Петрозаводский государственный университет
Институт математики и информационных технологий
Кафедра информатики и математического обеспечения

Направление подготовки бакалавриата

09.03.04 — Программная инженерия

Отчет по практике

Разработка приложения «MathTrainer»

Выполнила:
Хуснутдинова Айгуль Рустемовна студентка 2 курса группы 22207

А. Р. Хуснутдинова _____
подпись

Петрозаводск — 2021

Содержание

Введение

MathTrainer - приложение, позволяющее тренировать навыки устного счёта. Умение быстро производить операции над числами в уме очень полезно в современном мире. Приложение предоставит пользователю возможность тренировать свои навыки, постепенно увеличивая сложность вычислений и наблюдая за своими успехами.

Цель: проектирование и разработка приложения для тренировки навыков устного счёта

Задачи:

1. составить требования к приложению
2. спроектировать дизайн приложения
3. спроектировать архитектуру приложения
4. реализовать приложение

1 Требования к приложению

Требования к приложению с точки зрения пользователя:

1. Понятный и доступный интерфейс
2. Возможность следить за уровнем своих успехов
3. Возможность сбросить текущий результат
4. Возможность узнать лучший результат за всё время

Основные функции приложения:

1. Хранение текущего уровня сложности
2. Рассчёт текущей статистики
3. Хранение текущей статистики
4. Генерация примера на основе текущего уровня
5. Проверка результата с ответом пользователя
6. Подсчёт результата в конце каждого уровня
7. Возможность выйти из уровня в течение раунда
8. Возможность сброса текущего результата и уровня
9. Отображение текущих уровня и результата
10. Запрос подтверждения важных действий (сброс уровня, сброс текущего раунда)

2 Проектирование приложения

1. override fun onCreate(savedInstanceState: Bundle?) - функция запуска MainActivity
    - обработчик кнопки старта / перехода к следующему вопросу
    - обработчик кнопки выхода из раунда / сброса уровней
2. private fun resetScore() - функция сброса счёта и уровня
3. private fun load() - функция для отображения текущего счёта и уровня
4. private fun generateTask() : Int - функция генерации и отображения примера на основе уровня и возврата ожидаемого ответа
5. private fun checkAnswer(expected: Int) : Int - функция проверки ответа пользователя. Принимает правильный ожидаемый ответ, возвращает количество прибавляемых очков к результату текущего раунда
6. private fun showResults(results : Int) - функция для отображения результата в конце раунда, получает на вход количество набранных очков за раунд
7. private fun showImage(res: Int) - функция для отображения мотивирующей картинки в соответствии с результатами за раунд
8. private fun showStart() - функция меняет названия кнопок на режим вне раунда
9. private fun showGame() - функция меняет назввания кнопок для раунда
10. private fun showTask(a: Int, b: Int, sgn: Char) - функция отображает пример для пользователя

3 Реализация приложения

Для работы приложения функции получают и возвращают необходимые значения, которые хранятся в переменных, объявленных в главной функции. Для хранения уровня, текущего счёта и лучшего результата используется метод SharedPreferences.

При старте приложения пользователь видит статистику и кнопки старта игры и сброса счёта. Статистика загружается из хранилища и отображается в текстовом виде. После старта игры приложение меняет значение переменной, обозначая, что в данный момент происходит раунд и меняет названия кнопок.

В раунде пять примеров, каждый из которых генерируется произвольным образом. Каждые четыре уровня повышается на единицу разрядность чисел примера. В начале пользователю предлагается практиковать навыки сложения. На следующих уровнях добавляются вычитание, умножение и деление. Всего уровней 20.

Счёт хранится как количество всех правильных ответов. При отображении и обновлении счёт счиается как процент от всех примеров, таким образом максимальный счёт в игре - 100, минимальный - 0.

При нажатии на кнопки, которые есть в приложении, издаются характерные звуки. Окончание рауна с результатами также сопровождается звуковыми эффектами.

Для разработки приложения использовался язык Kotlin и среда разработки Android Studio, для реализации исаользовались следующие библиотеки:

- android.content.Context
- androidx.appcompat.app.AppCompatActivity
- android.os.Bundle
- android.util.Log
- android.widget.*
- androidx.appcompat.app.AlertDialog

Оценка сложности разработки:

- Число модулей: 1
- Число функций: 10

- Число исходных файлов: 1

Разработка интерфейса:

Для разработки интерфейса использовались методы Android Studio и языка Kotlin.

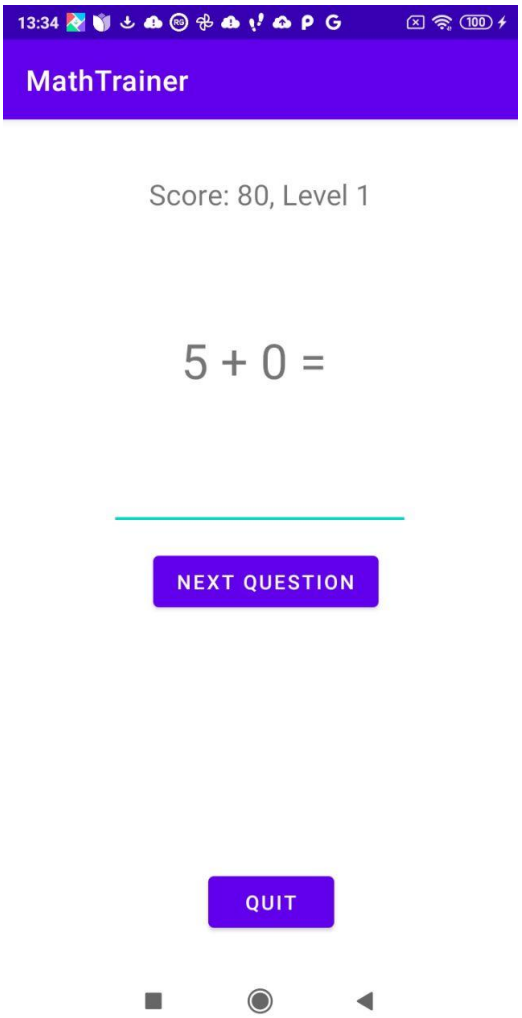

Рис. 1: Окно интерфейса

Заключение

Таким образом, в ходе разработки приложения, мы реализовали его основные функции: генерация примеров, их проверка, ведение и сохранение счёта и динамика уровней. Также была реализована возможность выхода их раунда, сброс счёта и уровней с подтверждением действия. Были выполнены основные требования к приложению:

- Интерфейс примитивен и будет понятен пользователю, а также подтверждает важные действия в отношении сохранения и удаления данных.
- После каждого раунда пользователь получает текущую статистику, а также видит общий счёт на основе всех его предыдущих результатов.
- Пользователь может сбросить прогресс текущей попытки и начать её с первого уровня, не сохраняя предыдущий прогресс
- После завершения последнего уровня, приложение сохраняет лучший результат за все прохождения, а также отображает проследний лучший.