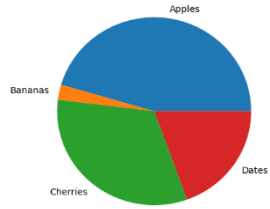
```
y = np.array([35, 2, 25, 15])
mylabels = ["Apples", "Bananas", "Cherries", "Dates"]
plt.pie(y, labels = mylabels)
plt.show()
```



Пайдаланылған әдебиеттер тізімі

1. «Информатика негіздері» ғылыми-әдістемелік журнал. – Алматы, №1, 2010 ж.
2. «Информатика негіздері» ғылыми-әдістемелік журнал. – Алматы, №5, 2011 ж.
3. Свейгарт Эл С24 Python. Чистый код для продолжающих. — СПб.: Питер, 2022. — 384 с.: ил. — (Серия «Библиотека программиста»)
4. Лутц, Марк. Л86 Изучаем Python, том 2, 5-е изд. : Пер. с англ. — СПб. : ООО “Диалектика”, 2020. — 720 с. : ил. — Парал. тит. англ.
5. http://ru .wikipedia.org/wiki/
6. https://www.w3schools.com/python/matplotlib_intro.asp
7. https://rus-linux.net/MyLDP/soft/An_Introduction_to_Matplotlib.html
8. https://practicum.yandex.ru/blog/biblioteki-python-dlya-data-science/

Гузенко Н.А., студент
Нурпеисова Ж.С., старший преподаватель
Костанайский региональный университет им. А. Байтурсынова

МОБИЛЬНОЕ ANDROID ПРИЛОЖЕНИЕ «MEMORY»

Программирование под Android остаётся высоко востребованным из-за огромной пользовательской базы, постоянного роста мобильного рынка и поддержки со стороны Google. Это открывает разработчикам широкие возможности для карьеры и монетизации приложений.

Игра "Memory" развивает визуальную память и внимательность, поскольку игрокам необходимо запоминать расположение и изображения на карточках для успешного нахождения всех пар. Игра подходит для любых возрастных групп и может играться как индивидуально, так и с друзьями для сравнения результатов

В этой игре игроку представлено поле с 20 перевёрнутыми карточками. Каждая карточка имеет пару с идентичным изображением, и все карточки расположены в случайном порядке. Задача игрока - находить пары одинаковых картинок. Игровой процесс следующий:

1. Игрок нажимает на одну из перевёрнутых карточек, чтобы перевернуть её и увидеть картинку.

2. Затем игрок выбирает вторую карточку, чтобы перевернуть и показать её изображение.

3. Если изображения на обеих карточках совпадают, они остаются открытыми на поле. Это означает, что игрок успешно нашёл пару.

4. Если изображения не совпадают, обе карточки снова переворачиваются лицевой стороной вниз, и игрок должен попытаться запомнить их расположение, чтобы использовать эту информацию в последующих ходах.

Цель игры - найти все пары и очистить игровое поле от карточек.

В данном проекте реализованы следующие правила:

1. На нахождение всех пар дается 39 секунд.

2. Если найдены все пары до окончания времени, награда начисляется по следующей формуле: *if* (timeHas >= 20) 100 *else* (100 - (20 - timeHas) * 5). Где timeHas — сколько секунд осталось.

Проект написан на основе архитектуры Single Activity с MVVM (Model — View — ViewModel). Приложение состоит из 3 пакетов и одного класса Diamond.

Diamond — класс, который хранит в себе ссылку на изображение. Также у класса есть объект-компаньон, в нем реализована функция, которая возвращает массив со ссылками на изображения.

3 пакета: *model, view, viewmodel.*

Model — реализовано 3 класса:

1) DiamondForAdapter — класс, который описывает карточку, нужен для работы с RecyclerAdapter.

2) RecyclerAdapter — классический класс адаптер, для работы и отображения RecyclerView, в нем реализована suspend функция, которая проверяет, являются ли две карточки одинаковыми, если да, то они остаются открытыми и в фрагмент GameScene увеличивает счетчик на 1, если нет, то они переворачиваются обратно, также реализована простая анимация вращения карточки с отображением самой картинки.

3) SharedPreferencesPoints — класс, в котором реализована инициализация SharedPreferences и две функции: получение игровой валюты, сохранение игровой валюты.

View — реализовано 4 класса (во всех классах и фрагментах сначала инициализируется binding):

1) MainActivity — единственное activity, получаем из класса viewModelActivity (из пакета viewmodel) количество игровой валюты, далее инициализируем фрагмент MenuView и передаем в него значение игровой валюты, если не удалось получить, по умолчанию будет значение «-1».

2) MenuView — фрагмент, главное меню приложения, из основного здесь 2 кнопки. Кнопка license вызывает Toast: «Created by Nikita Guzenko». Кнопка btnStartGame запускает следующий фрагмент с самой игрой, в качестве аргумента передается количество игровой валюты.

3) GameScene — фрагмент, сама игра. Реализован CountDownTimer (таймер), переопределен его метод onFinish() - по окончанию запускается фрагмент EndGame, который требует два аргумента — количество нынешней игровой валюты и сколько пользователь получил по формуле. Также есть наблюдатель (observe) переменной counterOfPairs — количество найденных пар, если найдены все пары — вычисляется награда и также запускает фрагмент EndGame.

4) EndGame — фрагмент, конечный экран с поздравлениями. Всего 2 кнопки: btnDoubleReward — удваивает награду, btnHome — сохраняет итоговое количество монет и запускает фрагмент MenuView с аргументов количество монет.

Viewmodel — реализован 1 класс:

1) MainActivityViewModel — класс ViewModel, всего написано 2 функции: получить игровую валюту и сохранить ее.

Из использованных материалов:

✓ Картинки SVG — icons8.ru,

✓ Библиотеки — lifecycle-viewmodel-ktx, lifecycle-runtime, lifecycle-livedata

Интерфейс программы представлен на рисунках 1-3.

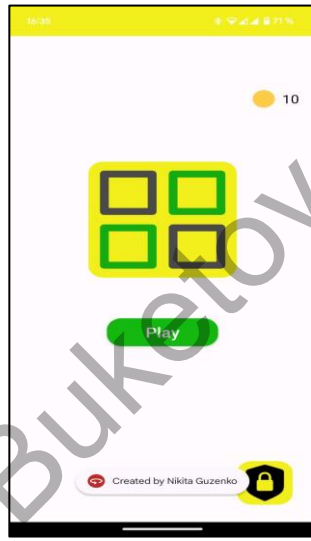


Рисунок 1. Главное меню

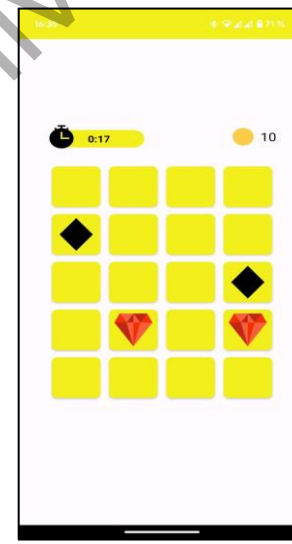


Рисунок 2. Экран игры

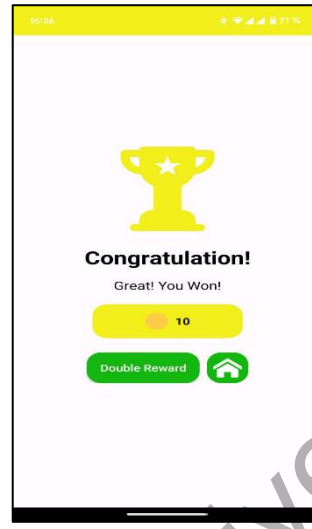


Рисунок 3. Экран окончания игры

Этапы разработки можно увидеть по разным push'ам на github.

Литература:

1. https://habr.com/ru/articles/333890/ - LiveData
2. https://habr.com/ru/companies/otus/articles/766774/Coroutine
3. https://habr.com/ru/articles/705064/ - RecyclerView
4. https://habr.com/ru/articles/207036/ - Fragment

Жарасов Y.A., студент
Мухаметжанова Б.О., Phd, старший преподаватель
Карагандинский Технический Университет им. А.Сагинова

ИССЛЕДОВАНИЕ И РАЗРАБОТКА СИСТЕМЫ УПРАВЛЕНИЯ ПРОЦЕССОМ СОРТИРОВКИ ПРОДУКЦИИ НА ОСНОВЕ НЕЙРОННОЙ СЕТИ

Введение. В современном производстве слияние передовых