МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«кубанский государственный университет»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра информационных технологий**

**курсовая работа**

**РАЗРАБОТКА СИСТЕМЫ ОБУЧЕНИЯ «СЛЕПОМУ  
МЕТОДУ ПЕЧАТИ» В ФРЕЙМВОРКЕ QT.**

Работу выполнил А.Э. Айрапетов

(подпись)

Направление подготовки 01.03.02 Прикладная математика и информатика

Направленность (профиль) «Системное программирование и компьютерные технологии» (Математическое и программное обеспечение вычислительных машин)

Научный руководитель

доц., канд. физ.-мат. наук\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ О.В. Гаркуша

(подпись)

Нормоконтролер

ст. преп. А.В. Харченко

(подпись)

Краснодар

2019

**РЕФЕРАТ**

Курсовая работа содержит 35 страниц, 18 изображений. Ключевые слова: слепая печать, клавиатурный тренажёр, обучение, курс, тренировка.

Цель курсовой работы – разработка системы развития навыка «слепого метода набора», а именно – создание клавиатурного тренажёра с режимами обучения и тренировки для оттачивания навыка.

В ходе курсовой работы будет подробно рассмотрена концепция десятипальцевого метода печати, изучены проекты популярных клавиатурных тренажёров, а также создан собственный тренажёр «Акселератор».

В работе реализована программа с удобным интерфейсом, позволяющая пройти обучение слепой печати и впоследствии оттачивать навыки на пользовательских словарях. В приложении прописано предоставление отчёта по средней и текущей скорости при наборе текста, точности набора (или количестве ошибок), а также специфичные функции (например, сохранение самых медленных сочетаний). Под данную программу была написана своя база данных, которая является курсовым проектом Косикова Андрея Витальевича и будет подробно рассмотрена в его работе.

**СОДЕРЖАНИЕ**

Введение 4

1 Концепция десятипальцевого метода набора 6

1.1 Описание метода 6

1.2 Преимущества 7

1.3 Сложности, связанные с обучением 9

2 Клавиатурные тренажёры 12

2.1 Анализ популярных тренажёров 13

2.1.1 «СОЛО на клавиатуре» 13

2.1.2 «Ratatype» 14

2.1.3 «Стамина» 15

2.1.4 «Клавогонки» 16

2.2 Разбор функциональной части 17

2.2.1 Деятельность пользователя 17

2.2.2 Программный функционал 18

3 Разработка тренажёра «Акселератор» 21

3.1 Выбор языка и среды разработки 21

3.2 Семантика процесса 21

3.3 Реализация 22

3.3.1 База данных 22

3.3.2 Интерфейс 24

3.3.3 Функционал 29

Заключение 33

Список использованных источников 35

**ВВЕДЕНИЕ**

Современный мир, изобилующий гаджетами и характеризуемый наличием компьютеров у большинства людей, требует, в частности, регулярного взаимодействия с клавиатурой. В зависимости от рода деятельности, частота работы с машинописью может различаться, однако во всех случаях прослеживается прямая пропорциональность между КПД и скоростью печати.

Так, на перевод в печатные символы 500-символьного текста у рядового пользователя клавиатуры уходит около 6 минут. Если печатать текст без ошибок, то этот процесс может быть не таким изматывающим, но, как показывает практика, ошибки обычно составляют 10% от общего текста, не говоря уже о том, что тексты для печати зачастую бывают во много раз больше. Как результат, простая процедура печати становится для обывателя довольно неприятной рутиной. Более того, постоянный перевод взгляда с текста на клавиатуру и с клавиатуры на монитор, как известно, крайне негативно сказывается на зрении.

Таким образом, если людям, контактирующим с клавиатурой не больше получаса в сутки, отсутствие навыка быстрой печати не слишком усложняет жизнь, то стенографам и редакторам, печатающим на уровне ~100 знаков в минуту, каждый рабочий день наверняка кажется катастрофой. Как выразился в своей книге В.Ю. Холкин, «в современных условиях человека, не умеющего быстро и качественно печатать, можно назвать полуграмотным!» [1].

Почему же обыденный процесс является таким сложным? Проблема заключается в технике набора. Бόльшая часть людей печатает 2-4 пальцевым или, так называемым, новичковым методом («найти и нажать»). Изучив статистику соревнований по печати, можно заметить, что предел скорости для данного метода составляет около 300 знаков в минуту. Речь идёт о профессиональном наборе новичковым методом, не говоря уже о рядовом.

Естественно, вопрос оптимизации данной проблемы захватил умы людей уже очень давно, и эвристика скорости печати на данный момент имеет довольно ёмкий содержательный характер. «Венцом творения» данной науки является *10-пальцевый* (*слепой*) метод печати.

Данный метод был разработан стенографистом из Солт-Лейк-Сити Франком Эдгардом Макгуррином. 25 июля 1888 года он одержал победу в первом соревновании по печатанию, проведённому в Цинциннати. Его выигрыш составил 500 долларов (в нынешнем эквиваленте 11 675 долларов). Эта победа вызвала широкий интерес и к слепому методу, и к пишущим машинкам [2].

В данной работе содержится подробный анализ «слепого метода печати», а также приведён отчёт по разработке системы обучения этому методу.

**1 Концепция «десятипальцевого метода набора»**

**1.1 Описание метода**

Полное название метода – «Слепой десятипальцевый метод набора». Это методика набора текста десятью пальцами «вслепую», то есть без зрительного контакта с клавиатурой. Как было сказано во введении, метод был изобретён Франком Эдгардом Макгуррином более 125 лет назад.

У метода есть два фундаментальных положения, которые его и определяют:

- нельзя устанавливать зрительный контакт с клавиатурой;

- за каждым пальцем закреплены свои клавиши, то есть клавиатура разбита на непересекающиеся области для соответствующих пальцев.

Для овладения навыком слепой печати необходимо, в первую очередь, знать о «домашних» клавишах для пальцев:

'ф' – левый мизинец;

'ы' – левый безымянный палец;

'в' – левый средний палец;

'а' – левый указательный палец;

'о' – правый указательный палец;

'л' – правый средний палец;

'д' – правый безымянный палец;

'ж' – правый мизинец.

На кнопках 'а' и 'о' расположены специальные выступы, которые называются *питами*. Они предназначены для правильной первоначальной установки пальцев на клавиатуре (быстрого поиска «домашних» кнопок).

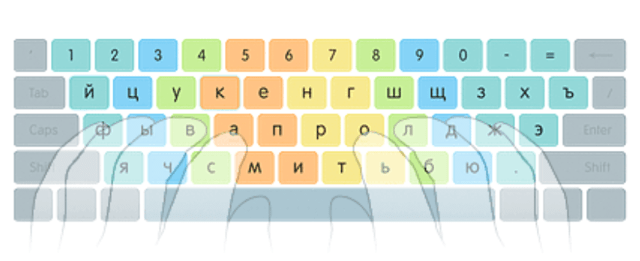
****

Рисунок 1 – Правильное расположение пальцев при «слепой» печати

**1.2 Преимущества**

У «слепого» метода печати есть ряд достоинств перед «зрячим» методом, рассмотрев которые можно с уверенностью сказать, что каждый человек должен им овладеть.

1) при десятипальцевом «слепом» методе значительно возрастает скорость набора текстов. Так, пользователь, печатающий десятью пальцами, может достигать скорости, в 3 раза превышающей скорость двумя-четырьмя. На данный момент рекорд на соревнованиях по набору данным методом незнакомого текста находится на отметке 940 зн/мин и принадлежит Михаилу Шестову. Применяя «скоропечатание», набор текста станет более автоматизированным процессом. В отличие от «зрячего» метода набора, у которого прослеживается резкое снижение эффективности на больших текстах, которое можно сравнить с гиперболической функцией, изменение эффективности при наборе таких же текстов «слепым» методом является почти константной функцией.

2) как было отмечено во введении, от скорости набора текста зависит трудоёмкость и время выполнения задания. Сосредотачиваясь на тексте, а не на поиске клавиш, можно избежать рассеянности при наборе, значительно повышается концентрированность и внимательность. Также слепой метод имеет безоговорочное преимущество при работе в плохо освещённом помещении.

3) при правильном быстром наборе, в сравнении с обычным, значи-тельно сокращается количество допускаемых ошибок и опечаток. Это связано с тем, что набор текста пальцами происходит машинально, а мозг в это время может полноценно функционировать, анализируя информацию с дисплея.

4) «слепой» метод набора позволяет избежать многих проблем со здоровьем, связанных с печатью на клавиатуре: отсутствие необходимости переводить взгляд с клавиатуры на монитор и назад способствует сбережению зрения, а также позволяет пользователю поставить правильную осанку и находиться на безвредном расстоянии 30 см от экрана. Также нельзя не упомянуть, что уменьшение количества ошибок благоприятно сказывается на нервной системе.

5) данный метод позволяет пользователю добиться ритмичности при наборе текста. Это очень важно, поскольку аритмичный набор ведёт к сильному переутомлению во время печати.

6) десятипальцевый набор текста воздействует сразу на несколько отделов мозга, активная работа которого способствует быстрой обработке информации и формированию синапсов между нейронами. Это ведёт к улучшению памяти, концентрации и даже воображения. К тому же, учёными-физиологами установлено, что центры головного мозга, отвечающие за речь и за движение пальцев, расположены близко друг к другу. Таким образом, стимуляция центра, отвечающего за движение пальцев, приводит к активизации процессов в речевом центре [3]. Однако необходимо отметить, что данная связь проявляется у детей дошкольного и младшего школьного возраста, поэтому важно привить им желание заниматься печатанием десятью пальцами ещё в детстве.

По данным источников, в США слепым методом владеет около 95% людей, причём начинают учиться ему ещё в начальной школе [4].

**1.3 Сложности, связанные с обучением**

Итак, преимущества «слепого» метода набора неоспоримы. Более того, все недостатки, которые можно у него выделить, будут «притянутыми за уши». Тем не менее, бльшая часть людей, осведомлённых об этом совершенном методе, не изъявляют желания начинать обучение, а среди начавших 90% не проходят даже десятой части курса. Что же мешает пользователям овладеть данным навыком?

Во-первых, переход от 2-4-пальцевого набора к 10-пальцевому для рядового человека проходит довольно тяжело. Это связано с тем, что при поиске нужных клавиш мышечная память у пальцев почти не развивается: каждый палец нажимает ту кнопку, которая ему удобна в этот момент. Более того, из анатомии известно, что человеческая кисть имеет сложное строение: мышца-разгибатель (экстензор) разгибает или выпрямляет палец при сокращении, мышца-сгибатель (флексор) сгибает его. У мизинца, указательного и большого пальцев есть свои экстензоры, поэтому процесс и разгибания проходит довольно легко. А вот у среднего и безымянного пальцев они общие. Таким образом, у людей, не развивавших мелкую моторику музыкальными инструментами или пальцевой гимнастикой, связка среднего и безымянного пальцев не позволяет свободно ими управлять. Также стоит упомянуть, что мизинец, в силу процесса эволюции, в настоящее время является слабым, малоэффективным в процессе печати и требует отдельной подготовки, которую тяжело проходить обычному пользователю.

Во-вторых, вопреки тому, что подавляющим большинством пользователей используется стандартная раскладка клавиатуры – ''qwerty'', некоторые люди продолжают активно пользоваться другими, менее известными или устаревшими раскладками, например, ''Dworak''. При этом большинство существующих клавиатурных тренажёров направлено на разработку «слепой» печати именно для раскладки ''qwerty''.



Рисунок 2 – Раскладка клавиатуры ''Dworak''

В-третьих, несмотря на то, что запоминание пальцами положения кнопок проходит в течение недели, полученный навык должен «подпитываться» постоянными тренировками и разминками для того, чтобы установить для каждого сочетания чёткую синоптическую связь. Без этого способность «слепой» печати будет неполноценной. Таким образом, взяв в рассмотрение человеческий фактор, можно понять, почему такая большая цифра людей не хочет проходить полное обучение.

В-четвёртых, от самой клавиатуры также зависит эффективность тренировки «слепого» метода. Самыми популярными на данный момент видами клавиатур являются механическая, мембранная и ножничная (используется на большинстве ноутбуков) клавиатура. Они отличаются высотой положения кнопок, долговечностью, силой, необходимой для нажатия, издаваемыми звуками при печати, ну и, естественно, структурно. Известно, что на ножничной клавиатуре намного легче развивать большие скорости, чем, например, на механической. Более того, пальцам не приходится прилагать большие усилия для нажатия. Однако ножничные клавиатуры дороже механических, поэтому большинство пользователей пользуется механическими.

В-пятых, при печатании человек очень сильно привыкает к своей клавиатуре, вследствие чего даже небольшие изменения при смене клавиатуры приводят к *снижению эффективности печати*!Печатающему на одной клавиатуре зачастую приходится почти с нуля тренироваться на другой, пусть даже такого же типа клавиатуре.

Несмотря на все эти сложности, подавляющее большинство пользователей, научившихся быстро печатать «вслепую», оценивает этот навык, как легко усваиваемый и доступный, коим он и является. Поэтому целью этой исследовательской работы является также пролитие света на кажущийся непосильным и ненужным курс обучения «слепому» методу.

Далее будут рассмотрены концепция клавиатурных тренажёров и основные элементы функционала.

**2 Клавиатурные тренажёры**

Как можно было понять из материала выше, «слепой» десятипальцевый метод набора требует такой же специальной подготовки, как и любой вид спорта, поэтому за 125 лет существования данной методики много сил и времени было вложено в разработку всевозможных клавиатурных тренажёров.

Что такое клавиатурный тренажёр? Это вид компьютерных программ или онлайн-сервисов, предназначенных для обучения набору на компьютерной клавиатуре [5].

Направленность:

1) освоение навыка десятипальцевого «слепого» метода печати;

2) увеличение скорости печатания;

3) уменьшение количества ошибок;

4) улучшение ритмичности набора.

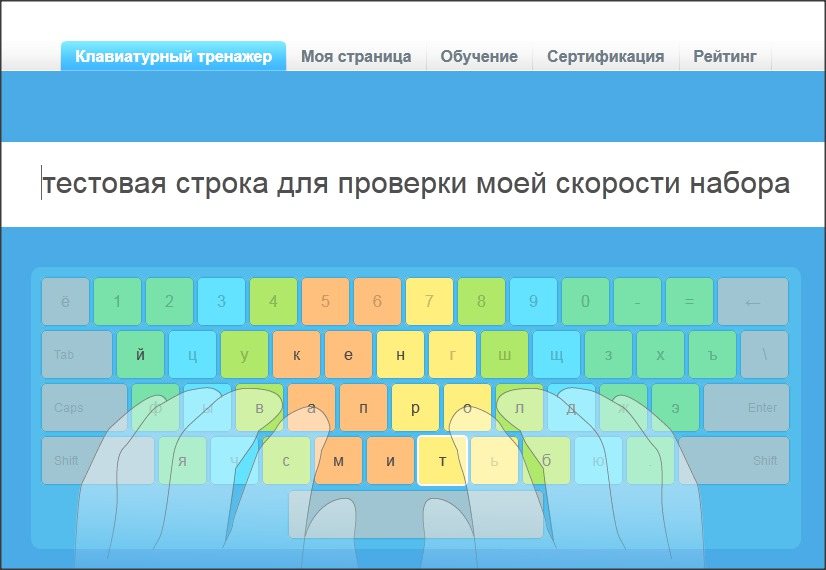


Рисунок 3 – Окно клавиатурного тренажёра, демонстрация

Как было сказано ранее, некоторые из клавиатурных тренажёров предлагают возможность выбора раскладки и тренировки «слепого» метода печати на них.

Рассмотрим несколько популярных онлайн-сервисов для развития «слепого» метода печати.

**2.1 Анализ популярных тренажёров**

**2.1.1 «СОЛО на клавиатуре»**

Автор данного курса – известный психолог, журналист и преподаватель МГУ Владимир Владимирович Шахиджанян. Для реализации проекта он организовал дружный коллектив единомышленников, и на данный момент «СОЛО на клавиатуре» является самым известным и проверенным временем клавиатурным тренажёром.

Данный сайт предоставляет возможность научиться «слепому» методу набора на русской, английской и немецкой раскладках, а также пройти обучение набора на цифровом блоке справа.

Отличительной особенностью данного тренажёра является глубокий анализ активности пользователя, большой функционал, а также разбавление череды уроков психологическими монологами и развлекательными текстами.

Курс обучения является тщательно проработанной совокупностью грамотно расставленных и гармонично переходящих друг в друга уроков. В одном курсе собрано около 100 уроков средней сложности. После прохождения курса у пользователя есть возможность практиковаться в новом навыке на специальной площадке «Гонки на клавиатуре». Каждому игроку по окончании заезда предоставляются такие характеристики, как ритмичность, задержка, пауза и так далее. Также есть возможность практиковаться под диктовку и развить навыки транслитерации.

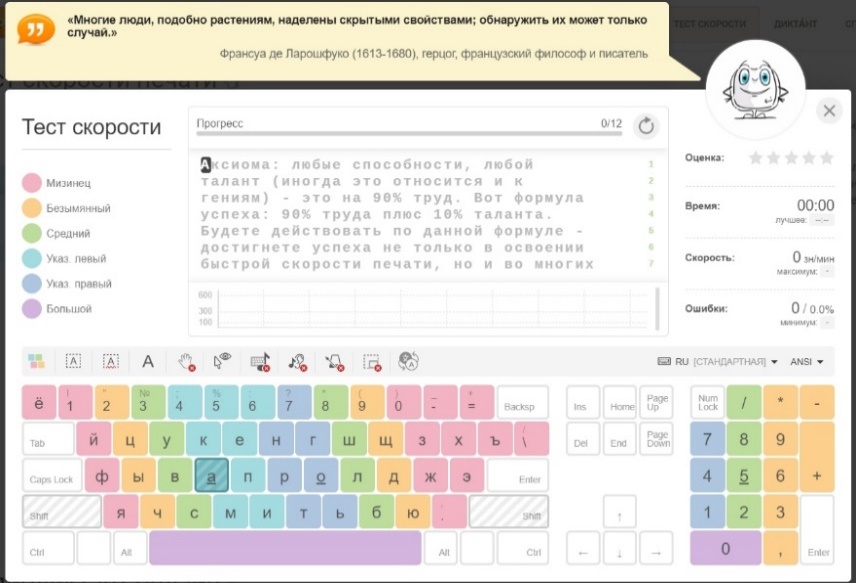


Рисунок 4 – СОЛО на клавиатуре, окно набора текста

**2.1.2 «Ratatype»**

Этот клавиатурный тренажёр позволяет пройти курс «слепой» печати для русской, украинской, английской, испанской и французской раскладок.

В отличие от «массивного» тренажёра «СОЛО на клавиатуре», курсы данного сайта отличаются довольно маленьким объёмом, но при этом большой эффективностью. Весь курс, при должном терпении и усидчивости, можно пройти за 2-3 дня, и прогресс будет очевидным.

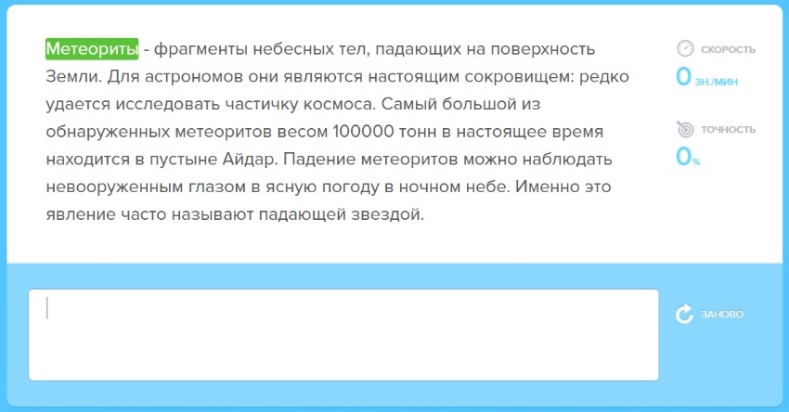


Рисунок 5 – «Ratatype», окно набора текста

На сайте не предусмотрена возможность оттачивания своего мастерства на специально подобранных текстах, однако есть функция сертификации: пользователь, напечатавший специфичный текст, получает уникальный «сертификат печати», который может впоследствии использовать в резюме. Пример сертификата приведён на рисунке ниже.



Рисунок 6 – Сертификат печати с сайта «Ratatype»

**2.1.3 «Stamina»**

У многих людей сочетание «клавиатурный тренажёр» ассоциируется со словом «Стамина». Это один из самых популярных тренажёров в мире, выпущенный в 1999 году и на данный момент представляющий собой «старую школу» печати. Фундаментом для него служит очень мощный курс обучения «слепой» печати, прорабатывающий каждую мелочь.

Возможности:

- работа с реальным текстом;

- уроки на сочетания букв;

- возможность набора текста из внешнего файла;

- возможность набора под диктовку;

- редактор уроков [6].

Так же, как и во многих тренажёрах, здесь пользователю предо-ставляется подробный анализ активности по дням и сеансам, а также специфичная статистика набранных за определённое время килобайт.

Курсы отличаются грузностью и большой эффективностью, так как в процессе обучения отрабатываются все «сложные» кнопки и сочетания.

Стоит отметить, что интерфейс программы не предлагает чёткой инструкции, в какой последовательности следует работать с уроками. Одной из особенностей программы является её юмористическая составляющая.

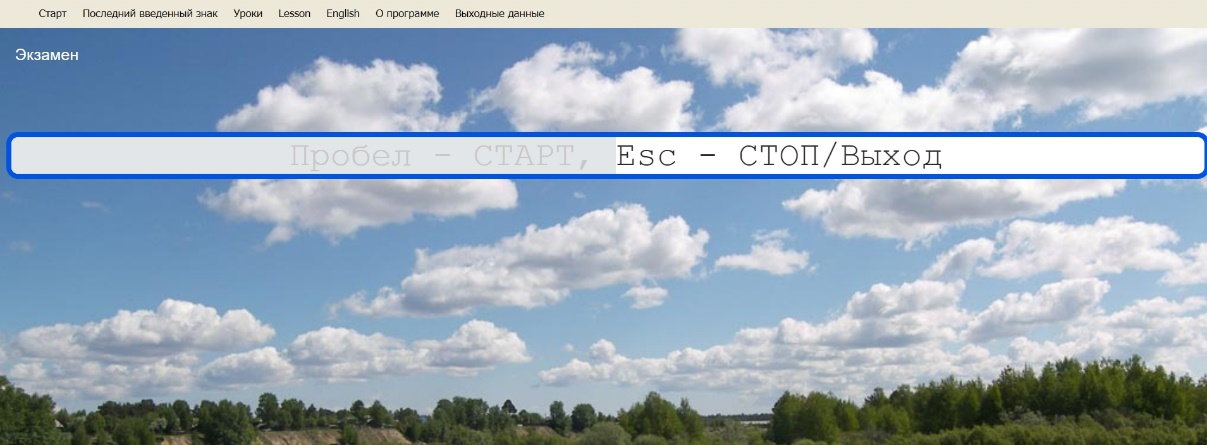


Рисунок 7 – «Stamina», окно набора текста

**2.1.4 «Клавогонки»**

Большой акцент необходимо сделать на этом уникальном сайте. Несмотря на то, что данный клавиатурный тренажёр не содержит в себе обучающих курсов от модераторов и вообще не направлен на обучение «слепой» печати, он является незаменимым для тех, кто планирует тренировать этот навык.

Основанный в 2008 году, этот сайт нашёл огромную популярность среди печатающих вслепую по всему миру. Возможности пользователя ограничиваются добавлением «словарей» - текстов или генераторов текстов из сочетаний, а также использованием словарей от других пользователей. Тем не менее, тренажёр разросся в огромную площадку с большими комплексами упражнений разной сложности, создаваемыми пользователями всех возрастов и профессий.

Сам набор текста, как можно догадаться из названия тренажёра, представляет собой гонку на автомобилях, где побеждает тот, кто быстрее набрал текст. При этом на тексты могут быть наложены определённые ограничения по ошибкам или скорости.

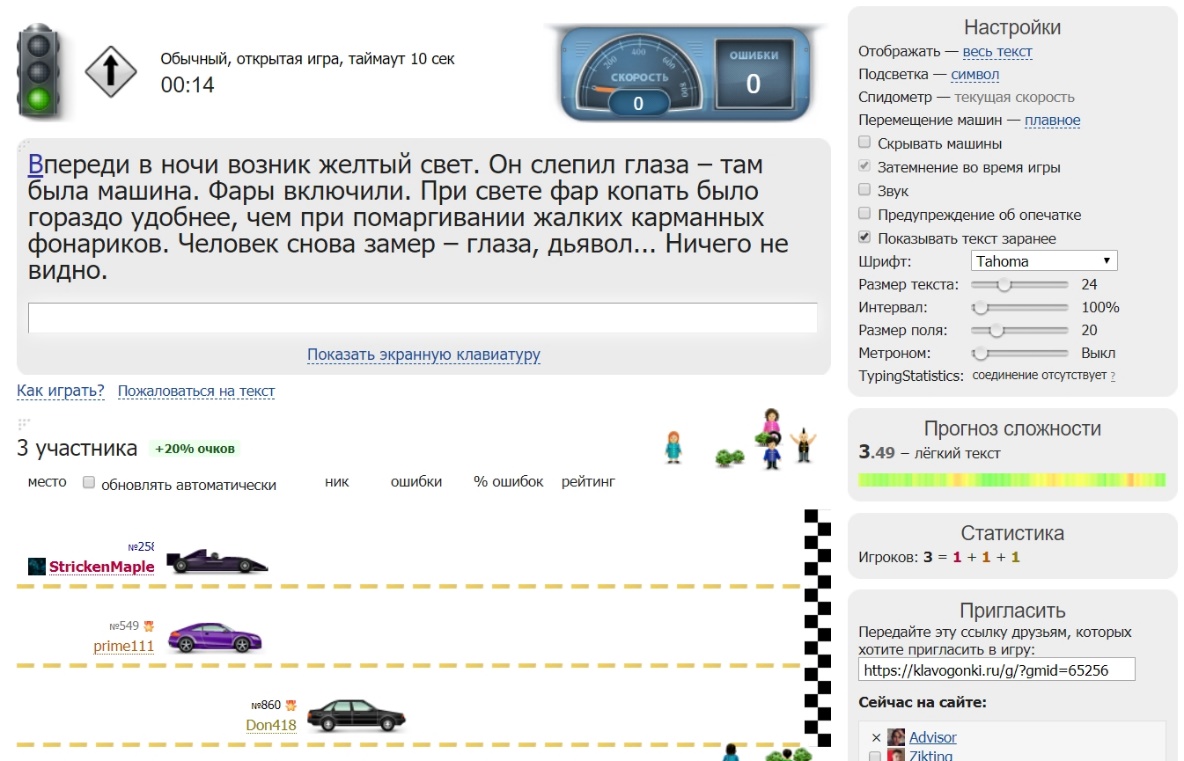


Рисунок 8 – «Клавогонки», окно набора текста

Почему этот сайт стоит принять во внимание? На данный момент на нём тренируется люди, способные в любое время занять место Михаила Шестова в книге рекордов Гиннеса. Именно этот тренажёр позволяет превзойти собственные пределы и преобразовать тренировки в чистую эффективность.

**2.2 Разбор функциональной части**

Перед разработкой тренажёра необходимо рассмотреть основной функционал, который должен быть в нём реализован.

**2.2.1 Деятельность пользователя**

Проанализировав популярные клавиатурные тренажёры, можно выделить несколько основных интерфейсных действий, доступных пользователю.

1) авторизация. Необходимым элементом любого тренажёра является сохранение результатов пользователя с целью последующего анализа статистики и активности. Таким образом, в нашем клавиатурном тренажёре должна быть система авторизации и аутентификации пользователя.

2) режим обучения. Вернувшись к пункту 2.1.4, можно заметить, что клавиатурный тренажёр не обязан содержать в себе обучение «слепому» методу, так как смысловая значимость его всё-таки в оттачивании этого навыка. Тем не менее, для каждого крупного клавиатурного тренажёра в своё время был составлен уникальный курс с целью обучить пользователя максимально понятным и простым способом. Исходя из этого, можно добавить обучающий курс в список значимых элементов тренажёра.

3) режим тренировки. Как было отмечено ранее, «слепой» десятипальцевый навык набора, обособленный от регулярных тренировок, является неполноценным. В силу этого, подавляющее большинство сайтов с клавиатурной тематикой снабжено режимом тренировок, богатым в функциональном плане. Его эффективность можно охарактеризовать тем, насколько улучшил свои показатели пользователь после оттачивания мастерства в этом режиме. Поэтому наполненный качественный текстами режим тренировок также является неотъемлемой частью реализации.

**2.2.2 Программный функционал**

Элементы пользовательского функционала разобраны. Теперь нужно определиться с функциями, которые должны быть прописаны в клавиатурном тренажёре.

1) таймеры

Для подсчёта всего, что связано со скоростью или ритмичностью набора, необходима организация таймеров. Также один из них нужен для интерфейсного отображения текущего времени набора.

2) скорость

Минимальные требования для тренажёра – это подсчёт средней и текущей скорости набора. Для подсчёта скорости можно выделить две основные единицы измерения:

- слов в минуту;

- знаков в минуту.

Первая единица измерения является более чем абстрактной и отдалённой от реальности, поскольку единица «слово» не имеет определённого значения. Поэтому редко можно встретить тренажёр, использующий данную единицу измерения.

Вторая, в свою очередь, характеризуется количеством напечатанных (правильно!) символов в минуту и используется в подавляющем большинстве тренажёров.

С подсчётом средней скорости всё предельно просто – расстояние на время. Однако с текущей скоростью всё не так однозначно. Мы вернёмся к этому в процессе разработки программы.

3) точность набора

Характеристика, получаемая делением количества символов в тексте на количество напечатанных пользователем, или, по более простой формуле:

(1)

где – точность набора, – количество допущенных ошибок, – количество символов в тексте. Данная характеристика не является основной для интерфейсной части и может быть заменена количеством ошибок при наборе, но для статистики имеет более важное значение, поэтому вернёмся к ней позже.

4) сохранение данных набора

Занесение данных в базу с целью последующего обращения к ним также является неотъемлемой частью полноценного клавиатурного тренажёра. На этом этапе разработки программы функция анализа данных не прописывается, но перспектива развития имеется.

5) система рангов в курсе обучения

Данный пункт не будет рассмотрен в части реализации, поскольку не имеет ничего общего с программными вычислениями и полностью организуется в базе данных. Тем не менее, он стоит упоминания в этом разделе.

Для каждого урока в базе данных хранятся «нормативы», которые в какой-то степени характеризуют уровень пользователя при прохождении урока. В базе будет предусмотрено 5 рангов:

- новичок;

- любитель;

- профи;

- эксперт;

- чемпион.

Ранги отличаются между собой порогом скорости набора и количества допустимых ошибок. По прохождении главы/курса пользователю предоставляется его средний ранг по данной главе/курсу.

**3 Разработка тренажёра «Акселератор»**

**3.1 Выбор языка и среды разработки**

В качестве языка программирования для написания программы был выбран язык C++.

Данная программа разрабатывалась с использованием Qt – самого популярного кроссплатформенного фреймворка для разработки программного обеспечения на C++, имеющего очень удобный API и задающего красивую и легко модифицируемую архитектуру приложения. Изучение данного фреймворка происходило с нуля, поэтому потребовался источник [7].

Соответственно, в качестве среды разработки для данной программы был выбран Qt Creator.

**3.2 Семантика процесса**

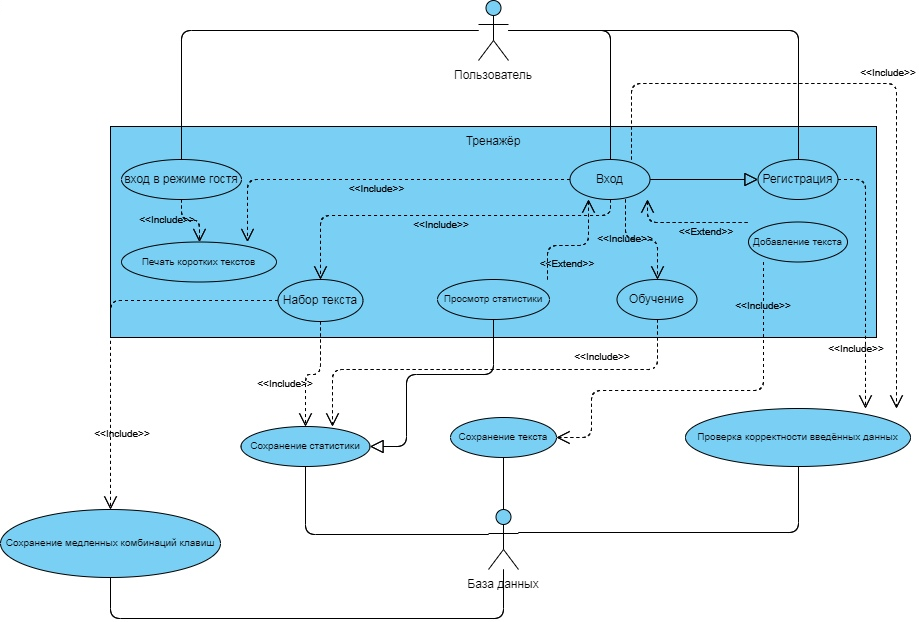


Рисунок 9 – Диаграмма вариантов использования для тренажёра

На рисунке выше приведена диаграмма, отражающая отношения между кторами (пользователем и базой данных) и прецедентами.

Несмотря на то, что диаграмма составлена для визуализации работы базы данных (курсовой проект «Разработка базы данных клавиатурного тренажёра»), она очень удачно описывает систему клавиатурного тренажёра на концептуальном уровне.

**3.3 Реализация**

Разобьём процесс разработки на несколько этапов:

1) база данных;

2) пользовательский интерфейс;

3) функционал.

Почему разработка будет рассматриваться именно в таком порядке?

База данных является фундаментом для проекта. Бльшая часть функций, которые имеют реализацию в программе, должна будет взаимодействовать с сущностями базы, получая или отправляя в них пользовательские данные. Более того, подавляющее большинство методов в процессе набора работает с элементами пользовательского интерфейса, например, панелью ввода текста.

**3.3.1 База данных**

Ниже приведена ER-диаграмма, представляющая физическую модель базы данных клавиатурного тренажёра. Далее будем рассматривать базу, как готовый продукт, который нужно только подключить к программе (подробное описание всех сущностей и связей между ними приведено в курсовой работе «Разработка базы данных клавиатурного тренажёра»).

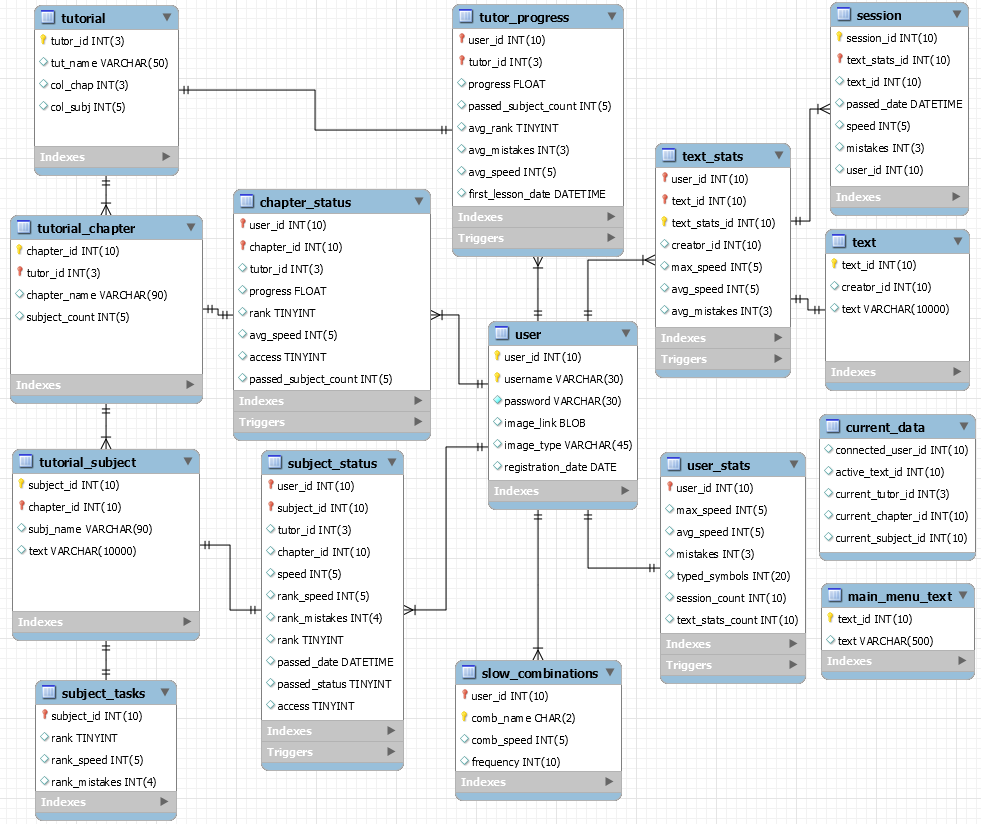


Рисунок 10 – ER-диаграмма, составленная для базы данных

База данных реализована на языке MySQL в инструменте MySQL Workbench. Для обслуживания базы, а также обеспечения сохранности и целостности хранимых данных необходимо зарегистрировать сервер базы данных.

Перед подключением к программе необходимо дописать в файле с расширением .pro слово «sql»: это даст возможность QT понимать SQL-запросы.

QT += core gui sql

Для подключения базы потребуется всего одна библиотека:

#include <QSqlQuery>

Класс QSqlQuery обеспечивает интерфейс для выполнения SQL запросов и навигации по результирующей выборке.

Чтобы осуществить подключение, пропишем в программе следующий код:

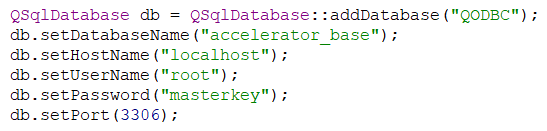


Рисунок 11 – Подключение базы данных к программе

Для открытия и закрытия базы данных используются методы open() и close() соответственно.

Оформление запросов и обработка результирующей выборки не являются задачами данного курсового проекта, поэтому примеры по запросам к базе в основном тексте работы приведены не будут.

Также в работе не будут рассмотрены триггеры, хранимые процедуры и функции, являющиеся частью базы и поступающие в качестве данного.

**3.3.2 Интерфейс**

Самой интересной и, в то же время, рутинной частью реализации является создание в программе полноценного пользовательского интерфейса. Основные окна:

- главное окно;

- окно авторизации;

- окно пользовательских данных;

- окно изменения пользовательских данных.

Все кнопки pushbuttons, используемые в окне программы, имеют один стиль, созданный специально под данный интерфейс.



Рисунок 11 – Пример кнопки в активном, нажатом и пассивном состоянии

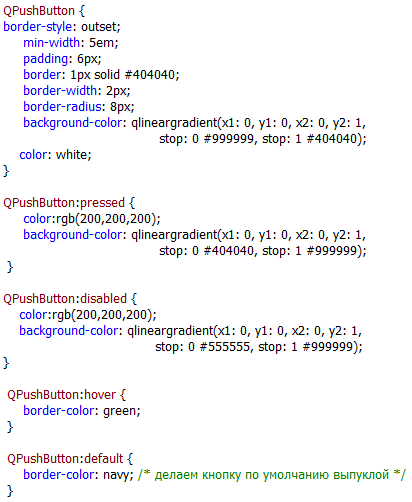


Рисунок 12 – Код стиля для Pushbuttons

В главном окне, помимо выбора режимов и авторизации пользователя, будет происходить и сам набор текста. Также реализована подсветка правильных клавиш на клавиатуре, что будет описано ниже.

Сам текст будет отражаться в метке QLabel Text.

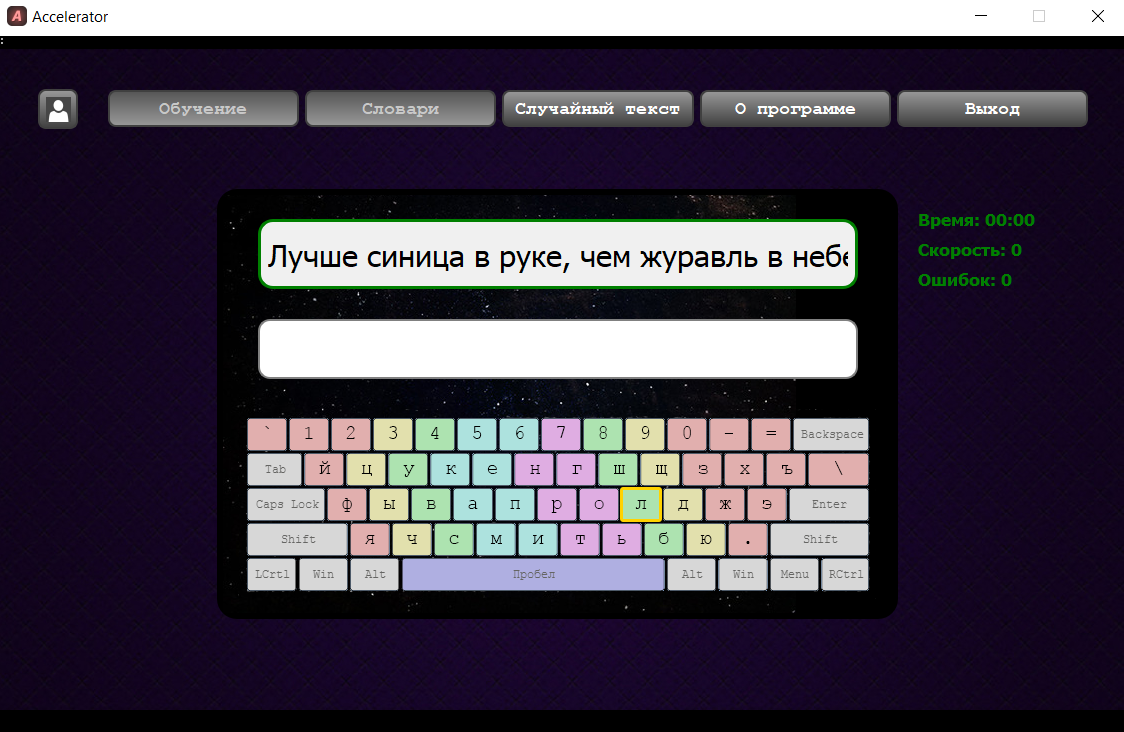


Рисунок 13 – Демонстрация основного окна программы

Окно содержит 6 функциональных кнопок, среди которых особый интерес представляют авторизация, обучение и словари.

1) *Авторизация*

При нажатии на кнопку пользователю предоставляется возможность аутентификации с целью сохранения результатов обучения и набора текстов.

2) *Обучение*

На данном этапе разработки при нажатии на кнопку пользователь может только продолжить обучение с того места, на котором остановился. Режим доступен только зарегистрированным пользователям. Это связано с необходимостью хранить статистику набора и прохождения курсов.

3) *Словари*

При нажатии на эту кнопку пользователь может набирать специальные тексты, при этом результаты набора сохраняются в его статистике.

Отдельного рассмотрения требуют следующие интерфейсные функции приложения:

1) Считка и проверка символов во время ввода

Пользователь вводит символы в QLineEdit text\_input, находящийся под меткой Text, при этом программа реагирует на любое изменение панели text\_input. Если введённый символ не соответствует первому символу строки в Text, то количество ошибок увеличивается на единицу, иначе в строке удаляется первый символ. В обоих случаях символ, введённый в text\_input, стирается. Такая проверка ввода является наиболее компактной и легкой в реализации. В приложении можно найти код данной проверки.

2) Появление красной рамки при неправильном наборе

3) Подсветка «правильных» кнопок на картинке с клавиатурой

На рисунке 13 можно заметить, что буква 'л' имеет золотую рамку. Для организации подсветки нужных кнопок при вводе текста было создано ровно 50 слоёв формата png. Для автоматизации смены подсветки каждой картинке было дано название, соответствующее подсвечиваемому ей символу, что немного облегчило программную обработку слоев. Программисту остаётся только отдельно прописать подсветку пробела, шифта и обратного слэша (который является специальным символом и не может присутствовать в названии файла).

4) Выдача результатов набора

В главном окне характеристики набора представляются пользователю в двух экземплярах: справа – текущие данные, обновляемые с определённой частотой (например, полсекунды), слева – статистика за пройденный текст, появляющаяся в конце набора и исчезающая через определённое время (10 секунд).

Более подробно функциональная часть будет рассмотрена в разделе «Функционал».

Как было сказано при разборе интерфейсных действий, пользователь должен иметь возможность авторизоваться с целью сохранения результатов обучения и данных набора. Это будет происходить в окне авторизации.

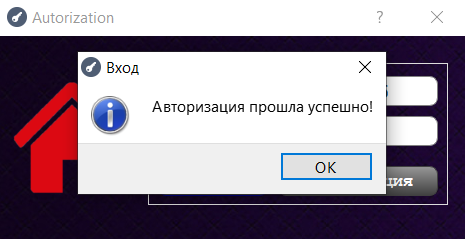
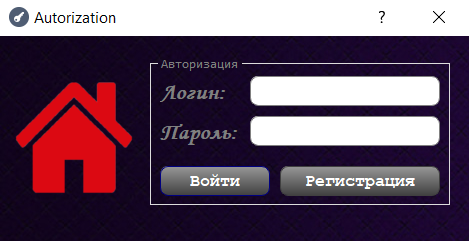


Рисунок 14 – Окно авторизации, демонстрация

Для полей ввода логина и пароля был создан валидатор, запрещающий символы кириллицы и пробелы.

Можно догадаться, что кнопки «Войти» и «Регистрация» взаимодействуют с полями сущности «Пользователь» в базе данных, после чего данные в базе могут быть изменены с помощью триггеров или хранимых процедур, а пользователю предоставляется доступ к режимам «Обучение» и «Словари» в главном окне.

В окне пользовательских данных не предусмотрены пользовательские действия, кроме нажатий на кнопки, поскольку данные статистики пользователя полностью хранятся в базе и получаются напрямую из полей сущностей или подсчётами в хранимых процедурах и функциях.

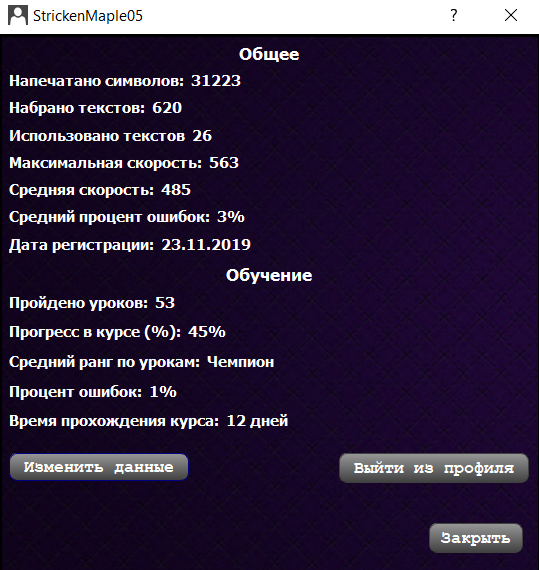
****

Рисунок 15 – Окно пользовательских данных, демонстрация

В окне изменения пользовательских данных и, соответственно, в базе данных, предусмотрена возможность изменения пользователем логина и пароля. Проверке на корректность ввода данных также уделено большое значение, а именно, выдача ошибок на ввод существующего логина, неправильного старого пароля, несоответствие новых паролей, совпадение старого пароля и нового.

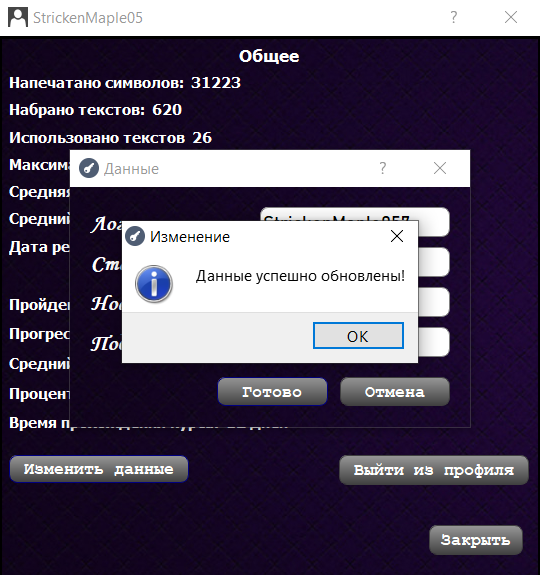
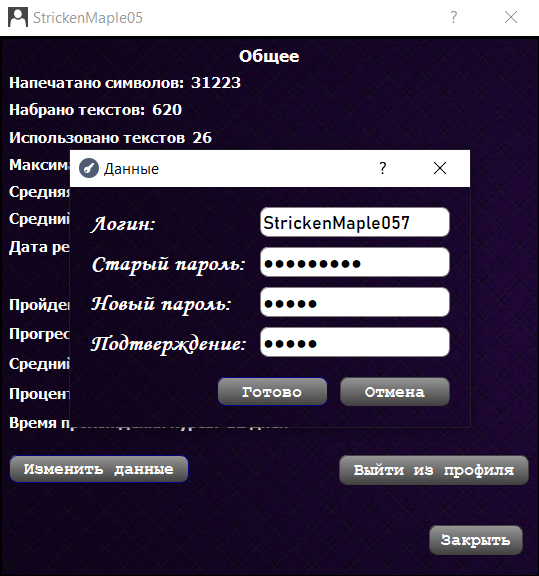


Рисунок 16 – Окно изменения данных, демонстрация

**3.3.3 Функционал**

В этом разделе приведён разбор основных программных действий, методов и алгоритмов, используемых в данном тренажёре.

Коды описанных функций, а также интересующие моменты в реализации можно будет найти в приложении.

Многие функции, используемые в программе, прописаны, как методы классов окна, с целью взаимодействия с ui-файлами, принадлежащими соответствующим классам.

Необходимо отметить, что среди используемых в программе функций интерес могут представлять только таймеры, поскольку остальные методы предназначены, в основном, для упрощения взаимодействия с графическим интерфейсом. В связи с этим в данном разделе будет содержаться только разбор основных таймеров программы.

Итак, для клавиатурного тренажёра было создано 5 таймеров:

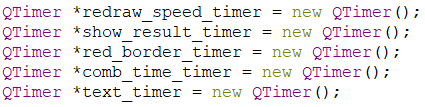


Рисунок 17 – Основные таймеры

1) text\_timer – это основной таймер, запускаемый при первом изменении text\_input после окончания набора предыдущего текста и вызываемый каждую миллисекунду. Прошедшие миллисекунды накапливаются в переменной msec. Также данный таймер отвечает за частоту перерисовки интерфейсного секундомера, на чём, пожалуй, не стоит заострять внимание.

2) show\_result\_timer – время отображения результатов набора слева от экрана (по умолчанию 10 секунд). Вызывается в момент, когда строка, отображаемая в Text, становится пустой, то есть пользователь ввёл все символы текста. Данный таймер отвечает за отображение средней скорости, количества ошибок и времени набора текста.

Перед вызовом таймера программой высчитывается средняя скорость набора текста – количество символов в тексте, делённое на значение msec и умноженное на 60 000 (для получения нужной единицы измерения). Время набора и количество ошибок получаются из соответствующих полей справа от окна набора.

3) red\_border\_timer – время существования красной рамки после неправильно набранного символа (по умолчанию полсекунды).

4) comb\_time\_timer – очень специфичный таймер, созданный для подсчёта текущей скорости набора. Дело в том, что уменьшение расстояния, то есть рассматриваемого количества напечатанных символов, в величине «текущая скорость» делает её более абстрактной и лишает прикладного значения.

Бльший интерес представляет так называемая средняя текущая скорость, вычисляемая по формуле:

(2)

– количество символов , – время набора -й пары символов, при этом -я и -я пара имеют смежный символ.

Несмотря на явную зависимость данной величины от параметра , влияющего и на погрешность в вычислении, скорость, посчитанная по этой формуле, будет семантически гораздо ближе к искомой. Всё из-за того, что сама по себе величина текущей скорости не имеет чёткой математической модели в том понимании, в котором она представляется сейчас. Вот почему у разных клавиатурных тренажёров можно наблюдать небольшие отличия в измерении данной величины.

Для определённости параметр в «Акселераторе» взят равным 5. Посчитанная на таком значении довольно точно определяет текущую скорость в привычном нам её представлении.

На рисунке ниже приводится схема заполнения массива из 5 элементов временными интервалами (в миллисекундах): по нажатию «правильной» кнопки временной интервал заносится в массив, при этом смещая все элементы влево. Таким образом, после каждого вызова таймера comb\_time\_timer будем иметь актуальную на данный момент текущую скорость.

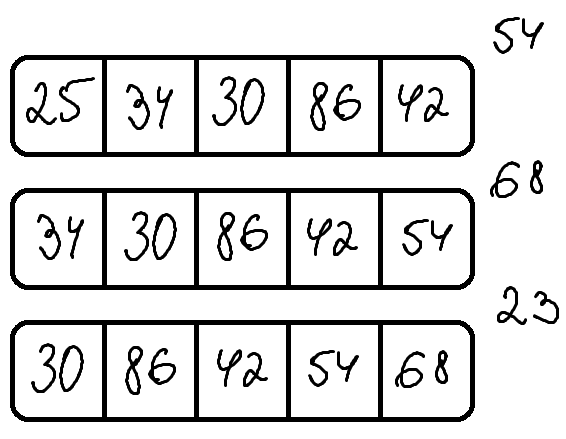


Рисунок 18 – Схема сохранения временных интервалов

У этого алгоритма есть один существенный недостаток: если пользователь долгое время не нажимает на «правильную» кнопку, данные массива обновляться не будут, следовательно, и текущая скорость будет постоянной. Во избежание этого на время набора пары символов установлен верхний предел (800 мс), по преодолении которого последний элемент массива просто получает на вход текущее значение временного интервала. Данная реализация позволяет текущей скорости свободно меняться, в зависимости от деятельности пользователя.

5) redraw\_speed\_timer – таймер, отвечающий за перерисовку текущей скорости набора.

Разобрав тонкости реализации данной величины, можно догадаться, что проверку на преодоление верхнего предела можно проводить именно в этом таймере, потому что пользователь не может получать значение средней скорости чаще, чем она перерисовывается на экране.

Также в методе, являющемся слотом для данного таймера, реализован, непосредственно, сам пересчёт величины текущей скорости по формуле (2).

Говоря о функциях, которые могут представлять интерес, нужно также указать выборку самых медленных сочетаний символов по данному тексту.

Реализация метода довольно проста: поиск минимумов временных интервалов, а также сохранение сочетаний, соответствующих данных интервалам. Однако данная функция будет иметь большое прикладное значение (см. заключение).

Также необходимо отметить, что упоминание ранее величины «точность набора» сопровождалось заменой её на количество ошибок в интерфейсной части. Это связано с тем, что бльшую практическую ценность для пользователя, в силу человеческого фактора, имеет количество допущенных им ошибок, нежели точность набора текста, хотя данные характеристики могут быть напрямую выражены друг через друга.

Однако для базы данных более приоритетной характеристикой является именно точность, поскольку количество ошибок является более громоздкой и неудобной для статистических анализов характеристикой.

**ЗАКЛЮЧЕНИЕ**

Цель курсовой работы – изучение концепции «слепого» метода набора и разбор известных клавиатурных тренажёров с последующей реализацией системы «Акселератор» – достигнута.

Перед написанием данной работы было изучено и пройдено около 5 различных курсов «слепой» печати, а также разобран функционал 7 популярных клавиатурных тренажёров.

Проект «Акселератор» разбит на две части: «База данных» и «Интерфейс», и, хотя целью данной курсовой работы не было исследование базы, для лучшего понимания процесса были принято участие в изучении и разработке концептуальной, логической и физической моделей базы данных.

На данном этапе реализации приложение предоставляет пользователю возможность авторизоваться в системе с целью прохождения курса печати или тренировки на «словарях». Результат набора каждого текста сохраняется для возможности последующего анализа и сбора статистики. Также во время тренировок пользователя происходит накопление самых медленных сочетаний в базе.

Основной перспективой развития является перенос данного приложения в онлайн-сервис с ёмкой функциональной частью.

Следующий этап реализации:

- доработка и интеграция собственного курса «слепой» печати;

- введение таких характеристик, как ритмичность и аритмия, имеющих большое прикладное значение, а также ряд величин из математической статистики (математическое ожидание, среднеквадратическое отклонение и так далее);

- добавление метронома, налаживающего ритмичность набора, которая играет большую роль при печатании больших текстов;

- использование данных о медленных сочетаниях в генерации текстов для тренировки навыков. Данная методика является наиболее эффективной при оттачивании мастерства «слепой» печати. Возможно использование нейронной сети.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1 Холкин В.Ю. Десятипальцевый метод печати на компьютере / СПб., 2000. – 60 с. (07.12.2019)

2 Слепой метод печати – Википедия: [Электронный ресурс] // ru.wikipedia.org. URL: <https://ru.wikipedia.org/wiki/Слепой_метод_печати#Раскладка_клавиатуры>. (08.12.2019)

3 Развитие мелкой моторики: [Электронный ресурс] // URL: <http://ds497nsk.edusite.ru/p21aa1.html>. (08.12.2019)

4 Слепой метод печати 10 пальцами – 4Brain: [Электронный ресурс] // 4brain.ru. URL: <https://4brain.ru/blog/слепой-метод-печати-10-пальцами>. (07.12.2019)

5 Список клавиатурных тренажёров – Википедия: [Электронный ресурс] // ru.wikipedia.org. URL: <https://ru.wikipedia.org/wiki/Список_клавиатурных_тренажёров>. (08.12.2019)

6 Stamina – Википедия: [Электронный ресурс] // ru.wikipedia.org URL: <https://ru.wikipedia.org/wiki/Stamina>. (08.12.2019)

7 Шлее М. Qt 4.8 Профессиональное программирование на C++ / СПб.: БХВ-Петербург, 2012. – 912 с. (08.12.2019)

**ПРИЛОЖЕНИЕ**

**Коды основных функций**

#define changeText(tut,i) {if (!tut) text = mas[i]; else text = tutorial[i]; text\_length = text.length();}

QString green\_border = "background-color: rgb(240,240,240); border style: outset; border: 3px solid green; border-width: 3px; border-radius: 15px; min-width: 10em; padding: 6px;";

QString red\_border = "background-color: rgb(240,240,240); border style: outset; border: 3px solid red; border-width: 3px; border-radius: 15px; min-width: 10em; padding: 6px;";

QTimer \*redraw\_speed\_timer = new QTimer();

QTimer \*show\_result\_timer = new QTimer();

QTimer \*red\_border\_timer = new QTimer();

QTimer \*comb\_time\_timer = new QTimer();

QTimer \*text\_timer = new QTimer();

struct combination

{

QString comb;

int time;

};

combination comb[5];

QString curr\_comb = "";

int msec = 0, msec\_pred = 0;

int ms = 0, s = 0, m = 0, h = 0;

int comb\_time[] = {0,0,0,0,0};

const int MAXX = 50;

QString tutorial[MAXX];

extern bool autorized;

extern QString login;

int text\_length;

QString text = "";

QString mas[MAXX];

QString tim = "Время: ";

QString mis = "Ошибок: ";

QString spd = "Скорость: ";

int i = 0, k = 0, mistakes = 0;

bool SCF = 0;

void MainMenuTextDB(QSqlDatabase db)

{

db.open();

QSqlQuery query = QSqlQuery(db);

query.exec("SELECT text FROM main\_menu\_text");

while (query.next())

mas[k++] = query.value(0).toString();

changeText(0, i = rand()%k);

db.close();

}

void Accelerator::SelectBacklight(QCharRef c)

{

bool shift = 0;

QChar q = c;

if (q == '!'){shift = 1; q = '1';}

if (q == '"'){shift = 1; q = '2';}

if (q == '№'){shift = 1; q = '3';}

if (q == '$'){shift = 1; q = '4';}

if (q == '%'){shift = 1; q = '5';}

if (q == ':'){shift = 1; q = '6';}

if (q == '?'){shift = 1; q = '7';}

if (q == '\*'){shift = 1; q = '8';}

if (q == '('){shift = 1; q = '9';}

if (q == ')'){shift = 1; q = '0';}

if (q == '\_'){shift = 1; q = '-';}

if (q == '+'){shift = 1; q = '=';}

if (q == ','){shift = 1; q = '.';}

if (q == '\\'){shift = 1; q = 'T';}

if (q == ' '){q = 'P';}

if ((q>='А')&&(q<='Я'))

{

shift = 1; q.toLower();

}

ui->backlight\_shift->setPixmap(QPixmap(shift ? ":/kb/pic/Keyboard/shift.png" : ""));

ui->backlight->setPixmap("pic\\keyboard\\" + (QString)q + ".png");

}

Accelerator::Accelerator(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::Accelerator)

{

QDir::setCurrent("E:\\Qt\_5.2.1\\Projects\\Accelerator\\");

ui->setupUi(this);

ui->intro->raise();

intro\_timer = new QTimer(this);

connect(intro\_timer,SIGNAL(timeout()),this,SLOT(IntroTimer()));

intro\_timer->start(1500);

db = QSqlDatabase::addDatabase("QODBC","printing\_trening");

db.setDatabaseName("accelerator\_base");

db.setHostName("localhost");

db.setUserName("root");

db.setPassword("masterkey");

db.setPort(3306);

ui->time->setText(tim + "00:00");

ui->current\_mistakes->setText(mis + "0");

ui->current\_speed->setText(spd + "0");

ui->past\_time->setText("");

ui->mistakes->setText("");

ui->avg\_speed->setText("");

srand(time(NULL));

MainMenuTextDB(db);

ui->Text->setText(text);

InitSlowestCombinations();

Accelerator::SelectBacklight(text[0]);

}

void Accelerator::SlowestCombinations(QString combination, int time)

{

int k = -1;

for (int i = 0; i < 5; ++i)

if (time > comb[i].time)

{k = i; break;}

if (k != -1)

{

for (int i = 4; i > k; --i)

comb[i] = comb[i-1];

comb[k].time = time;

comb[k].comb = combination.toLower();

}

}

void CurrentTime(int time)

{

comb\_time[0] = comb\_time[1];

comb\_time[1] = comb\_time[2];

comb\_time[2] = comb\_time[3];

comb\_time[3] = comb\_time[4];

comb\_time[4] = time;

}

void Accelerator::TextTimer()

{

ms = msec%1000; s = msec/1000; m = s/60; h = m/60; ++msec;

QString output\_time = tim + (h ? QString::number(h) + ":" : "") + (!(m/10) ? "0" : "") + QString::number(m) + ":" + (!(s/10) ? "0" : "") + QString::number(s) + ":" + QString::number(ms);

ui->time->setText(output\_time);

}

void Accelerator::RedrawSpeedTimer()

{

int delta = msec - msec\_pred;

if (delta > 800) comb\_time[4] = delta;

int time = 0;

for (int i = 0; i < 5; ++i)

time += comb\_time[i];

ui->current\_speed->setText(spd + QString::number(5\*60000/time));

}

void Accelerator::ShowResultTimer()

{

ui->past\_time->setText("");

ui->mistakes->setText("");

ui->avg\_speed->setText("");

show\_result\_timer->stop();

}

void Accelerator::RedBorderTimer()

{

ui->text\_background->setStyleSheet(green\_border);

red\_border\_timer->stop();

}

void Accelerator::on\_Autorization\_clicked()

{

if (!autorized)

{

Autorization sign\_in;

sign\_in.setModal(true);

sign\_in.*exec*();

if (autorized)

{

ui->tutorial->setEnabled(true);

ui->tutorial->setToolTip("Обучающий курс слепой печати");

ui->texts->setEnabled(true);

ui->texts->setToolTip("Тренировка на пользовательских текстах");

}

}

else

{

User user;

user.setWindowTitle(login);

user.setModal(true);

user.*exec*();

if (!autorized)

{

ui->tutorial->setEnabled(false);

ui->tutorial->setToolTip("Обучающий курс слепой печати (необходимо войти)");

ui->texts->setEnabled(false);

ui->texts->setToolTip("Тренировка печати на пользовательских текстах (необходимо войти)");

changeText(tut,i = rand()%k);

ui->Text->setText(text);

SelectBacklight(text[0]);

}

}

}

void Accelerator::on\_tutorial\_clicked()

{

if (!tut)

{

TutorialText();

tut = 1;

db.open();

QSqlQuery query = QSqlQuery(db);

query.exec("SELECT current\_subject\_id FROM current\_data");

query.next();

QMessageBox::warning(this,"information",QString::number(query.value(0).toInt()));

changeText(tut, query.value(0).toInt());

db.close();

ui->Text->setText(text);

DefaultValues();

}

}