске заданного размера и для заданной длины выигрышной последовательности путем построения полного дерева решений(например, доска 5 на 5 и длина выигрышной последовательности 4)

Вариант 9 Функция, производящая поиск заданного набора строк в текстовом файле. Поиск должен уметь находить любую строку из набора, при этом должен правильно обрабатывать переносы текста. Использовать алгоритм Ахо-Корасик.

Вариант 10 Парсер, использующий простые регулярные выражения, вводимые с клавиатуры, содержащие управляющие конструкции . – любой символ, * - 0 и более символов, + - 1 и более символов (вводится регулярное выражение и строка, результатом является позиция, с которой это выражение встречается в тексте)

Вариант 11 Молекула ДНК состоит из последовательностей нуклеотидов А, Г, Ц и У. Несколько одинаковых молекул известной длины были нарезаны на фрагменты произвольной длины. Функция восстанавливает исходную молекулу в том случае, если это возможно сделать единственным образом

Пример: АГЦЦГГУААЦЦ нарезана на фрагменты АГЦЦ, ЦГГУ , ГГУАА и УААЦЦ.

Пример невосстанавливаемой последовательности: АГЦЦГГУААЦЦ нарезана на фрагменты АГЦЦ, ГГУАА и УААЦЦ.

Вариант алгоритма решения. В памяти строится ориентированный граф, в вершинах которого находятся фрагменты, а связи соединяют два фрагмента, если фрагмент-источник может быть слева от фрагмента-приемника. Далее в графе ищутся все пути и для каждого проверяется, что он содержит в себе все фрагменты. Если такой путь один, то задача считается решенной

Вариант 12 В матричной форме задается система линейных уравнений, необходимо ее решить (например, методом Гаусса).

Вариант 13. Реализовать структуру «Список с пропусками”, см. https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B8%D1%81%D0%BE%D0%BA_%D1%81_%D0%BF%D1%80%D0%BE%D0%BF%D1%83%D1%81%D0%BA%D0%B0%D0%BC%D0%B8

Реализовать функции добавления, удаления и поиска.

Вариант 14. В заданном произвольном тексте найти все повторяющиеся фрагменты текста длиной не менее трех слов (без использования стемминга, т.е. слова в различных склонениях и падежах считаются разными, знаки препинания не учитываются). При этом для каждого повторяющегося фрагмента должна указываться максимальная длина, например, для данного текста

- «поиска пути в неориентированном графе методом поиска в” встречается дважды
- «поиска пути в неориентированном графе методом” встречается трижды
- Более короткие части отдельно не встречаются, поэтому не рассматриваются

Лабораторная работа номер 5- Функциональное тестирование оконного интерфейса

В рамках лабораторной работы необходимо произвести функциональное тестирование Rich интерфейса приложения

Задание на лабораторную работу.

6. Взять задание из лабораторной работы номер 2 и добавить к нему оконный интерфейс (Rich). Реализовать поля ввода, поле вывода результата, кнопку расчета результата и кнопку отмены. Сделать так, чтобы кнопку расчета можно было нажать, только если все поля ввода заполнены
7. Разработать функциональные сценарии и реализовать их с помощью одного из средств автоматизации

Решать задачу можно на языках C++ и Java. Желающим выбрать другой язык необходимо убедиться, что

А) для этого языка существуют необходимые инструменты

Б) он в состоянии объяснить преподавателю, что именно написано в его коде

Отчет должен содержать описание применения всех указанных шагов.

Лабораторная работа номер 6- Функциональное тестирование веб-приложения

В рамках лабораторной работы необходимо произвести функциональное тестирование Thin интерфейса приложения

Задание на лабораторную работу.

8. Взять задание из лабораторной работы номер 2 и добавить к нему Web-интерфейс. Реализовать поля ввода, поле вывода результата, кнопку расчета результата и кнопку отмены. Сделать так, чтобы кнопку расчета можно было нажать, только если все поля ввода заполнены
9. Разработать функциональные сценарии и реализовать их с помощью одного из средств автоматизации

Рекомендуется для решения задачи использовать Selenium

Лабораторная работа номер 7- Исследование производительности программного продукта с помощью профайлера

В рамках лабораторной работы необходимо произвести профилирование приложения.

Задание на лабораторную работу.

10. Взять задание из лабораторной работы номер 2. Модифицировать полученный код чтобы разработанную функцию можно было выполнять большое количество раз (например, 10000).
11. Подключиться к исполняемому коду профилировщиком (или использовать встроенный в IDE) и проанализировать – какой фрагмент кода занял больше всего процессорного времени.
12. Предложить способы оптимизации

Лабораторная работа номер 8- Стресс тестирование веб-приложения

В рамках лабораторной работы необходимо произвести стресс тестирование Thin интерфейса приложения

Задание на лабораторную работу.

13. Взять приложение, разработанное в рамках лабораторной работы номер 6
14. Разработать тестовый сценарий нагрузочного тестирования (рекомендуется использовать JMeter)
15. Ответить на вопрос – сколько запросов в секунду может обработать приложение при условии, что они идут последовательно.
16. Построить график зависимости времени ответа от количества параллельных запросов (рассматривать логарифмическую шкалу по основанию два, т.е. 1, 2,4,8,16,32 и т.д. запроса)
17. Ответить на вопрос – какое максимальное количество параллельных запросов может обработать приложение без сбоев.

Содержание