



POLITECHNIKA ŚLĄSKA
WYDZIAŁ INŻYNIERII BIOMEDYCZNEJ

Projekt inżynierski

Aplikacja do testowania pamięci krótkotrwałej

Autor: Kamil Olesz

Kierujący pracą: Dr inż. Jacek Kawa

Zabrze, 29 grudnia 2017

Spis treści

1. Wstęp	1
2. Projekt aplikacji	3
2.1 Założenia wstępne	3
2.2 Opcjonalne funkcje	3
2.3 Metodologia	4
3. Specyfikacja wewnętrzna	5
3.1 Wprowadzenie	5
3.2 Interfejs użytkownika	6
3.2.1 activity_menu	6
3.2.2 activity_numbers	7
3.2.3 activity_options	8
3.2.4 activity_results	8
3.2.5 list_row	9
3.2.6 frame	9
3.2.7 gradient	9
3.3 Klasy i metody	10
3.3.1 Menu	10
3.3.2 Numbers	10
3.3.3 Tutorial	12
3.3.4 Options	13
3.3.5 Results	14
3.3.6 MultirowLVAdapter	14
4. Specyfikacja zewnętrzna	15
4.1 Menu główne	15
4.2 Rozgrywka	16
4.3 Samouczek	21
4.4 Opcje	25
4.5 Wyniki	26
5. Testy	27
6. Podsumowanie i wnioski	29

<i>Bibliografia</i>	30
-------------------------------	----

1. Wstęp

W dzisiejszych czasach nauka rozwija się w zawrotnym tempie. Mimo tego, zagadnienie działania ludzkiego mózgu wciąż pozostaje nie do końca rozwiązane [3]. Jedną z fundamentalnych umiejętności mózgu, bez której człowiek nie mógłby funkcjonować, jest pamięć. Odpowiada ona za wszystkie zdolności, których nauczył się człowiek przez całe swoje życie. Bez pamięci, wszystkie wrażenia, które ludzie doznają, zacierająby się i to, co człowiek odczuł, po chwili stałoby się dla niego nieistniejące [3, 4].

Za zdolność zapamiętywania odpowiedzialne są neurony, a dokładnie ich całe sieci połączeń. Przy doświadczeniu jakiegokolwiek bodźca, komórki nerwowe łączą się ze sobą. Takie połączenie zwane jest synapsą – może być mocniejsze lub słabsze, co wpływa na późniejsze utrzymanie tegoż wiązania. Po każdorazowym odebraniu tego samego wrażenia, synapsa wzmacnia się i pozostaje nieprzerwana. Przy jednorazowym, krótkim odebraniu bodźca, synapsa szybko zanika gdyż połączenie między komórkami jest zbyt słabe [5, 7]. Mózg sam potrafi ustalić ważność pobudzenia, zatem wybiórczo pozostawia rzeczy w swojej pamięci. Tak dla przykładu, pierwsze poparzenie się ogniem „uczy” człowieka, by ten starał się unikać kolejnych poparzeń. Wrażenie bólu jest bowiem ważnym aspektem dla mózgu. Z drugiej strony, mało ważną informacją dla człowieka jest wygląd mijającej go na chodniku osoby. Mózg przyswaja ten wygląd, lecz po chwili może dojść do wniosku, że nie jest to istotna informacja i połączenie neuronów się zerwie. Wszystkie operacje wiązań dzieją się w strukturze mózgu zwanej hipokampem. [1, 2].

Najbardziej popularnym podziałem pamięci jest podział ze względu na czas trwania pamiętania – ultrakrótkotrwała (zwana również sensoryczną), krótkotrwała oraz długotrwała. W dalszej części pracy uwaga zostanie skupiona tylko na pamięci krótkotrwałej [4].

Pamięć krótkotrwała jest wykorzystywana przez ludzi codziennie, przy zwykłych domowych czynnościach, do których nie przywiązuje się zbyt dużej wagi, np. przy zapamiętaniu czy herbata została już posłodzona albo drzwi wejściowe zostały zamknięte. Warto ćwiczyć tę umiejętność, by sprawniej wykonywać codzienne czynności (również te w pracy) oraz lepiej przyswajać informacje świata nas otaczającego. Trening pamięci krótkotrwałej ma również wpływ na zdolność pamiętania długoterminowego. Więcej neuronów jest wykorzystywanych w obrębie ośrodka pamięci, przez co człowiek jest w stanie zapamiętać więcej rzeczy. Pamiętając pewne informacje dłużej, łatwiej jest przenieść je z obszaru pamięci krótkotrwałej na długotrwałą [3–5].

Istnieje wiele sposobów testowania ludzkiej pamięci, a najczęściej odnosi się do badania pamięci krótkotrwałej. Przeprowadzenie takiego testu wiąże się z prostym zadaniem zapamiętania pewnych rzeczy, np. figur geometrycznych, par wyrazów, krótkich słów

czy ciągów cyfr. Prezentowana w pracy aplikacja, będzie odnosiła się do ostatniego testu [6].

Trening pamięci polega na regularnym powtarzaniu danych testów. Głównie warto skupić się na jednym, gdyż powtarzane czynności są bardziej przyswajane przez mózg. Powtarzanie ciągu cyfr jest uniwersalne dla każdej osoby, w końcu każdy zna cyfry, a bardzo łatwo przy tym można określić jak długi ciąg zapamiętuje dana osoba [4, 6].

Jest mnóstwo form trenowania (przy tym i testowania) pamięci [6]. Najbardziej odpowiednią w dzisiejszych czasach są wszelkie aplikacje mobilne. Na sam trening pamięci konieczna jest tylko chwila czasu, ale najważniejsza jest regularność, bowiem efekty widoczne są dopiero po długim okresie. Łatwo jest włączyć aplikację mobilną, korzystać z niej przez określony czas po czym bez żadnego opóźnienia wrócić do codziennych obowiązków. Treningi takie można wykonywać w każdych wolnych chwilach, chociażby przy podróży autobusem. To daje ogromną wygodę i przez to, coraz więcej ludzi decyduje się na trenowanie swojej pamięci – dlatego też występuje popyt na tego rodzaju oprogramowania.

Dostępne na rynku aplikacje ^{1 2 3} oferują przerost formy nad treścią. Twórcy skupiają uwagę na przyciągnięciu potencjalnych klientów swoim wyglądem. Sama forma aplikacji może w znaczący sposób dekoncentrować trenującego (np. zmiana kolorów ekranu, zbyt długie animacje czy migające obrazki). Każdy oczekuje od aplikacji czegoś innego, tak więc przydatna byłaby możliwość personalizacji, lub ewentualnie możliwość łatwej edycji programu i sprzedaż spersonalizowanych form.

Celem opisanej pracy jest zaprojektowanie i stworzenie aplikacji mobilnej, umożliwiającej prosty test pamięci krótkotrwałej. Dodatkowo, aplikacja powinna mieć formę pozwalającą na trening tego rodzaju pamięci. Program ma być łatwy w korzystaniu przez każdego użytkownika oraz spełniać jego potrzeby. Docelowe grupy osób to:

- osoby zdrowe, nie trenujące dotychczas pamięci,
- osoby zdrowe, trenujące regularnie pamięć krótkotrwałą,
- osoby chore na wszelkie ubytki pamięci – korzystanie w celach diagnostycznych i terapeutycznych,
- osoby młodsze – forma na tyle prosta by mogła korzystać z niej jak najmłodsza osoba
- osoby starsze – podobnie jak dla osób młodszych, forma na tyle prosta by bez problemów mogły korzystać z aplikacji starsze osoby.

Realizacja celu wymagać będzie następujących kroków: projekt, implementacja i testowanie. Zostaną one opisane w dalszej części pracy.

¹ Brainwell — Brain Training Games By Monclarity, LLC – App Store, 17.12.2017

² Lumosity – Trening mózgu, Lumos Labs, Inc. – Google Play, 17.12.2017

³ Gry na pamięć: Trening mózgu, Maple Media – Google Play, 17.12.2017

2. Projekt aplikacji

2.1 Założenia wstępne

Program musi zawierać prosty test pamięci krótkotrwałej, jakim jest zapamiętanie losowo generowanego ciągu cyfr. Test musi zostać zaimplementowany w aplikacji na urządzenia mobilne. Da to możliwości sprawdzenia pamięci w każdej chwili praktycznie każdej osobie. Dodatkowo, aplikacja powinna mieć regulowaną długość ciągu, co umożliwi test pamięci według umiejętności użytkownika.

Aplikacja ma być stworzona w formie programu umożliwiającego również trening pamięci. Dlatego poziom trudności powinien rosnąć wraz z umiejętnościami użytkownika. Powinien istnieć również warunek zakończenia testu czy treningu. Nie powinno być żadnego ograniczenia w związku z maksymalną czy minimalną długością ciągu. W końcu ktoś może mieć trudności z zapamiętaniem już jednej cyfry. Z drugiej strony, użytkownik może mieć ponadprzeciętną pamięć a zadanie maksymalnej długości ciągu może spowodować, że nie będzie sensu dalszego treningu pamięci, ponieważ bez większego wysiłku umysłowego byłby w stanie zapamiętać maksymalnie długi ciąg. Oczywiście muszą pojawiać się komunikaty informujące użytkownika o poprawnym lub błędnym zapamiętaniu wygenerowanego ciągu.

2.2 Opcjonalne funkcje

Dodatkową funkcją oprogramowania może być wprowadzanie głosowe zapamiętanego ciągu. Umożliwi to korzystanie z aplikacji osobom starszym, dla których problemem byłoby szybkie wpisanie z klawiatury numerycznej. Kolejnym plusem takiego rozwiązania, byłoby umożliwienie testowania pamięci krótkotrwałej osobom z dysfunkcjami ruchowymi, np. po udarze. Opcja wprowadzania głosowego byłaby również formą personalizacji. Dla pewnych osób wygodniejsze mogłoby być wypowiedzenie zapamiętanego ciągu niż wpisanie go.

Kolejną opcjonalnością mogłoby być zapamiętanie danych wyników. Z racji treningu pamięci, przydatna byłaby możliwość sprawdzenia i porównania jak długi ciąg użytkownik zapamiętał miesiąc temu, a jakie wyniki osiąga teraz. Zapisywanie wyników mogłoby odbywać się w formie zapisu do bazy danych, jednak wiązałoby się to z koniecznością dostępu do internetu podczas korzystania z aplikacji. Alternatywą tego rozwiązania byłby zapis wyników do pamięci urządzenia mobilnego. Wiazałoby się to jednak z koniecznością wysłania wyników do dogłębnej analizy chociażby przez lekarzy.

2.3 Metodologia

W celu wypełnienia wymienionych założeń, aplikacja zostanie zaprogramowana w języku Java w środowisku Android Studio. Umożliwia to stworzenie aplikacji na urządzenia mobilne z systemem android, czyli najbardziej popularnym systemem wśród obywateli Polski¹.

Aplikacja będzie posiadała ustawienia, umożliwiające wybranie początkowej długości ciągu, zadeklarowanie liczby szans – warunku kończącego rozgrywkę oraz wybór trybu wpisywania głosowego (zamień tekst lub dodaj do tekstu). Kolejnym dodatkowym aspektem będzie zapisywanie wyniku najdłuższego zapamiętanego ciągu cyfr w danej rozgrywce wraz z zapisem czasu zapamiętywania tej sekwencji.

Dla przejrzystości rozgrywki, program będzie posiadał samouczek, dostępny do uruchomienia z pozycji menu głównego. Wprowadzi on w panujące w trakcie testu zasady – co należy zrobić w danym etapie rozgrywki. Zaznajomi również użytkownika z funkcją wprowadzania głosowego.

Test pamięci będzie przebiegał według następującego schematu:

1. Rozpoczęcie testu po uprzednim ustawieniu początkowej długości ciągu.
2. Wygenerowanie losowego ciągu cyfr od 0 do 9.
3. Zapamiętanie wyświetlonego ciągu przez użytkownika.
4. Wprowadzenie zapamiętanej sekwencji.
5. Sprawdzenie poprawności zapamiętania.
6. Jeżeli warunek końca testu nie został spełniony - przejście do punktu drugiego.
7. Jeżeli warunek został spełniony - zakończenie rozgrywki.

¹ Według statcounter GlobalStats, 18.12.2017

3. Specyfikacja wewnętrzna

3.1 Wprowadzenie

Aplikacja została napisana w środowisku Android Studio. Klasy są napisane w języku Java a pliki layoutu mają rozszerzenie xml. Minimalna wspierana wersja API to 16, co odpowiada wersji 4.1 Jelly Bean systemu operacyjnego Android. Według Android Studio, daje to możliwość uruchomienia stworzonej aplikacji na, w przybliżeniu, 99.2% urządzeń aktywnych na Google Play Store, co daje możliwość testowania pamięci na praktycznie każdym urządzeniu mobilnym z systemem Android.

Aplikacja korzysta z jednej zewnętrznej biblioteki, a mianowicie biblioteki opencsv, pozwalającej na zapis i odczyt danych do/z plików o rozszerzeniu CSV (comma-separated values). Reszta wymaganych bibliotek dostarcza samo środowisko. Biblioteka opencsv jest biblioteką typ open source, możliwą do pobrania na stronie opencsv.sourceforge.net. Przy programowaniu zostały użyte następujące klasy i metody tej biblioteki:

- CSVWriter - klasa umożliwiająca zapisanie danych tablicowych do pliku typu CSV. Jako argumenty obiektu tej klasy podawany jest obiekt klasy FileWriter (przyjmuje ścieżkę oraz nazwę pliku, służy do ogólnego zapisywania plików) oraz znak separujący. Wykorzystywane metody to:
 1. Metoda writeNext, zapisująca kolejny wers do pliku. Wers jest podawany w formie tablicy z danymi typu String.
 2. Metoda close, zapisująca i zamykająca utworzony lub modyfikowany plik.
- CSVReader - klasa umożliwiająca odczyt z pliku CSV. Jako argumenty obiektu tej klasy podawany jest obiekt klasy FileWriter oraz znak separujący. Wykorzystywane metody to:
 1. Metoda readAll, odczytująca wszystkie wersy z danego pliku CSV. Zapisuje każdą komórkę do odpowiedniego miejsca w liście typu String tablicowy.
 2. Metoda close, zamykająca otwarty wcześniej plik CSV.

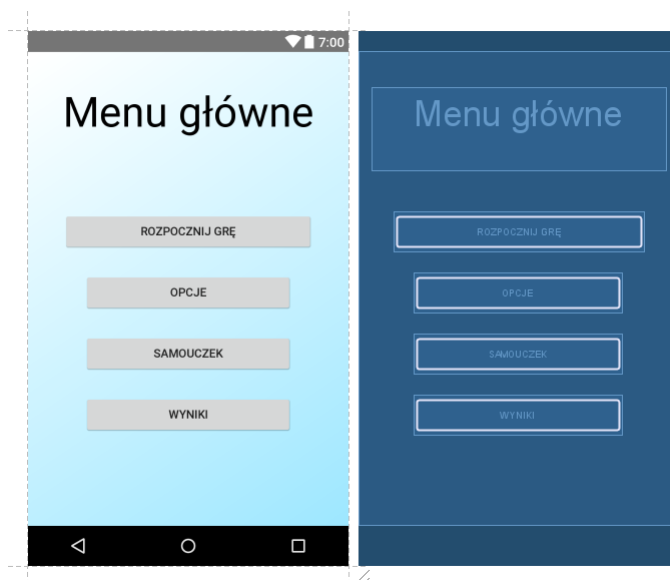
3.2 Interfejs użytkownika

Interfejs graficzny użytkownika tworzą następujące pliki:

- activity_menu.xml,
- activity_numbers.xml,
- activity_options.xml,
- activity_results.xml,
- list_row.xml,
- frame.xml,
- gradient.xml.

3.2.1 activity_menu

Jest to widok ukazywany od razu przy uruchomieniu aplikacji (Rys. 3.1). Składa się z czterech przycisków i jednego pola tekstowego. Każdy przycisk odpowiedzialny jest za przejście do innej aktywności, ukazanej w nazwie przycisku. Pole tekstowe informuje użytkownika, że znajduje się w menu głównym.



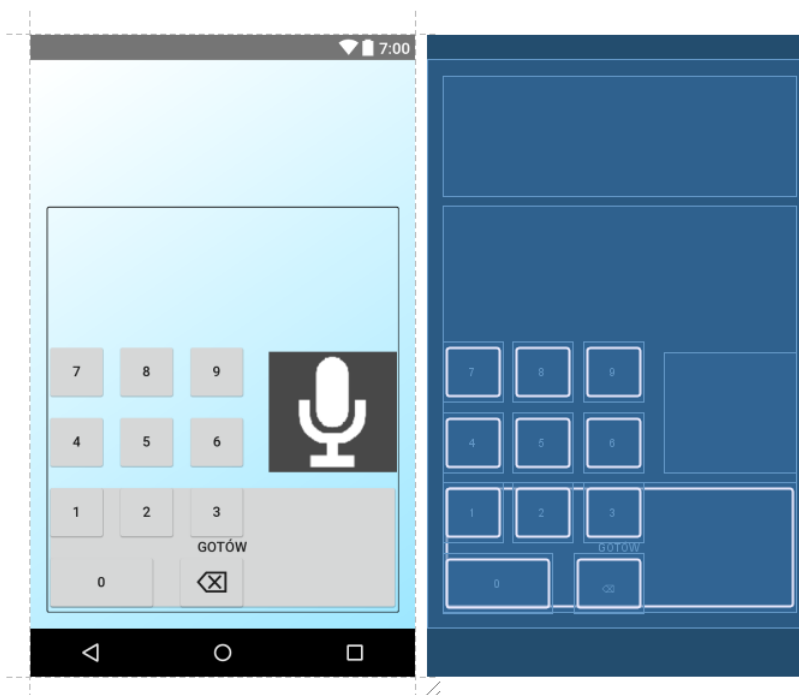
Rys. 3.1: Projekt wyglądu menu głównego aplikacji.

3.2.2 activity_numbers

Widok pokazujący wygląd całego testu pamięci (Rys. 3.2). Zawiera 10 przycisków odpowiedzialnych za wpisywanie cyfr. Są ułożone w wygodny do użytkowania sposób. Przyciski te są oznaczone odpowiednim symbolem, przez co użytkownik wie, jaka cyfra pojawi się po naciśnięciu na daną kontrolkę. Do usuwania wpisywanego przez omówione elementy ciągu, służy przycisk z wizerunkiem standardowego klawisza do usuwania tekstu w klawiaturach systemu Android. Kolejną kontrolką jest przycisk z napisem „GOTÓW”. Służy on do wyrażenia gotowości i przejścia do następnego etapu rozgrywki. ImageButton, z wizerunkiem mikrofonu, służy do włączenia wprowadzania głosowego.

Widok zawiera dwa pola tekstowe. Pierwsze, na samej górze, służy do informowania użytkownika o aktualnym etapie testu. W drugim, poniżej, ukazuje się ciąg do zapamiętania, wpisywana przez użytkownika zapamiętana sekwencja oraz statystyki po zakończeniu rozgrywki. Dodatkowo, dookoła drugiego pola tekstowego znajduje się obramowanie. Wygląd ramki został utworzony w pliku frame.xml opisanym w rozdziale 3.2.6.

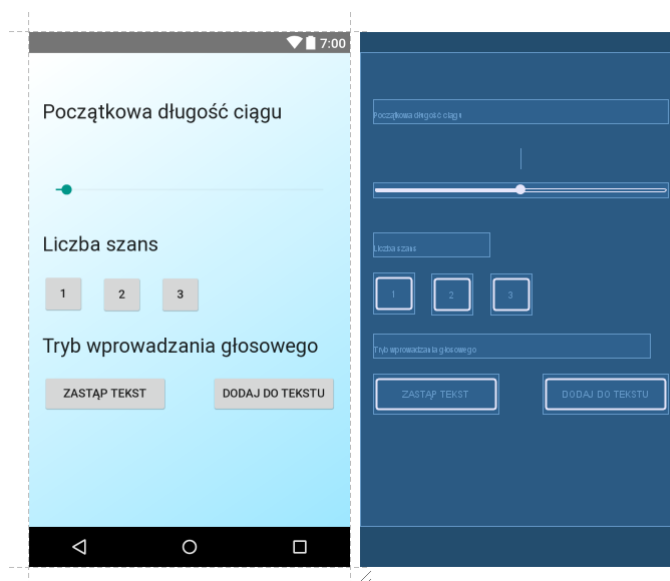
W projekcie widoku niektóre elementy nakładają się na siebie, jednakże podczas testu, kontrolki zostały tak zaprogramowane, że zmieniają swoje położenie w zależności od danego etapu rozgrywki. Wszystko jest zaprojektowane w taki sposób, by było jak najbardziej czytelne i intuicyjne dla użytkownika.



Rys. 3.2: Projekt wyglądu okna rozgrywki.

3.2.3 activity_options

Okno opcji pokazuje dostępną możliwość konfiguracji (Rys. 3.3). Trzy pola tekstowe informują użytkownika jakie ustawienia modyfikuje. Znajdujące się przyciski pokazują swoim tekstem skutek zmiany danej opcji. Przesunięcie suwaka zmienia początkową długość ciągu. Jest on przesuwalny w wartościach od 0 do 49, jednak do wybranej wartości suwaka dodawana jest jedynka. Da to możliwość ustawienia początkowej długości ciągu w zakresie od 1 do 50. Pole tekstowe, znajdujące się nad suwakiem, wyświetla zadaną wartość początkowej długości sekwencji.

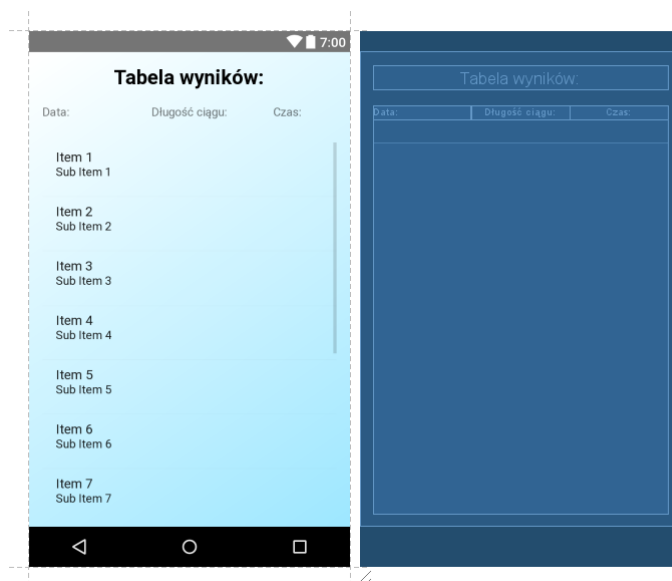


Rys. 3.3: Projekt wyglądu menu opcji.

3.2.4 activity_results

Jest to plik widoku, odpowiedzialny za wyświetlenie wyników ukończonych testów. Informuje o tym pole tekstowe, znajdujące się na samej górze okna. Poniżej znajduje się liniowy układ horyzontalny, w skład którego wchodzi trzy pola tekstowe. Informują one o typie wyświetlanej pod tekstem pozycji.

W celu wyświetlenia samych wyników, została dodana kontrolka ListView. Lista jest przewijalna i nieograniczona w ilości pokazywanych elementów. W celu prawidłowego wyświetlania żądanych wyników, został dodany plik formatujący pojedynczy wiersz listy. Został on opisany w rozdziale 3.2.5.



Rys. 3.4: Projekt wyglądu listy wyników.

3.2.5 list_row

Układ ustawiający wygląd pojedynczego wierszu listy z wynikami. Jeden wiersz składa się z trzech pól – data wykonania testu, długość zapamiętanego ciągu oraz czas zapamiętywania najdłuższej, poprawnie odtworzonej sekwencji. Pole z datą jest odrobinę dłuższe od pozostałych pól. Zapobiega to zawijaniu się tekstu w danym wierszu.

3.2.6 frame

Plik zawiera styl ramki wokół pola tekstowego wyświetlającego np. ciąg cyfr do zapamiętania. Ramka jest koloru czarnego, ma lekko zaokrąglone rogi, a jej szerokość wynosi 1 dp (density independent pixel). Tło wnętrza ramki zostało ustawione na przezroczyste, aby nie zasłaniało tła całego okna.

3.2.7 gradient

Odpowiedzialny jest za tło każdego okna aplikacji. Środowisko Android Studio umożliwia w łatwy sposób utworzenie gradientu o zadanych parametrach. Tak wygenerowany gradient należy załączyć do każdego pliku z widokiem, gdzie przeskaluje się na całe okno lub inny żądany wymiar.

3.3 Klasy i metody

Kod źródłowy aplikacji składa się z następujących klas:

- Menu.java,
- Numbers.java,
- Tutorial.java,
- Options.java,
- Results.java,
- MultirowLVAdapter.java.

Ważnym plikiem jest również plik `AndroidManifest.xml`, w którym zawarte są pozwolenia na zapis i odczyt danych z pamięci urządzenia. Plik ten zawiera również ustawienie początkowej aktywności, uruchamianej zaraz po włączeniu aplikacji. Dodatkowo określona jest orientacja ekranu każdego z widoków na „portrait”, co odpowiada pionowemu ułożeniu telefonu.

3.3.1 Menu

Klasa obsługująca akcje wykonywane w menu głównym aplikacji. Pobiera kontrolki z widoku `activity_menu` (Opisany w rozdziale 3.2.1). Do przycisków przyporządkowana jest akcja rozpoczęcia odpowiadającej przyciskowi aktywności. Tak dla przykładu, po naciśnięciu klawisza z napisem „Samouczek”, poprzez metodę `startActivity` (metoda środowiska Android Studio), zostaje uruchomiona aktywność odpowiedzialna za samouczek.

W tej aktywności, poprzez metodę `verifyStoragePermissions`, wysyłane jest żądanie o udzielenie pozwolenia na zapis i odczyt plików w pamięci urządzenia. Na początku sprawdzane jest czy aplikacja już nie ma przyznanego pozwolenia. Jeżeli nie ma, wysyłane jest zapytanie, a jeśli zezwolenie zostało przyznane, to nic nie robi. Taki zabieg umożliwia pominięcie wysłania żądania przy każdym uruchomieniu aplikacji, a nawet przy każdym wejściu do menu głównego.

Mimo, że przyzwolenia na zapis i odczyt są już przyznane w pliku `AndroidManifest.xml`, to dla nowszych wersji systemu Android (Android 6.0+), wymagane jest wyświetlenie tej informacji użytkownikowi i zapytanie go o wyrażoną zgodę.

3.3.2 Numbers

Klasa obsługująca cały przebieg testu pamięci krótkotrwałej. Pobiera on widok z pliku `activity_numbers.xml` (Opisany w rozdziale 3.2.2). Głównym elementem działania programu jest metoda `onCreate`, a dokładniej nasłuchiwanie na naciśnięcie przycisku z tagiem 'ok'. Po jego naciśnięciu, w zależności od etapu testu, uruchamiane są odpowiednie metody. Aktualny etap rozgrywki zależny jest od wartości zmiennej sterującej

'steering'. Zmienna ta przyjmuje wartości 0, 1 lub 2. Pierwsza wartość określa etap zapamiętywania ciągu cyfr przez użytkownika. W tym etapie uruchamia się pomiar czasu zapamiętywania. Generuje się również ciąg cyfr, po czym wyświetlany jest na ekranie. Przy naciśnięciu na przycisk wyrażający gotowość, pomiar czasu zapamiętywania zatrzymuje się, zmienia się wygląd okna aplikacji, oraz zmienna sterująca przyjmuje wartość 1. Ta wartość określa etap wpisywania zapamiętanej sekwencji. Po kolejnym przyciśnięciu, w zależności od tego, czy użytkownikowi zostały jeszcze szanse lub nie, zmienna sterująca przyjmuje wartości odpowiednio 0 lub 2. Wartość 2 określa zakończenie testu, gdzie po ponownym naciśnięciu na omawiany przycisk, aplikacja „wraca” do menu głównego. Przy tym etapie rozgrywki wyświetlana jest informacja o końcu gry oraz ukazywane są statystyki – najdłuższy zapamiętany ciąg oraz czas jego zapamiętywania. Jeśli użytkownikowi nie udało się zapamiętać żadnej sekwencji, aplikacja wychwyci to i poinformuje użytkownika o tym wyjątku.

Wygenerowanie losowego ciągu cyfr przebiega w metodzie `random_number`. Przyjmuje ona parametr `'length'`, który odpowiada długości wylosowanej sekwencji. Zadeklarowana zostaje zmienna tablicowa typu `int` o rozmiarze równym wartości przesłanego parametru. Następnie, do każdej komórki tejże tablicy zostaje wpisana losowa cyfra. Tak wygenerowany ciąg zostaje zwrócony do dalszych działań.

Kolejnym krokiem jest przekształcenie wygenerowanego ciągu z formy tablicy liczb całkowitych do typu `String`. Dzieje się to w metodzie `intarray2string`, która przyjmuje parametr typu `int` tablicowy. Pętla przechodzi po każdej komórce tablicy i dopisuje jej wartość do napisu. Przekonwertowany ciąg jest zwracany do dalszego użytku. Zabieg ten jest konieczny przy bezproblemowym wyświetleniu ciągu w polu tekstowym oraz łatwym porównaniu wartości wpisanej przez użytkownika z zapamiętywaną sekwencją.

Metoda sprawdzająca poprawność zapamiętanej sekwencji to metoda `check`. Przyjmuje ona dwie zmienne typu `String`. Jedna zawiera ciąg cyfr do zapamiętania, a druga wartość wpisaną przez użytkownika po zapamiętaniu. Funkcja sprawdza czy napisy są sobie zbieżne. Jeżeli tak, użytkownikowi zostaje wyświetlony stosowny komunikat. Długość sekwencji zostaje zwiększona o jeden, czas zapamiętywania zostaje zapisany a także zmienna `'remembered'` zostaje ustawiona na `true`. Zmienna ta jest typu `bool` i zawiera informacje o tym, czy użytkownik zapamiętał jakikolwiek ciąg. Do tego etapu, zmienna ta jest ustawiona na `false`. Przy negatywnym wyniku porównania, zmniejszana jest liczba szans. Jeżeli liczba szans jest większa od zera, użytkownikowi wyświetlony jest komunikat o pozostałej liczbie szans. W przeciwnym przypadku, gracz zostaje poinformowany o zakończeniu rozgrywki.

Przy ekranie końcowym zapisywany jest wynik testu do pliku CSV – dzieje się to w metodzie `SaveToCSV`. Korzysta ona z zaimportowanej biblioteki `opencsv`. Początkowo sprawdzane jest, czy plik o danej nazwie w danej lokalizacji już istnieje. Jeżeli nie, to program go tworzy. Następnie do zmiennej tablicowej zostają umieszczone dane: data zakończenia rozgrywki, długość zapamiętanego ciągu oraz czas jego zapamiętywania. Poprzez metodę biblioteki `opencsv`, `writeNext` (opisana w rozdziale 3.1), do pliku zapisywany jest kolejny wiersz z separatorem w postaci znaku `','`.

Pomiar czasu zapamiętywania odbywa się w metodzie `Timer`. Nie przyjmuje ani nie zwraca żadnych argumentów. Metoda uruchamia się automatycznie co jedną mi-

lisekundę. Po upływie 100 milisekund, do licznika sekund zostaje dodana jedynka, a po upływie minuty, licznik minut zwiększa się o jeden. Oczywiście pozostałe liczniki zerują się. Po każdorazowej zmianie czasu, informacja ta jest zapisywana do globalnej zmiennej, którą później można wykorzystać.

Przed rozpoczęciem testu pobierane są aktualne ustawienia. Odpowiedzialna za to funkcja nosi nazwę `Get_Config`. Otwiera on plik z ustawieniami i przypisuje do odpowiednich komórek tablicy typu `String` zapisane znaki, rozdzielone pionową kreską. Pobrane ustawienia to początkowa długość zapamiętywanego ciągu, liczbę szans oraz typ wprowadzania głosowego.

Sama obsługa wprowadzania głosowego zaczyna się po naciśnięciu przycisku z wizerunkiem mikrofonu, gdzie początkowo sprawdzane jest, czy ta funkcja jest w danym urządzeniu wspomagana. Jeżeli tak, uruchamia się nowa aktywność. W niej, wszystko, co powiedział użytkownik, zostaje zapisane do tablicy znaków. Następnie, każdy znak jest osobno wprowadzany do metody `isValidInteger`. Metoda ta służy przefiltrowaniu całego wprowadzonego tekstu, w celu wyciągnięcia tylko liczb całkowitych. Jeżeli wystąpił jakikolwiek inny znak, metoda zwraca fałsz i nie jest on w ogóle brany przy ukazywaniu na ekran. Po zakończonej filtracji, w zależności od wybranego trybu wprowadzania głosowego, pole tekstowe zostaje albo zamienione wprowadzonym ciągiem, albo do tekstu już wprowadzonego zostaje dopisane to, co użytkownik powiedział. Metoda jest wrażliwa na występowanie błędów – dla przykładu, jeżeli użytkownik wypowiedział tylko ciąg liter, aplikacja powiadomi go krótkim komunikatem, że wprowadzony ciąg nie został poprawnie wypowiedziany.

Metoda odpowiedzialna za zmianę wyglądu aplikacji w zależności od aktualnego etapu testu pamięci, to metoda `invisibility`. Pobiera ona zmienną typu `bool`, mówiącą o tym, czy elementy widoczne na ekranie mają się pokazać lub też stać się niewidoczne. Pierwszym etapem tej funkcji jest pobranie rozmiaru ekranu. Następnie pobiera granice przycisku wyrażającego gotowość, pola tekstowego wyświetlającego chociażby wygenerowaną sekwencję oraz przycisku z wizerunkiem mikrofonu. Służy to późniejszemu przesunięciu kontrolek. W zależności od wartości wprowadzonej do metody zmiennej, znikają lub pojawiają się przyciski odpowiedzialne za wprowadzanie zapamiętanego ciągu, kasowanie znaków oraz za wprowadzanie głosowe. Następnie przesuwany jest przycisk wyrażający gotowość oraz wspomniane wcześniej pole tekstowe, tak, aby nie nakładały się na siebie żadne elementy.

Obsługa wszystkich przycisków wprowadzających dzieje się w metodzie `ButtonClicked`. Sprawdzany jest który przycisk został wciśnięty i podejmowana jest odpowiadająca temu akcja - wprowadzenie czy kasowanie cyfry.

3.3.3 Tutorial

Jest to klasa dziedzicząca po klasie `Numbers` (opisana w rozdziale 3.3.2), a nawet korzystająca z tego samego widoku. Klasa ta ma za zadanie obsługę samouczka, uruchamianego po naciśnięciu odpowiedniego przycisku w menu głównym. Etapy samouczka określone są przez warunek wielokrotnego wyboru – `switch`. Zmienną poddaną temu warunkowi jest zmienna `'step'`, która informuje o aktualnym etapie samouczka. Po za-

kończącej jednej części, przy naciśnięciu na odpowiedni przycisk, 'step' przyjmuje inną wartość, przez co po następnym kliknięciu przejdzie do innego etapu. Cały samouczek opisany jest w rozdziale 4.3.

3.3.4 Options

Klasa obsługująca wszystkie zmiany zachodzące w ustawieniach. Korzysta z widoku `activity_options` (opisany w rozdziale 3.2.3). Na wybór opcji składają się trzy grupy - obsługa suwaka ustawiającego początkową długość ciągu, przyciski z liczbą szans oraz przyciski trybu wprowadzania głosowego.

Po jakiegokolwiek zmianie pozycji suwaka, pobiera się jej wartość, dodaje się do niej jeden i wpisuje ją do zmiennej 'length'. Również wartość ta jest wpisywana do pola tekstowego nad suwakiem w celu powiadomienia użytkownika o aktualnej wartości początkowej długości ciągu.

Wybór liczby szans jest określana w metodzie `click_chance`. Uruchamia się ona po każdorazowym kliknięciu w przycisk z określoną liczbą szans. Wartość tego przycisku wpisywana jest do zmiennej 'chances'.

Tryb wprowadzania głosowego działa na podobnej zasadzie co wybór liczby szans. kliknięcie w przycisk uruchamia metodę `click_voice`. Tam, w zależności od klikniętego przycisku, do zmiennej 'voice' przyporządkowana jest wartość 1 lub 2. Pierwsza wartość oznacza zamianę całego, widocznego tekstu po wprowadzeniu głosowym. Druga opcja pozwala użytkownikowi na dopisanie do wyświetlonego tekstu tego, co wprowadzi się głosowo.

Po każdej zmianie opcji uruchamia się metoda `Save_Config_to_File`, która przyjmuje omówione wcześniej zmienne konfiguracyjne. Metoda zapisuje wartości zmiennych do pliku tekstowego, oddzielając każdą wartość kreską pionową.

Również po każdej zmianie opcji oraz zaraz po uruchomieniu menu ustawień, uruchamiana zostaje metoda `Configure_Layout`. Pobiera ona wartości zmiennych konfiguracyjnych i na ich podstawie dostosowuje wygląd aplikacji. Zapisuje wartość początkowej długości ciągu do pola tekstowego oraz podświetla przyciski odpowiadające aktualnie wybranej opcji - liczbie szans i trybu wprowadzania głosowego.

Przy pierwszym uruchomieniu ekranu opcji, sprawdzane jest, czy plik z opcjami istnieje. Dzieje się to w metodzie `Check_if_config_exists`. Jeżeli plik konfiguracyjny nie istnieje, tworzy go, wpisując wartości odpowiadające konkretnym opcjom. Domyślnie, początkowa długość ciągu wynosi trzy, liczba szans również jest równa trzem, a tryb wprowadzania głosowego jest ustawiony na zamianę tekstu.

Aktualne ustawienia pobierane są w metodzie `Get_Config`. Dzieje się to zaraz po sprawdzeniu czy plik konfiguracyjny istnieje. Z pliku zostają odczytane i przyporządkowane do zmiennych odpowiednie wartości. Zabieg ten jest konieczny, by zapamiętywać aktualne ustawienia, w celu uniknięcia każdorazowego resetu konfiguracji przy uruchomieniu opcji.

3.3.5 Results

Klasa obsługująca wyświetlenie listy wyników na ekranie. W metodzie `ReadFromCSV`, korzystającej z biblioteki `opencsv`, opisanej w rozdziale 3.1, dochodzi do wczytania listy wyników z pliku `csv`. Początkowo zostaje pobrana nazwa pliku i jego ścieżka. Następnie, dzięki metodzie biblioteki `opencsv` – `readAll`, wczytane są wszystkie wiersze, i kolumny oraz zostają one dopisane do listy typu `String` tablicowy.

Pobrana lista jest wyświetlana na ekranie dzięki funkcji `setAdapter`. Jest to funkcja środowiska `Android Studio`, pozwalająca na dowolne formatowanie wyświetlanej listy według wcześniej ustalonego widoku. Przyporządkowanie adaptera dzieje się w klasie `MultirowLVAdapter`, opisanej poniżej.

3.3.6 MultirowLVAdapter

Skrypt ten odpowiada za formatowanie wyświetlanej listy. Zawiera kod wygenerowany automatycznie przez środowisko `Android Studio` wraz z modyfikacjami. Przeciążona metoda `getView` pobiera układ z pliku `list_row.xml` (opisany w rozdziale 3.2.5). Następnie, wprowadzona lista z danymi zostaje `zmapowana` (każda komórka listy ma swoją pozycję), aby w końcowym etapie można było przyporządkować każdy element listy do odpowiedniej komórki.

4. Specyfikacja zewnętrzna

Rozdział ten pełni funkcję instrukcji użytkownika. Poniżej wyjaśnione są wszystkie elementy związane z użytkowaniem aplikacji, począwszy od menu głównego, przechodząc przez test pamięci krótkotrwałej wraz z samouczkiem, a kończąc na ekranie opcji i zapisanych wynikach.

4.1 Menu główne

Menu główne aplikacji służy do szybkiego przejścia między poszczególnymi opcjami. Wygląd menu jest widoczny na rysunku 4.1. Poprzez kliknięcie w odpowiednie przyciski można rozpocząć grę, zmienić ustawienia aplikacji, uruchomić samouczek lub zobaczyć tabelę wyników.

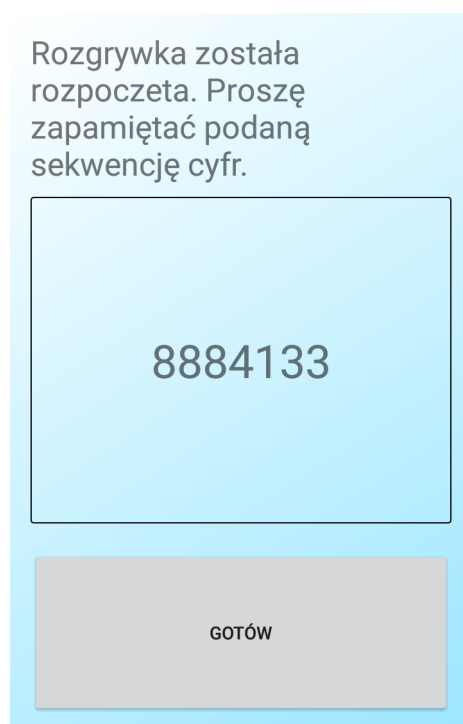


Rys. 4.1: Menu główne aplikacji.

4.2 Rozgrywka

Test pamięci krótkotrwałej rozpoczyna się po wciśnięciu pierwszego przycisku w menu głównym. Od razu po uruchomieniu wyświetla się ciąg cyfr do zapamiętania przez użytkownika. (Rys. 4.2). Po wyrażeniu gotowości, przechodzi się do następnego etapu, którym jest wpisanie zapamiętanej sekwencji (Rys 4.3). Gdy użytkownik stwierdzi, że odtworzył zapamiętany ciąg (Rys. 4.4), musi przejść do kolejnej części, naciskając odpowiedni przycisk. Tam dowie się, czy sekwencja została odtworzona poprawnie (Rys. 4.5), czy też nie (Rys. 4.6). Przy udanej próbie, wyświetli się kolejny ciąg do zapamiętania. Tym razem będzie on dłuższy od poprzedniego o jedną cyfrę. Niepowodzenie skutkuje utratą szansy, jednakże zapamiętywana sekwencja będzie tej samej długości. Oczywiście, ciąg do zapamiętania za każdym razem będzie inny, losowy.

Warunkiem zakończenia testu jest utrata wszystkich szans. Na ekranie końcowym wyświetla się informacja o końcu rozgrywki, a poniżej ukazują się statystyki związane z testem – długość najdłuższego zapamiętanego ciągu oraz czas, w którym został zapamiętany (Rys. 4.7). Statystyki zostają od razu zapisane do pliku znajdującego się w pamięci urządzenia. Po kliknięciu w przycisk u dołu ekranu, aplikacja powraca co menu głównego.



Rys. 4.2: Rozpoczęcie testu pamięci krótkotrwałej.

Użytkownik ma dwie możliwości wprowadzania zapamiętanego ciągu. Albo poprzez ręczne wpisywanie z ukazanej klawiatury (Rys. 4.3), albo poprzez wprowadzenie głosowe. Po kliknięciu w przycisk z wizerunkiem mikrofonu, graczowi ukazuje się okno



widoczne na rysunku 4.8. Po takim komunikacie należy wypowiedzieć zapamiętywaną sekwencję i odczekać chwilę na zadziałanie algorytmu. Aplikacja jest czuła na wykrywanie błędu złego wprowadzenia, np. wypowiedzenie samych liter. Przy wystąpieniu takiego wyjątku, program ukáže odpowiedni komunikat (Rys. 4.9). Użytkownik nie musi się martwić o nieznaczne szумы w tle czy wypowiedzenie dodatkowych słów podczas wprowadzania głosowego – Aplikacja jest na to czuła i wychwyci tylko same cyfry.



Rys. 4.3: Ekran wpisywania zapamiętanego ciągu.

Proszę wpisać zapamiętany ciąg cyfr.

8884133

7	8	9	
4	5	6	
1	2	3	
0		DALEJ	

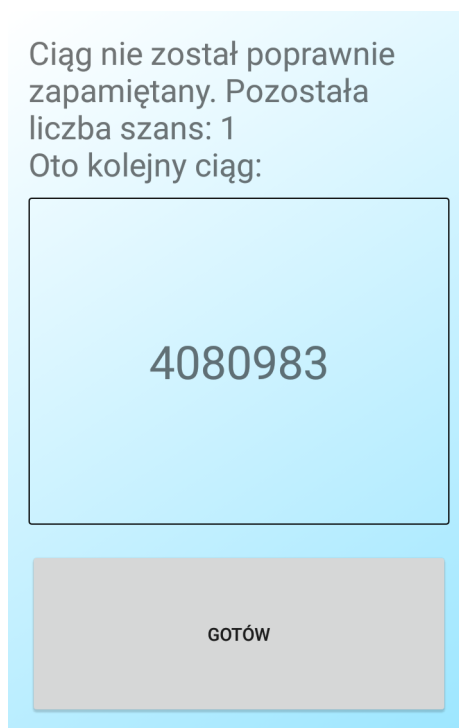
Rys. 4.4: Wpisany zapamiętany ciąg.

Ciąg został poprawnie zapamiętany, oto kolejny:

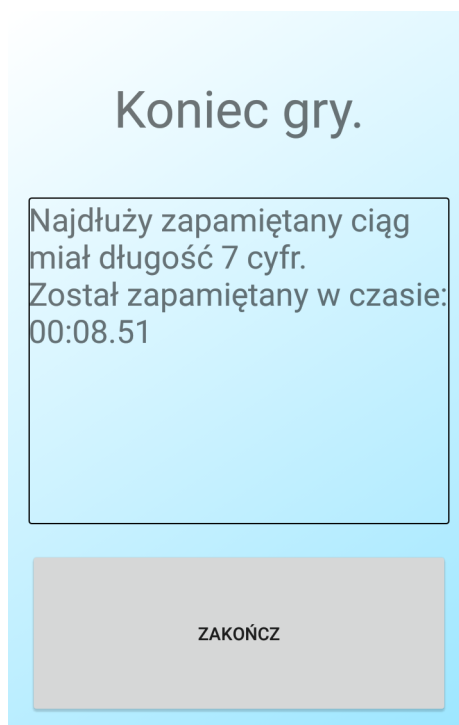
84677881

GOTÓW

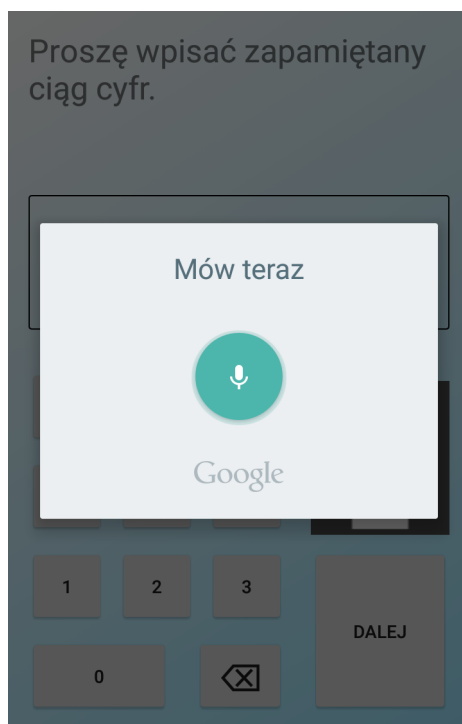
Rys. 4.5: Komunikat o dobrze zapamiętanim ciągu oraz rozpoczęcie kolejnego etapu testu.



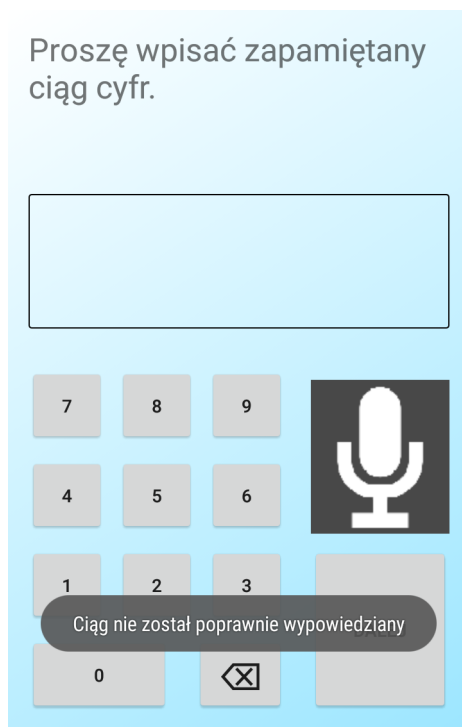
Rys. 4.6: Komunikat o źle zapamiętanym ciągu oraz rozpoczęcie kolejnego etapu.



Rys. 4.7: Zakończenie testu i wyświetlenie statystyk.



Rys. 4.8: Uruchomienie wprowadzania głosowego.



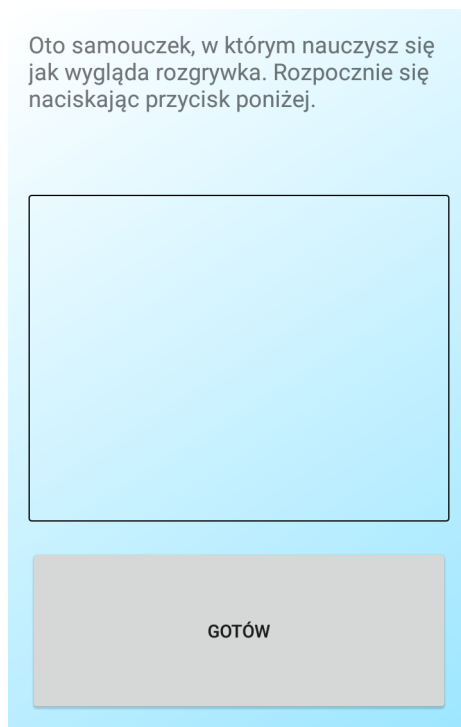
Rys. 4.9: Wychwycenie błędu złego wprowadzania głosowego.

4.3 Samouczek

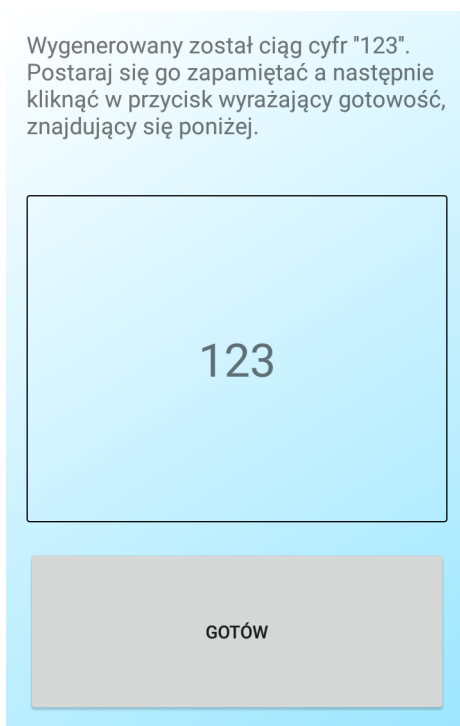
Samouczek wprowadza użytkownika w zasady panujące podczas rozgrywki. Po uruchomieniu go, ukazuje się komunikat o rozpoczęciu samouczka (Rys. 4.10). Klikając w odpowiedni przycisk przechodzi się do następnych etapów. Na początku, użytkownik zostaje zaznajomiony z losowym generowaniem ciągu (Rys. 4.11). Na potrzeby samouczka, ciąg ten jest zawsze taki sam. Użytkownik powinien go zapamiętać i wpisać w następnym etapie (Rys. 4.12). Jeżeli jednak nie uda mu się zrobić tego poprawnie, aplikacja przypomni zapamiętywaną liczbę i da kolejną szansę (Rys. 4.13).

Kolejnym etapem jest zaznajomienie użytkownika z wprowadzaniem głosowym (Rys. 4.14). Przy okazji gracz informowany jest o skutkach poprawnego lub złego otworzenia ciągu we właściwym teście pamięci. Wprowadzanie głosowe odbywa się poprzez naciśnięcie na przycisk z wizerunkiem mikrofonu. Wygląda i działa dokładnie tak samo jak we właściwej rozgrywce (Rys. 4.8). Jeżeli jednak z jakichś powodów użytkownik nie jest w stanie wprowadzić sekwencji głosowo, może uczynić to w sposób podany w poprzedniej części samouczka.

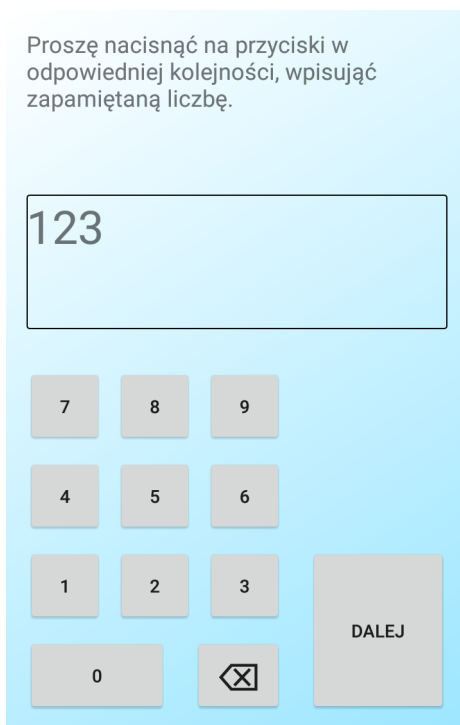
Po prawidłowym przejściu przez poprzedni etap, samouczek kończy się i użytkownikowi wyświetla się ekran końcowy, widoczny na rysunku 4.15). Po kliknięciu w przycisk u dołu, aplikacja przechodzi do właściwego testu pamięci.



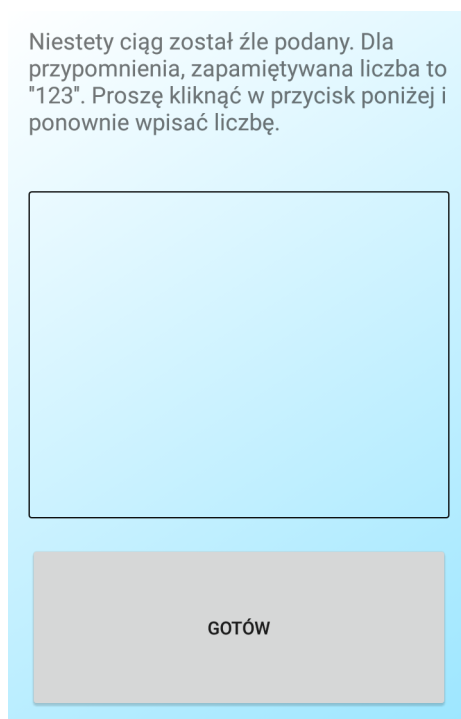
Rys. 4.10: Rozpoczęcie samouczka.



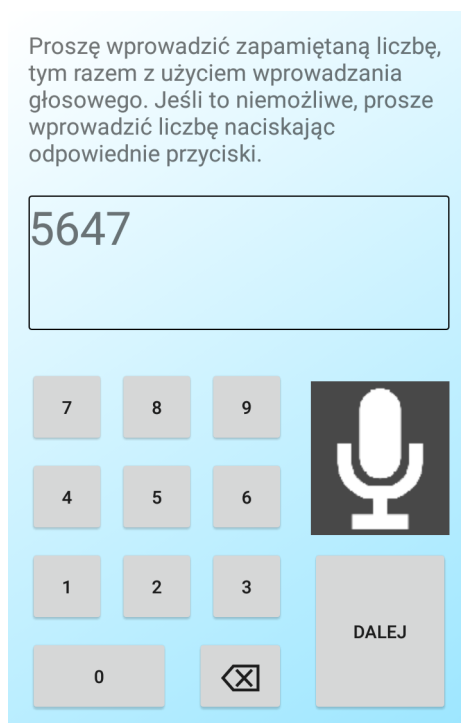
Rys. 4.11: Instrukcja zapamiętania ciągu.



Rys. 4.12: Instrukcja odtworzenia ciągu.

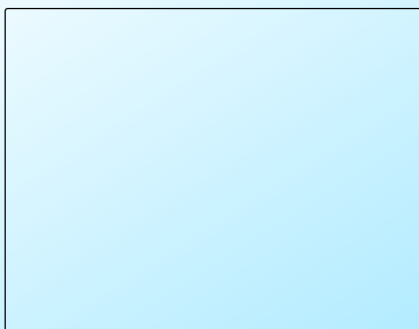


Rys. 4.13: Wychwycenie błędu przy złym odtworzeniu ciągu.



Rys. 4.14: Instrukcja wpisywania głosowego.

Liczba została poprawnie zapamiętana.
Gratulacje, udało się pomyślnie
zakończyć samouczek. Kliknij w
przycisk poniżej aby przejść do
właściwej gry.



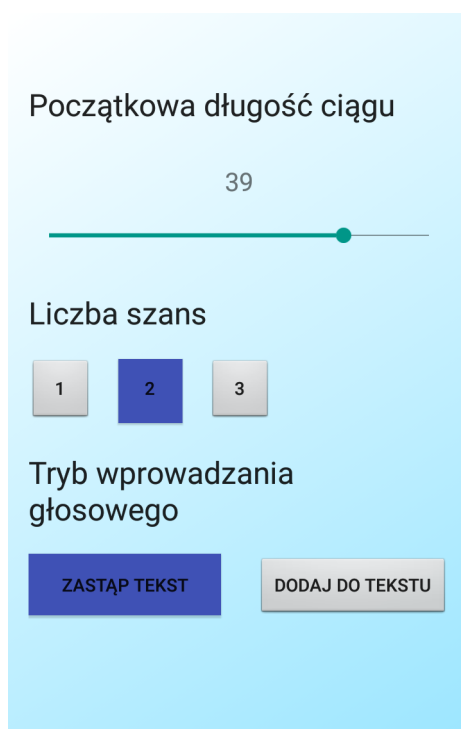
PRZEJDŹ DO GRY

Rys. 4.15: Zakończenie samouczka.

4.4 Opcje

Ekran opcji jest czytelny dla użytkownika (Rys. 4.16). Teksty nad kontrolkami informują go, jakie ustawienie zmieni klikając w elementy. Przesuwając suwak, zmienia się początkowa długość ciągu. Dla zwiększenia czytelności, długość ta jest wypisana nad kontrolką. W celu zmiany liczby szans, wystarczy nacisnąć przycisk z żadaną liczbą szans. Przy trzech szansach użytkownik może pomylić się dwa razy, a trzecia pomyłka spowoduje zakończenie rozgrywki. Analogicznie, jedna szansa oznacza spowodowanie końca testu pamięci po pierwszej pomyłce. Tryb wprowadzania głosowego to ostatnia, możliwa do zmiany, opcja. Możliwe są dwa tryby wprowadzania głosowego - zastąpienie tekstu po wypowiedzeniu nowego lub dodanie kolejnej części do już wprowadzonego.

Zmiany opcji są od razu zapamiętywane i wprowadzone do rozgrywki. Użytkownik informowany jest o zmianie ustawień poprzez podświetlenie klikniętego przycisku.



Początkowa długość ciągu

39

Liczba szans

1 2 3

Tryb wprowadzania głosowego

ZASTĄP TEKST DODAJ DO TEKSTU

Rys. 4.16: Ekran opcji aplikacji.

4.5 Wyniki

Tabela wyników jest widoczna po wybraniu odpowiedniej opcji z menu głównego. Uzupełniana jest po każdorazowym zakończeniu testu pamięci. Pola zawarte w tabeli wyników to:

- data zakończenia testu,
- największa długość zapamiętanego ciągu,
- czas w którym został zapamiętany najdłuższy ciąg.

Wyniki sortowane są po dacie zakończenia tekstu. Tabela jest przewijalna, co umożliwia odczytanie wielu wyników jednocześnie. Niemożliwe jest usunięcie pojedynczego wierszu. Wyniki mogą być odczytywane w celach diagnostycznych, a możliwość usunięcia rekordów może spowodować zakłamanie diagnozy.

Tabela wyników:		
Data:	Długość ciągu:	Czas:
21.12.2017, 21:02	8	00:03.02
21.12.2017, 21:03	8	00:05.84
21.12.2017, 21:04	9	00:06.63
21.12.2017, 21:04	8	00:02.81
21.12.2017, 21:05	9	00:03.37
21.12.2017, 21:06	7	00:02.61
23.12.2017, 00:07	7	00:02.67

Rys. 4.17: Tabela wyników.

5. Testy

Aplikacja została przetestowana na różnych grupach odbiorców, a mianowicie:

- osoby zdrowe, nie trenujące pamięci,
- osoby zdrowe, regularnie trenujące pamięć,
- osoby w różnych kategoriach wiekowych.

Każda grupa osób bezproblemowo poradziła sobie z korzystaniem z aplikacji. Nie wszyscy jednak postanowili skorzystać z samouczka i dopiero po pierwszej rozgrywce zaznajomili się z zasadami. Interfejs użytkownika został zaprojektowany w czytelny sposób, zatem nie utrudniał on użytkownika. Również samo tło nie odwracało uwagi i nie przerywało skupienia przy wykonywaniu testu.

Dla osób trenujących pamięć, aplikacja pozwalała osiągnąć wyniki zgodne z zadeklarowanymi. Oznacza to, że korzystanie z aplikacji nie wpływa negatywnie na zdolności pamięciowe. Każda z osób trenujących osiągała inne wyniki, gdyż ich zdolności zapamiętywania są różnie rozwinięte. Inny przypadek występował wśród ludzi nie trenujących pamięci. Ci potrafili zapamiętać ciąg o długości średnio 8-9 cyfr. Grupa badanych była zbyt mało liczna by można było stwierdzić różnice w zdolnościach pamięci między ludźmi o różnej płci czy różnym wieku.

Jedyną istotną uwagą do aplikacji była uwaga o możliwości rozegrania tylko jednego rodzaju testu. Jednakże, jako że kod aplikacji został napisany w sposób przystępny dla nawet początkującego programisty, istnieje łatwa możliwość rozszerzenia programu o kolejne testy. Kolejnym elementem rozwoju aplikacji jest zapisywanie wyników do zewnętrznej bazy danych, a nie do pliku na urządzeniu. Da to możliwość diagnozowania zdolności pamięciowych oraz dokonania ewentualnych badań naukowych.

....Zastanowić się co dalej, narazie jest blokada mózgu + ewentualne uwagi: co warto rozwijać itp.

6. Podsumowanie i wnioski

Celem pracy było Zrobiono to, tamto i siamto. Cel pracy został osiągnięty (lub nie)

Bibliografia

- [1] ANDERSON, J. R. *Uczenie się i pamięć – integracja zagadnień*. Wydawnictwa Szkolne i Pedagogiczne, Warszawa, 1998.
- [2] BUDOHOSKA, W., W. Z. *Psychologia uczenia się*. Przegląd badań eksperymentalnych i teorii. Państwowe Wydawnictwo Naukowe, Warszawa, 1977.
- [3] GAZZANIGA, M. S. *The Cognitive Neurosciences*, 4 ed. The MIT Press, Londyn, 2009.
- [4] JAGODZIŃSKA, M. *Psychologia pamięci. Badania, teorie, zastosowania*. Sensus, 2008.
- [5] JODZIO, K. *Neuronalny Świat umysłu*. Oficyna wydawnicza Impuls, Kraków, 2005.
- [6] NIEDŹWIEŃSKA, A. Rodzaje testów do badania pamięci.
- [7] VETULANI, J. Pamięć: podstawy neurobiologiczne i możliwości wspomagania.