Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей Кафедра
электронных вычислительных машин
Дисциплина: Операционные системы и системное программирование

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовому проектуна
тему

ПРОГРАММА-СТОРОЖ НЕПРАВИЛЬНОЙ РАСКЛАДКИ РУССКИЙ/АНГЛИЙСКИЙ С ЗАМЕНОЙ ВВЕДЕННОГО ФРАГМЕНТА И ПЕРЕКЛЮЧЕНИЕМ РАСКЛАДКИ

БГУИР КП 1-40 02 01 128 ПЗ

Студент: гр. 230501 Лазовский И. А.

Руководитель: старший преподаватель
каф. ЭВМ Поденок Л. П.

Минск 2024

Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»

Военный факультет

УТВЕРЖДАЮ
Заведующий кафедрой
_____
(подпись)
_____ 2024 г.

ЗАДАНИЕ
по курсовому проектированию

Студенту *Лазовскому Илье Александровичу*

1. Тема проекта: Программа-сторож неправильной раскладки русский/английский с заменой введенного фрагмента и переключением раскладки

2. Срок сдачи студентом законченного проекта: *15 мая 2024 г.*

3. Исходные данные к проекту: *язык программирования С*

4. Содержание расчетно-пояснительной записки (перечень вопросов, которые подлежат разработке):

Введение. 1. Обзор методов и алгоритмов решения поставленной задачи. 2. Обоснование выбранных методов и алгоритмов. 3. Описание программы для программиста. 4. Описание алгоритмов решения задачи. 5. Руководство пользователя. Заключение. Список использованных источников. Приложения.

5. Перечень графического материала (с точными обозначениями обязательных чертежей и графиков):

1. Схема алгоритма работы функции

2. Ведомость документов

6. Консультант по проекту (с обозначением разделов проекта) *Поденок Л. П.*

7. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоемкости отдельных этапов): *раздел 1 к 01.03. – 15 %;*

раздел 2, 3 к 01.04. – 50 %;

раздел 4, 5 к 01.05. – 80 %;

оформление пояснительной записки и графического материала к 15.05.2024 – 100 %

Защита курсового проекта с 29.05 по 09.06.

РУКОВОДИТЕЛЬ _____ *Л. П. Поденок*
(подпись)

Задание принял к исполнению _____ *И.А. Лазовский*
(дата и подпись студента)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Данный курсовой проект посвящен разработке программы-сторожа неправильной раскладки с заменой введенного фрагмента и переключением раскладки. Программа идентифицирует часть текста, набранную на языке, не соответствующем выбранной раскладке, обрабатывает обнаруженный фрагмент, переключается на нужную раскладку и в результате выводит нужный текст.

Необходимость и актуальность разработки подтверждается тем, что пользователи компьютеров в независимости от профессиональных навыков, сферыдеятельности и требуемого функционала сталкиваются с проблемой написания текста в результате использования раскладки другого языка. Вследствие того, что на современных операционных системах отсутствует поддержка автоматического переключения раскладки, пользователям в случае набора неправильного текста, необходимо удалить напечатанную часть и ввести ее заново. Этот способ решения описанной проблемы не является эффективным в силу того, что временные затраты на простейшую задачу увеличиваются более, чем в два раза.

Следует отметить, что наиболее распространена проблема включения неверной раскладки среди людей, владеющих несколькими языками, для которых смена раскладки имеет место не только для получения доступа к определенным символам, но и для полноценного использования другого языка.

Как уже было сказано выше, выделенная проблема не нова, вследствие чего на рынке представлены примеры программного обеспечения, реализующие автоматическое определение неверно набранного текста и переключение раскладки на соответствующую используемому языку. Так, поисковая система Google способна распознавать запросы даже в том случае, если они введены, используя несвойственный требуемому язык алфавит. Среди известных аналогов на рынке следует выделить Punto Switcher – программу для автоматического переключения между различными раскладками клавиатуры в операционных системах семейства Microsoft Windows и macOS. Схожим набором функций обладает программа LangOver.

Однако, несмотря на то, что данный курсовой проект не преследует цель создать конкурентоспособный аналог существующим на рынке продуктам, его тема была выбрана главным образом для обобщения и систематизации знаний, полученных в ходе изучения принципов программирования на базе операционной системы `Linux`, углубления навыков написания кода на языке Си, а также получения опыта взаимодействия с устройствами ввода и создания пользовательских приложений с удобным и понятным интерфейсом.

1 ПОСТАНОВКА ЗАДАЧИ

По исходным данным необходимо разработать: программу, проводящую анализ вводимых символов, и, если их сочетание оказывается нетипичным для языка, на котором осуществляется печать текста, переключающую раскладку; стирающую набранное; и повторно вводящую уже исправленный набор символов.

Ключевые возможности, которые требуется реализовать:

- автоматическое переключение раскладки клавиатуры на нужный из двух заданных языков: русского или английского;
- исправление явно указанного набранного фрагмента;
- проверка допустимости сочетания букв для текущего языка в введенной последовательности символов, исправление при обнаружении языковых аномалий.

Для разработки данного программного обеспечения используется язык программирования C, среда разработки Kate.

2 ОБЗОР МЕТОДОВ И АЛГОРИТМОВ РЕШЕНИЯ ПОСТАВЛЕННОЙ ЗАДАЧИ

Разработка программы-сторожа неправильной раскладки включает несколько основных этапов:

– идентификация фрагмента текста, набранного с использованием раскладки, отличной от языка, на котором предполагалось набирать текст;
– замена обнаруженного фрагмента на верно написанный вариант;
– переключение раскладки для предотвращения дальнейших ошибок.

Прежде всего, для отслеживания набираемого пользователем при помощи клавиатуры текста в процессе работы с операционной системой Linux можно использовать стандартные функции ввода/вывода для языка программирования Си, такие как scanf(), fgets(), getline().

Однако, необходимо учесть, что текстовая информация преимущественно отображается в кодировке UTF-8, согласно которой символы, соответствующие кириллице, занимают по 2 байта в отличие от латинских символов,на которые отводится по одному байту. Поэтому для корректной работы программы потребуется использовать библиотеку для работы с так называемыми

«широкими словами» – «wchar.h». Она позволяет обрабатывать символы, занимающие в памяти два байта.

Важно учесть, что данный способ предполагает отслеживание правильности набираемого текста только в специальных средах разработки, таких как Microsoft Visual Studio, Xcode, Clion и прочих, а также в терминале. Длярасширения сферы применения программного обеспечения на все активные окна программ, в которых производится ввод текста, может быть применена библиотека «Xlib.h», которая обеспечивает функционал для отслеживания событий нажатия клавиш и, как следствие, считывания пользовательского текста.

Пользователи, владеющие несколькими языка и пользующиеся ими для выполнения повседневных задач, мог совершать разнообразные ошибки в процессе набора текста, не привязанного к одному языку, или при переходе с одного языка на другой. Набранные слова могут как полностью не соответствовать включенной раскладке, так и частично, к примеру, при попытке исправить орфографическую ошибку путем замены лишь одного символа. Следовательно, необходимо реализовать проверку на то, существует ли напечатанноепользователем слово или потенциально могла произойти опечатка. Такой механизм может быть разработан посредством нескольких методов, некоторые из которых будут описаны далее.

В качестве одного из самых очевидных решений может выступать лексикографический анализ слова – процесс определения различных морфологических характеристик слова на основе его лексических компонентов. Целесообразность его применения можно обосновать тем, что в каждом языке присутствует набор правил и шаблонов, по которым происходит словообразование:

допустим, в русском языке слово не может полностью состоять из согласных букв в отличие от хорватского. Основываясь на подобных языковых особенностях, можно сделать вывод касательно того, напечатал ли пользователь

задуманное слово или же не переключился на нужную раскладку клавиатуры. Тем не менее, описанный метод с точки зрения программной реализации неэффективный и подразумевает учет большого количества абстрактных условий, для которых будет затруднительно вывести общие формулировки.

Еще одним вариантом воплощения поставленной задачи может стать использование сторонних библиотек, таких как Hunspell или WN. Первая библиотека может встраиваться в проект для проверки орфографии на разных языках. К числу ее преимуществ можно отнести быстрое осуществление запрашиваемой процедуры определения правильности введенного текста с точки зрения орфографии и небольшой объем требуемого кода.

Вторая библиотека предоставляет набор функций для работы с лексической базой данных WordNet, которая в рамках рассматриваемой темы может быть применена в качестве тезауруса. Общим недостатком этого подхода является факт того, что поставляемый продукт не будет работать на компьютерепользователя в случае, если на нем не предустановлена какая-либо из упомянутых библиотек.

На ряду с названными способами, которые, так или иначе, осуществляют локальное решение проблемы, возможно также использовать открытое API, поставляемое каким-либо веб-сервисом, предоставляющим доступ к словарюанглийского или русского языка. Для этого необходима предварительная регистрация на сайте веб-сервиса с целью получения API-ключа, который можетбыть использован в программном коде для получения доступа к требуемому словарю. После чего отправляется GET-запрос на сервис и ожидается ответ в формате JSON. Недостатками данного метода являются следующие аспекты:

– необходимость наличия подключения к сети Интернет;

в случае отсутствия библиотеки «json-c» требуется ее установка. Одним из простейших средств может выступать использование локального словаря, в данном случае английского и русского языка, в виде соответствующих файлов, поставляемых в составе готового проекта. Когда возникаетпотребность проверить то или иное слово, производится поиск данного словав файле при помощи стандартной функции для чтения информации из файла fgets().

Преимущественной характеристикой вышеизложенного подхода является простота реализации и, в отличие от предыдущего метода, универсальность, проявляющаяся в том, что отсутствует необходимость проверки наличия тех или иных компонентов на компьютере пользователя. Однако отрицательная сторона проявляется во временных затратах на прохождение по словарю и поиск нужного слова, которого может и не существовать. Тем не менее, данную проблему можно решить за счет ускоренных методов поиска подстроки в файле.

Переходя ко второму этапу, следует отметить, что основным методом реализации требуемой функции можно назвать использовании таблице транслитерации в виде двух массивов одинаковой длины, один из которых соответствует набору клавиш английской раскладки, другой – русской. Так, например, если требуется заменить букву «q» на «й», можно произвести поиск по массиву латинских букв и, найдя нужный символ (на 0-ой позиции), заменитьего на символ кириллицы, расположенный на том же месте во втором массиве. Описанный

алгоритм также может нуждаться в оптимизации в целях увеличения скорости работы.

Наконец, последний этап заключается в переключении раскладки на ту, которая должна отвечать потребностям пользователя в данный момент времени. В первую очередь, ее тип определяется истинным языком обнаруженного фрагмента. Далее, что касается средств для осуществления самого процесса выбора нужного режима.

Для этого можно, например, задействовать системный вызов system(), которому отправить аргументы для установки требуемого языка. Однако этотспособ может быть менее гибким и функциональным, как нижеперечисленные.

Помимо упомянутого средства, можно применить специализированные библиотеки, такие как уже названная «Xlib.h» и «libxkbcommon». Первый вариант, в целом, располагает широкими возможностями для работы с клавиатурой, как, впрочем, и второй. В контексте решаемой задачи из второй библиотеки понадобится функция xkb_keymap_layout_by_name(), которая и осуществит переключение на нужную раскладку.

Таким образом, выбор методов и средств для реализации, поставленной задачи в рамках данного курсового проекта должен основываться на соответствии следующим характеристикам:

- максимальная производительность, насколько это возможно при сочетании тех или иных подходов;
- удобный интерфейс и интуитивно понятное и доступное использование с точки зрения среднестатистического пользователя;
- минимальное количество занимаемого дискового пространства;
- сочетаемость с пользовательскими настройками операционной системы Linux;
- рациональное использование программных средств и функций.

3 ОБОСНОВАНИЕ ВЫБРАННЫХ МЕТОДОВ И АЛГОРИТМОВ

Исходя из подробного описания и анализа возможных методов и алгоритмов решения поставленной в курсовом проекте задачи, были выбраны следующие программные средства.

Для обнаружения фрагмента текста, набранного с использованием раскладки клавиатуры, не соответствующей языку, который предполагался пользователем, в программе любые процессы взаимодействия с введенным текстом осуществляются как с набором «широких слов».

Несмотря на то, что проблемы могут возникать только в случае обработки символов кириллицы, которые, как упоминалось выше, кодируются посредством двух байт, нецелесообразно и неэффективно разграничивать используемые переменные и функции для работы с данными типа *char (в случае с английскими словами) и *wchar_t (кириллические символы). Такой подход позволяет, например, грамотно разбирать посимвольно слово, напечатанное на русском языке, когда необходимо заменить те или иные (а возможно, все) его буквы на соответствующие иной раскладке. Потому как, если бы для этого использовался стандартный тип данных в языке программирования Си – char, то, предположим, русское слово из четырех букв заменилось бы на псевдоанглийское из восьми.

Что касается проверки набранного слова на существование в словаре английского или русского языка, были произведены попытки реализации всех приведенных методов в целях выделения из числа прочих того, которыйнаиболее четко будет отвечать поставленным задачам и требованиям к ним. Каждый из них показал свои преимущества и недостатки, которые в разной степени подтвердили теоретические обоснования их вероятности, приведенные в предыдущем разделе.

В итоге, было принято решение остановиться на варианте проверки очередного ошибочно введенного слова при помощи локальных словарей английского и русского языка. Хотя этот способ и не самый эффективный по показателю среднего времени работы, но используемые алгоритмы можно оптимизировать. Главным критерием в пользу его выбора послужил тот факт, что дляего внедрения в программный код, не потребуется со стороны пользователя установка сторонних библиотек или подключение интернета.

Последующая замена слов, которые не были найдены в словарях, производится указанным выше способом, предполагающим объявление двух массивов и инициализацию их последовательностями символом, присущими обеимуказанным раскладкам клавиатуры, при условии, что порядок повторяет тот, согласно которому расположены клавиши используемого устройства ввода. Благодаря малому количеству хранимых данных достигается скорость операции, не создающая ощутимую задержку в исполнении кода программы и обеспечивающая требуемую точность.

Для переключения текущей раскладки выбран метод, заключающийся в применении библиотеки функций для взаимодействия с X-сервером, которая, по сути, является основным графическим интерфейсом пользователя во многих

Юникс-подобных операционных системах, к числу которых также относится и Linux. В частности удобным средством для решения поставленной задачи может быть функция XkbLockModifiers().

В результате изучения предоставленного набора алгоритмов и методов были отобраны те решения, которые обеспечат точность, простоту и доступность программного обеспечения.

4 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПРОГРАММИСТА

На данном этапе разработки общий пакет программы включает в себя четыре файла:

– один исходный (LayoutController.c);

– текстовый (english.txt);

– текстовый (russian.txt);

– Makefile.

После запуска программа ожидает ввода пользователем текста с клавиатуры. В начале процесса полученный фрагмент дробится на отдельные токены слова, каждое из которых отправляется в качестве аргумента в функцию checkSpelling(), после чего проверяется структура слова на однородность. В этой же функции происходит вызов функции checkWord(), которая при помощи локальных словарей, хранящихся в файлах english.txt и russian.txt. В зависимости от значения, возвращенного функций checkWord(), выставляется флаг, сигнализирующий о необходимости замены фрагмента текста.

В случае, если флаг принимает истинное значение, происходит вызов функции replaceFragment(), которая, ориентируясь по второму аргументу – исходному языку раскладки – производит посимвольную замену букв на правильные.

Далее, исходя из того, какую раскладку клавиатуры предполагалось использовать, посредством вызова функции switchLayout() реализуется требуемое переключение.

Структура текстовых файлов представляют собой фактически список слов, каждое из которых находится на новой строке с той разницей, что в словаре английского языка они идут сплошным потоком, а в русском словаре возможен вариант упрощенного поиска за счет наличия строк, состоящих из одного символа – буквы, с которой будут начинаться все последующие слова, или слога. Структура данных в файле приведена в таблице 4.1.

Таблица 4.1 – Структура данных файла russian.txt

б
ба
баальбек

Для компиляции проекта используется файл Makefile, в котором задаются правила для утилиты make, необходимые для корректного построения исполняемого файла на основании исходного текста программы. В нем прописаны следующие параметры компиляции:

1) -w;
2) -Wall;
3) -Wextra;
4) -std=c11;
5) -pedantic;
6) -lX11.

5 ОПИСАНИЕ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧИ

5.1 Схема алгоритма функции checkWord()

Схема алгоритма функции представлена на чертеже ГУИР.

5.2 Алгоритм по шагам функции checkSpelling()

Шаг 1. Начало.

Шаг 2. Входные данные: wchar_t word – слово, подвергающееся проверке на наличие разнородных символов.

Шаг 3. Вход в цикл for, осуществляющий проход по символам слова. Шаг 4. Если буква принадлежит английскому алфавиту, инкрементируется счетчик латинских символов.

Шаг 5. Иначе инкрементируется счетчик кириллических символов.

Шаг 6. Выход из цикла for по выполнении установленного условия.

Шаг 7. Если счетчик английских букв равен ненулевому значению, в отличие от счетчика русских букв, осуществляется проверка наличия слова в соответствующем словаре при помощи функции checkWord().

Шаг 8. Если возвращаемое значение равно единице, то флаг замены replace сбрасывается и производится выход из функции.

Шаг 9. Иначе флаг замены устанавливается в единицу и вызывается функция replaceFragment() для замены введенного фрагмента.

Шаг 10. Иначе, если счетчик русских букв равен ненулевому значению, в отличие от счетчика английских букв, осуществляется проверка наличия слова в соответствующем словаре при помощи функции checkWord().

Шаг 11. Аналогично шагу 7.

Шаг 12. Аналогично шагу 8.

Шаг 13. Иначе, если оба счетчика не равны нулю, флаг замены выставляется в единицу, последовательно вызывается функция replaceFragment(), заменяющая сначала русские символы на английские.

Шаг 14. Если функция checkWord() возвращает не единицу, вызывается функция replaceFragment(), заменяющая теперь английские символы на русские.

Шаг 15. Вызывается функция checkWord().

Шаг 16. Конец.

5.3 Алгоритм по шагам функции replaceFragment()

Шаг 1. Начало.

Шаг 2. Входные данные: wchar_t word – слово, в котором необходимо произвести замену символов, int lang – флаг исходного языка: если равен единице, замене подлежат английские символы, если сброшен – русские.

Шаг 3. Установка текущей локали системы на en_US.UTF-8.

Шаг 4. Вход в цикл for, осуществляющий проход по символам слова.

Шаг 5. Если значение переменной lang равно единице, вход в цикл for, осуществляющий проход по массиву символов английской раскладки в порядке расположения клавиш на клавиатуре.

Шаг 6. Если на текущей итерации внешнего цикла символ переданного в функцию слова соответствует найденному в массиве, он заменятся на русский символ и производится выход из внутреннего цикла.

Шаг 7. Выход из внутреннего цикла for по выполнении установленного условия.

Шаг 8. Иначе, если значение переменной lang равно нулю, вход в цикл for, осуществляющий проход по массиву символов русской раскладки в порядке расположения клавиш на клавиатуре.

Шаг 9. Если на текущей итерации внешнего цикла символ переданного в функцию слова соответствует найденному в массиве, он заменятся на английский символ и производится выход из внутреннего цикла.

Шаг 10. Выход из внутреннего цикла for по выполнении установленного условия.

Шаг 11. Выход из внешнего цикла for по выполнении установленногоусловия.

Шаг 12. Конец.

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

6.1 Минимальные системные требования

– ОС Linux (базовая, Dual-boot, VM) или Mac OS High Sierra, Mojave,Catalina, Big Sur, Monterey, Ventura;
– ОЗУ: 1 ГБ;
– клавиатура.

6.2 Рекомендуемые системные требования

– ОС Linux (базовая, Dual-boot, VM) или Mac OS Big Sur, Monterey, Ventura;
– ОЗУ: 2 ГБ и более;
– клавиатура, мышь.

6.3 Комплект поставляемого ПО

В комплекте с поставляемым программным обеспечением должны быть:
1) Makefile для компиляции проекта;
2) исходные файлы, содержащие описание методов и функций:
3) LayoutController.c;
4) Текстовые файлы, содержащие словари английского и русского языков:
– english.txt;
– russian.txt.

6.4 Как пользоваться программой

При запуске программы откроется окно с интерфейсом рисунок 6.1.

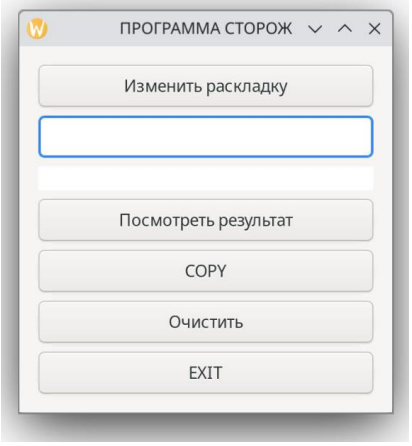

Рисунок 6.1 – Интерфейс программы

Далее вводим текст для работы и проверяем результат, нажав кнопку «Просмотреть результат» рисунок 6.2.

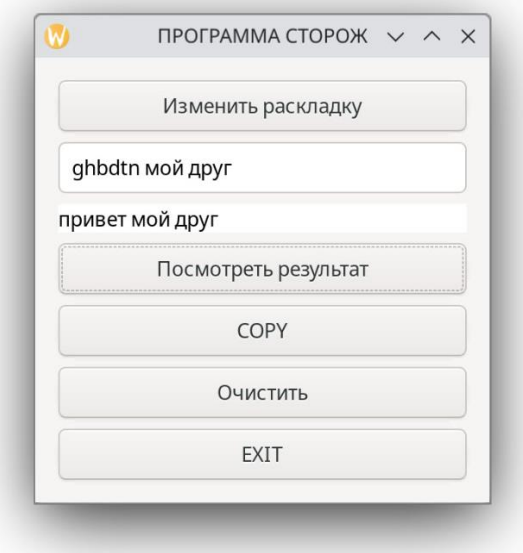

Рисунок 6.2 – Результат работы программы

При нажатии на «Очистить» строка ввода будет очищена.

При нажатии на «Изменить раскладку» раскладка клавиатуры будет изменена.

При нажатии на «COPY» пользователь скопирует результат.

При нажатии на «EXIT» программа закроется.

ЗАКЛЮЧЕНИЕ

Данная курсовая работа посвящена разработке программы-сторожа, предназначенной для облегчения работы с неправильной раскладкой клавиатуры при вводе текста на русском и английском языках. Целью работы было создание эффективного инструмента, способного автоматически определять неправильно введенные символы и заменять их на соответствующие символы в правильной раскладке.

В процессе исследования были изучены основные принципы работы с раскладкой клавиатуры в операционных системах Linux, а также алгоритмы определения и исправления неправильно введенных символов. Была проведена аналитическая работа по анализу существующих программных решений и выбран наиболее подходящий подход для разработки программы-сторожа.

В заключение, Linux – это мощная и гибкая операционная система, которая широко применяется в различных областях, включая программирование на языке C. Она предоставляет пользователям большую степень свободы и контроля над их системой, что делает ее особенно привлекательной для разработчиков. Кроме того, Linux обладает обширным набором инструментов и библиотек для программирования на языке C, что позволяет разработчикам создавать высококачественное программное обеспечение. В целом, использование Linux для программирования на языке C может значительно улучшить процесс разработки и повысить качество конечного продукта.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Реймонд, Э. С. Искусство программирования для Unix / Э. С. Реймонд; пер. с англ. Москва: Вильямс, 2005. – 544 с.: ил.

[2] Стивенс, У. Р. UNIX. Профессиональное программирование / У. Р. Стивенс, С. А. Раго ; пер. А. Киселева. – 2-е изд. – Санкт-Петербург : Символплюс, 2007. – 1040 с. : ил.

[3] Документация Xlib [Электронный ресурс]. – The Open Group. 2002. – Режим доступа: https://www.x.org/releases/X11R7.5/doc/libX11/libX11.html

ПРИЛОЖЕНИЕ А

Схема алгоритма

ПРИЛОЖЕНИЕ Б

Ведомость документов