Пример ВКР «SPbPU-student-thesis-template». Версия от 5 июня 2021 г.

Министерство науки и высшего образования Российской Федерации
Санкт-Петербургский политехнический университет Петра Великого
Институт прикладной математики и механики

Работа допущена к защите
Руководитель образовательной программы «Прикладная математика и информат
_____ К.Н. Козлов
«_____» _____ 2021 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
РАБОТА БАКАЛАВРА
ЭКСПЕРЕМЕНТЫ С РЕАЛИЗАЦИЕЙ ЯЗЫКА ОХРАНЯЕМЫХ КОМАНД ДЕЙКСТРЫ

по направлению подготовки 01.03.02 Прикладная математика и информатика
Направленность (профиль) 01.03.02_02 Системное программирование

Выполнил
студент гр. 3630102/70201

Руководитель
профессор высшей школы прикладной математики и вычислительной физики,
доктор технических наук, старший научный сотрудник[1]

Консультант[2]
доктор технических наук, старший научный сотрудник

Консультант
по нормоконтролю[3]

Санкт-Петербург
2021

[1]Должность указывают сокращенно, учёную степень и звание — при наличии, а подразделения — аббревиатурами. «СПбПУ» и аббревиатуры институтов не добавляют.

[2]Оформляется по решению руководителя ОП или подразделения. Только 1 категория: «Консультант». В исключительных случаях можно указать «Научный консультант» (должен иметь степень). Без печати и заверения подписи.

[3]Обязателен, из числа ППС по решению руководителя ОП или подразделения. Должность и степень не указываются. Сведения помещаются в последнюю строчку по порядку. Рецензенты не указываются.

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

Институт прикладной математики и механики

УТВЕРЖДАЮ
Руководитель образовательной программы
«Прикладная математика и информатика»
_____ К.Н. Козлов
«_____» _____ 2021г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы студенту

Соломатину Макару Александровичу гр. 3630102/70201

1. Тема работы: Эксперементы с реализацией языка охраняемых команд Дейкстры.
2. Срок сдачи студентом законченной работы: июнь 2021.
3. Исходные данные по работе:
   - Синтаксическое описания языка охраняемых команд
   - Денотационная семантика языка охраняемых команд
   - Теоретические примеры программ

   Инструментальные средства:
   - Язык программирования Python
   - Генератор лексических и синтаксических анализаторов ANTLR4
   - Библиотеки python-antlr4 и sympy

   Ключевые источники литературы:
   - Дейкстра Э. Дисциплина программирования, 1978.
   - Лавров С.С. Программирование. Математические основы, средства, теория.
   - Карпов Ю.Г. Model Checking. Верификация параллельных и распределенных программных систем, 2009.
4. Содержание работы (перечень подлежащих разработке вопросов):
   4.1. Постановка задачи
   4.2. Обзор литературы по теме ВКР

4.3. Исследование программных продуктов
4.4. Разработка интерпретатора и анализатора
4.5. Эксперементы с программами на реализованном языке
4.6. Выводы

5. Дата выдачи задания: 20.12.2020.

Руководитель ВКР _____Ф.А. Новиков

Задание принял к исполнению 20.12.2020

Студент _____М.А. Соломатин

РЕФЕРАТ

На 16 с., 0 рисунков, 1 таблицу, 2 приложения

КЛЮЧЕВЫЕ СЛОВА: РАЗРАБОТКА ИНТЕРПРЕТАТОРА, ЯЗЫК ОХРАНЯЕМЫХ КОМАНД, ФОРМАЛЬНАЯ ВЕРИФИКАЦИЯ ПРОГРАММ, ПОШАГОВОЕ УТОЧНЕНИЕ.[4]

Тема выпускной квалификационной работы: «Эксперементы с реализацией языка охраняемых команд Дейкстры»[5].

В данной работе изложена сущность подхода к созданию динамического информационного портала на основе использования открытых технологий Apache, MySQL и PHP. Даны общие понятия и классификация IT-систем такого класса. Проведен анализ систем-прототипов. Изучена технология создания указанного класса информационных систем. Разработана конкретная программная реализация динамического информационного портала на примере портала выбранной тематики...[6]

В данной работе изложена сущность подхода к созданию динамического информационного портала на основе использования открытых технологий Apache, MySQL и PHP. Даны общие понятия и классификация IT-систем такого класса. Проведен анализ систем-прототипов. Изучена технология создания указанного класса информационных систем. Разработана конкретная программная реализация динамического информационного портала на примере портала выбранной тематики...

ABSTRACT

16 pages, 0 figures, 1 tables, 2 appendices

[4]Всего **слов**: от 3 до 15. Всего **слов и словосочетаний**: от 3 до 5. Оформляются в именительном падеже множественного числа (или в единственном числе, если нет другой формы), оформленных по правилам русского языка. *Внимание! Размещение сноски после точки является примером как запрещено оформлять сноски.*

[5]Реферат **должен содержать**: предмет, тему, цель ВКР; метод или методологию проведения ВКР: результаты ВКР: область применения результатов ВКР; выводы.

[6]ОТ 1000 ДО 1500 печатных знаков (ГОСТ Р 7.0.99-2018 СИБИД) на русский или английский текст. Текст реферата повторён дважды на русском и английском языке для демонстрации подхода к нумерации страниц.

KEYWORDS: INTERPRETER DEVELOPMENT, GUARDED COMMAND LANGUAGE, PROGRAM FORMAL VERIFICATION, STEPWISE REFINEMENT.

The subject of the graduate qualification work is «Experiments with the implementation of the Dijkstra's guarded command language».

In the given work the essence of the approach to creation of a dynamic information portal on the basis of use of open technologies Apache, MySQL and PHP is stated. The general concepts and classification of IT-systems of such class are given. The analysis of systems-prototypes is lead. The technology of creation of the specified class of information systems is investigated. Concrete program realization of a dynamic information portal on an example of a portal of the chosen subjects is developed...

In the given work the essence of the approach to creation of a dynamic information portal on the basis of use of open technologies Apache, MySQL and PHP is stated. The general concepts and classification of IT-systems of such class are given. The analysis of systems-prototypes is lead. The technology of creation of the specified class of information systems is investigated. Concrete program realization of a dynamic information portal on an example of a portal of the chosen subjects is developed...

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В мире современного программирования все более актуальными становятся методы формальной верификации программ. Однако этот процесс является крайне трудоемким, и его стоимость настолько высока, что он применяется лишь в критических местах программного обеспечения, в котором цена ошибки слишком велика: ракетостроение, военная оборона, медицина и т.п. Поэтому, и по некоторым иным причинам, часто прибегают к тестированию программ, которое, как известно, доказывает корректность программ только в некоторых частных случаях. Подобная сложность обусловлена, по-видимому, незрелостью разработанного аппарата для описания программных моделей и соответствующей семантики программ. Э. Дейкстра в своей работе (цитата) рассматривает денотационный способ формализации семантики программ. Он предлагает теоретический язык программирования, доказывает несколько важных утверждений про его семантику и снабжает его множеством примеров. Однако практической реализации в виде интерпретатора языка, и вместе с ней – инструментария для анализа семантики программы, не существует или не было опубликовано. Подобная реализация позволила бы строить недетерминированные алгоритмы и проверять их корректность полуавтоматически, а в некоторых случаях и вовсе без участия программиста.

Формальную верификацию готовых программ, как и предварительное построение корректных программ может быть сделано и для других, распространенных языков программирования, таких как Си. Однако они обладают значительно более сложной семантикой операторов и сложность применения формальных методов значительно выше.

Объектом исследования работы является разработка интерпретатора недетерминированного языка, в основе которого лежит понятие охраняемой команды, а также статический анализ кода программы и вывод ее семантики. Неотъемлимой частью исследования является применение разработанного инструментария к реальным примерам, а также разработка алгоритмов методами, предложенным Э. Дейкстрой.

Предметом исследования работы является создание интерпретируемого недетерминированного языка охраняемых команд Дейкстры. Язык был предложен Э.Дейкстрой с тем, чтобы строить программы, являющиеся корректными по построению.

Автоматический синтез корректных программ не является предметом исследования работы, однако статический анализ программ, полуавтоматически выводящий их семантику, может служить полезным и удобным инструментом для формальной верификации.

Целью исследований является разработка синтаксиса языка охраняемых команд и интерпретатора для программ, построенных на этом языке.

Для достижения этой цели необходимо решить следующие задачи:

A. описать язык формально – задать синтаксис, определить основные операторы языка
B. разработать интерпретатор программ на языке охраняемых команд, исполняющий ее и выводящий результат
C. разработать программу-анализатор, позволяющей по исходному тексту программы выводить предусловие, если постусловие задано пользователем
D. исследовать приведенные в (цитата) примеры, сравненить методы прямого и обратного преобразования предикатов

Теоретической основой выпускной квалификационной работы послужили исследования Э. Дейкстры в своем труде «Дисциплина программирования», дополненные результатами, полученными С.С. Лавровым в книге «Математические основы, средства, теория». Практическая часть работы выполнялась на основании примеров, предложенных Э. Дейкстрой, и некоторых других алгоритмах.

ГЛАВА 1. ДЕНОТАЦИОННАЯ СЕМАНТИКА И ОХРАНЯЕМЫЕ КОМАНДЫ

В этой главе рассматривается денотационный способ задания семантики языка программирования в терминах преобразования предикатов, а также описывается семантика операторов языка охраняемых команд Дейкстры.

1.1. Семантика языка программирования

Построение семантики языка программирования – формальное описание смысла, придаваемого его элементам – выражениям, предложениям, операторам. На настоящий момент известны следующие способы определения семантики языка программирования:

A. Операционная семантика
B. Аксиоматическая, или деривационная семантика
C. Денотационная семантика

В основе денотационной семантики является сопоставления синтаксическим единицам языка математических объектов – множеств и отоношений. Денотационная семантика оператора описывает связь между двумя состояниями:

– *начальным* – состоянием программы перед выполнением оператора
– *конечным* – состоянием программы после выполнения оператора

1.2. Множество состояний

Любая программа, предназначенная для выполнения на вычислительном устройстве, оперирует с определенным множеством ячеек памяти – записывает в них и считывает из них. Назовем это множество ячеек памяти, или как принято называть переменных, состоянием программы. Нетрудно заметить, что множеством всех возможных состояний программы является прямое произведение областей значений каждой переменной. Программа может изменять состояние программы как изменением значений имеющихся переменных, так и добавлением новых переменных. Экспоненциальный рост числа состояний программы с количеством переменных, также известный как проклятие размерности, дает основание считать множество возможных состояний практически несчетным.

Для описания подмножеств множества состояний используется исчисление предикатов первого порядка, в которых формулы содержат свободные термы – переменные состояния. Таким образом каждому предикату соответствует некоторое множество состояний, в интерпретации которых формула истинна.

1.3. Преобразование предикатов

Преобразователь предикатов – это функциональное отношение на множестве предикатов. В рамках денотационной семантики, каждой программе S языка охраняемых команд сопоставляется преобразователь предикатов p, который по предикату R определяет предикат $p(S, R)$.

1.4. Предусловие и постусловие

Рассмотрим некоторую программу S языка охраняемых команд. Предикаты R и $p(S, R)$ называются *постусловием* и *предусловием* программы S соответственно, если программа S, запущенная в состоянии, удовлетворяющем $p(S,R)$, обязательно завершится и после своего выполнения останется в состоянии, удовлетворяющем предикату R.

Предикат $wp(S, R)$ называется *слабейшим предусловием*, если он характеризует множество *всех* состояний, запуск программы S из которых обязательно приведет к завершению программы и оставит ее в состоянии, удовлетворяющем предикату R.

Предикат $wlp(S, R)$ называется *слабейшим свободным предусловием*, если он характеризует множество *всех* состояний, запуск программы S из которых, если он приведет к завершению программы, оставит ее в состоянии, удовлетворяющем предикату R.

1.5. Выводы

Текст выводов по главе 1.

Кроме названия параграфа «выводы» можно использовать (единообразно по всем главам) следующие подходы к именованию последних разделов с результатами по главам:

– «выводы по главе N», где N — номер соответствующей главы;

– «резюме»;

– «резюме по главе N», где N — номер соответствующей главы.

Параграф с изложением выводов по главе *является обязательным*.

ГЛАВА 2. ОПИСАНИЕ ЯЗЫКА

В этой главе изложено синтаксическое описание языка охраняемых команд наряду с его семантикой. Для этого поэтапно вводятся используемые в нем операторы, понятие охраняемой команды, а в приложении находятся полное синтаксическое описание языка с помощью расширенной формы Бэкуса-Наура (РБНФ) и синтаксических диаграмм Вирта.

2.1. Оператор «skip»

Оператор *skip* ничего не делает и оставляет программму в том же состоянии, какое было и перед ее выполнением. Поэтому

$$wp(skip, R) = R \tag{2.1}$$

2.2. Оператор «reject»

Оператор *reject* не может завершиться ни в каком начальном состоянии. Поэтому

$$wp(reject, R) = False \tag{2.2}$$

2.3. Оператор присваивания

Оператор присваивания связывает значение некоторого выражения E с переменной x:

```
x := E
```

Пусть R – постусловие, а предикат $R_{E \to x}$ получен из R подстановкой выражения E вместо всех вхождений переменной x. Тогда

$$wp(x := E, R) = R_{E \to x} \tag{2.3}$$

2.3.1. Множественное присваивание

Вполне естественным расширением оператора присваивания является оператор множественного присваивания:

```
x_1, x_2 \ldots, x_n := E_1, E_2 \ldots, E_n
```

Он вычисляет значения выражений E

2.3.2. Литералы и типы выражений

2.4. Охраняемая команда

2.5. Условный оператор

2.6. Оператор цикла

2.7. Макроподстановки

2.8. Выводы

Текст заключения ко второй главе. Пример ссылок [**Article**; **Book**; **Booklet**; **Conference**; **Inbook**; **Incollection**; **Manual**; **Mastersthesis**; **Misc**; **Phdthesis**; **Proceedings**; **Techreport**; **Unpublished**; **badiou:briefings**], а также ссылок с указанием страниц, на котором отображены те или иные текстово-графические объекты [**Naidenova2017**] или в виде мультицитаты на несколько источников [**Naidenova2017**; **Ganter1999**]. Часть библиографических записей носит иллюстративный характер и не имеет отношения к реальной литературе.

Короткое имя каждого библиографического источника содержится в специальном файле `my_biblio.bib`, расположенном в папке `my_folder`. Там же находятся исходные данные, которые с помощью программы `Biber` и стилевого файла `Biblatex-GOST` [**ctan-biblatex-gost**] приведены в списке использованных источников согласно ГОСТ 7.0.5-2008. Многообразные реальные примеры исходных библиографических данных можно посмотреть по ссылке [**ctan-biblatex-gost-examples**].

Как правило, ВКР должна состоять из четырех глав. Оставшиеся главы можно создать по образцу первых двух и подключить с помощью команды `\input`

к исходному коду ВКР. Далее в приложении 1 приведены краткие инструкции запуска исходного кода ВКР [**latex-miktex**; **latex-texstudio**].

В приложении 2 приведено подключение некоторых текстово-графических объектов. Они оформляются по приведенным ранее правилам. В качестве номера структурного элемента вместо номера главы используется «П» с номером главы. Текстово-графические объекты из приложений не учитываются в реферате.

ГЛАВА 3. РАЗРАБОТКА ИНТЕРПРЕТАТОРА И АНАЛИЗАТОРА

Хорошим стилем является наличие введения к главе. Во введении может быть описана цель написания главы, а также приведена краткая структура главы.

3.1. Название параграфа

3.2. Название параграфа

3.3. Выводы

Текст выводов по главе 3.

ГЛАВА 4. ЭКСПЕРИМЕНТЫ С ЯЗЫКОМ ОХРАНЯЕМЫХ КОМАНД

4.1. Выводы

Текст выводов по главе 4.

ЗАКЛЮЧЕНИЕ

Заключение (2 – 5 страниц) обязательно содержит выводы по теме работы, *конкретные предложения и рекомендации* по исследуемым вопросам. Количество общих выводов должно вытекать из количества задач, сформулированных во введении выпускной квалификационной работы.

Предложения и рекомендации должны быть органически увязаны с выводами и направлены на улучшение функционирования исследуемого объекта. При разработке предложений и рекомендаций обращается внимание на их обоснованность, реальность и практическую приемлемость.

Заключение не должно содержать новой информации, положений, выводов и т. д., которые до этого не рассматривались в выпускной квалификационной работе. Рекомендуется писать заключение в виде тезисов.

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

DOI Digital Object Identifier.

WoS Web of Science.

ВКР Выпускная квалификационная работа.

ТГ-объект Текстово-графический объект.

СЛОВАРЬ ТЕРМИНОВ

TeX — язык вёрстки текста и издательская система, разработанные Дональдом Кнутом.

LaTeX — язык вёрстки текста и издательская система, разработанные Лэсли Лампортом как надстройка над TeX.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

Приложение 1

Краткие инструкции по настройке издательской системы LaTeX

В SPbPU-BCI-template *автоматически выставляются необходимые настройки и в исходном тексте шаблона приведены примеры оформления текстово-графических объектов*, поэтому авторам достаточно заполнить имеющийся шаблон текстом главы (статьи), не вдаваясь в детали оформления, описанные далее. Возможный «быстрый старт» оформления главы (статьи) под Windows следующий[П1.1]:

A. Установка полной версии MikTeX [**latex-miktex**]. В процессе установки лучше выставить параметр доустановки пакетов «на лету».
B. Установка TexStudio [**latex-texstudio**].
C. Запуск TexStudio и компиляция `my_chapter.tex` с помощью команды «Build&View» (например, с помощью двойной зелёной стрелки в верхней панели). *Иногда, для достижения нужного результата необходимо несколько раз скомпилировать документ.*
D. В случае, если не отобразилась библиография, можно
   - воспользоваться командой Tools → Commands → Biber, затем запустив Build&View;
   - настроить автоматическое включение библиографии в настройках Options → Configure TexStudio → Build → Build&View (оставить по умолчанию, если сборка происходит слишком долго): `txs:///pdflatex | txs:///biber | txs:///pdflatex | txs:///pdflatex | txs:///view-pdf`.

В случае возникновения ошибок, попробуйте скомпилировать документ до последних действий или внимательно ознакомьтесь с описанием проблемы в log-файле. Бывает полезным переход (по подсказке TexStudio) в нужную строку в pdf-файле или запрос с текстом ошибке в поисковиках. Наиболее вероятной проблемой при первой компиляции может быть отсутствие какого-либо установленного пакета LaTeX.

В случае корректной работы настройки «установка на лету» все дополнительные пакеты будут скачиваться и устанавливаться в автоматическом режиме. Если доустановка пакетов осуществляется медленно (несколько пакетов за один запуск

[П1.1] Внимание! Пример оформления подстрочной ссылки (сноски).

компилятора), то можно попробовать установить их в ручном режиме следующим образом:

1. Запустите программу: меню → все программы → MikTeX → Maintenance (Admin) → MiKTeX Package Manager (Admin).
2. Пользуясь поиском, убедитесь, что нужный пакет присутствует, но не установлен (если пакет отсутствует воспользуйтесь сначала MiKTeX Update (Admin)).
3. Выделив строку с пакетом (возможно выбрать несколько или вообще все неустановленные пакеты), выполните установку Tools → Install или с помощью контекстного меню.
4. После завершения установки запустите программу MiKTeX Settings (Admin).
5. Обновите базу данных имен файлов Refresh FNDB.

Для проверки текста статьи на русском языке полезно также воспользоваться настройками Options → Configure TexStudio → Language Checking → Default Language. Если русский язык «ru_RU» не будет доступен в меню выбора, то необходимо вначале выполнить Import Dictionary, скачав из интернета любой русскоязычный словарь.

Далее приведены формулы (П1.2), (П1.1), рис.П1.2, рис.П1.1, табл.П1.2, табл.П1.1.

$$\pi \approx 3{,}141. \tag{П1.1}$$

Рис.П1.1. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П1.1

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2

П1.1. Параграф приложения

П1.1.1. Название подпараграфа

Название подпараграфа оформляется с помощью команды `\subsection{...}`. Использование подподпараграфов в основной части крайне не рекомендуется.

П1.1.1.1. Название подподпараграфа

$$\pi \approx 3{,}141. \tag{П1.2}$$

Рис.П1.2. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П1.2

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2

Приложение 2

Некоторые дополнительные примеры

В приложении[П2.1] приведены формулы (П2.2), (П2.1), рис.П2.2, рис.П2.1, табл.П2.2, табл.П2.1

$$\pi \approx 3{,}141. \qquad \text{(П2.1)}$$

Рис.П2.1. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П2.1

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2

[П2.1] Внимание! Пример оформления подстрочной ссылки (сноски).

П2.1. Подраздел приложения

$$\pi \approx 3{,}141. \qquad \text{(П2.2)}$$

Рис.П2.2. Вид на гидробашню СПбПУ [**spbpu-gallery**]

Таблица П2.2

Представление данных для сквозного примера по ВКР [**Peskov2004**]

G	m_1	m_2	m_3	m_4	K
g_1	0	1	1	0	1
g_2	1	2	0	1	1
g_3	0	1	0	1	1
g_4	1	2	1	0	2
g_5	1	1	0	1	2
g_6	1	1	1	2	2