Michał Kopczyński, 184817
Maksymilian Terebus, 181595
KSD ACiR 1

Sprawozdanie – Zadanie 3
Architektura Systemów Komputerowych
Opis zadania oraz założenia szczegółowe

Wstęp

Głównym celem zadania było stworzenie aplikacji uzależnionej od czasu. Aplikacja została napisana w języku Python, używając przy tym framework'a PyQt5.
Napisaliśmy aplikację, która pozwala na wykonanie 4 testów (gier) pozwalających na precyzjne odmierzanie czasu.
Są to:

Główne menu

Każda gra wymaga kliknięcia przycisku w odpowiednim czasie, wtedy jest mierzony czas reakcji.
W przypadku reakcji audio, użytkownik ma obowiązek kliknąć przycisk po usłyszeniu sygnału dźwiękowego.
W przypadku reakcji video, użytkownik ma obowiązek kliknąć przycisk po zmianie koloru przycisku.

W przypadku dopasowania kolor-tekst, użytkownik ma obowiązek kliknąć przycisk po odpowiednim wyświetleniu napisu koloru w odpowiadającym mu w rzeczywistości kolorze czcionki.
W przypadku ciągu rytmicznego, użytkownik ma obowiązek wpasować się w rytm migającego na czerwono przycisku (interwał 1s), aby samemu wcisnąć przycisk na 4 takt (zgodnie z rytmem wyznaczonym przez poprzednie 3 mignięcia przycisku/takty).
Nie będziemy tutaj wklejać poszczególnych interfejsów, ponieważ zajęłoby to ¾ sprawozdania by wszystko pokazać, wszystko zostanie pokazane na odbiorze laboratorium.
Końcowy interfejs po wykonaniu badań (pomierzone czasy):

Główne menu po wykonaniu badania

Jak widać pomierzone czasy zostały zapisane po wykonaniu zadania.

Opis przyjętych założeń programowych

Każdy test, jest widgetem z biblioteki PyQt5. Po kliknięciu odpowiedniego przycisku wyświetla się odpowiedni widget.
Przykład tworzenia widgetu na podstawie reakcji audio:

```
class Test2(QWidget):
    Maksymilian Terebus
    def __init__(self, parent=None):
        super().__init__(parent)
        self.main_window = parent
        self.timer = None
        self.start_time = None
        self.big_button_visible = False
        self.background_color = 'white'
        self.color_changed = False
```

Mierzenie czasu reakcji:

```
    1 usage   Maksymilian Terebus
    def play_audio_and_start_timer(self):
        # Create a QMediaPlayer instance
        self.player = QMediaPlayer()

        # Set the media source (audio file) and play it
        self.player.setMedia(QMediaContent(QUrl.fromLocalFile("beep.wav")))
        self.player.play()

        self.audio_played = True
        self.start_time = time.time()
        self.timer.start()

    1 usage   Maksymilian Terebus
    def stop_timer(self):
        if self.audio_played:
            elapsed_time = self.timer.elapsed()
            self.player.stop()
            if self.test_approach:
                self.main_window.showTestResult(elapsed_time, 1)
            else:
                self.main_window.showTestResult(elapsed_time)
```

Tutaj przykład jak działa timer w teście nr 1. Jeżeli pojawił się sygnał dźwiękowy, timer rozpoczyna liczenie czasu. Po kliknięciu na odpowiedni guzik, timer się zatrzymuje i oblicza czas reakcji użytkownika. Działa to analogicznie we wszystkich testach, bazując na odpowiednim warunku zatrzymania timera oraz jego startu.

Po wykonaniu badania, rezultaty wyświetlają się za pomocą funkcji showTestResult w oknie głównym:

```
def showTestResult(self, reaction_time, retry_callback=None):
    self.clearLayout()
    self.test_approach = False



    central_widget = QWidget()
    self.setCentralWidget(central_widget)

    layout = QVBoxLayout()
    central_widget.setLayout(layout)

    # Test name
    test_label = QLabel(self.test_names[self.test_id])
    test_label.setFont(QFont('Arial', 30))
    test_label.setAlignment(Qt.AlignCenter)
    layout.addWidget(test_label)

    # Test result
    result_label = QLabel(f"Reaction time: {reaction_time} ms")
    result_label.setFont(QFont('Arial', 30))
    result_label.setAlignment(Qt.AlignCenter)
    layout.addWidget(result_label)
```

Wady programu oraz zalety

Dużą zaletą progamu jest na pewno przyjazny i łatwy w obsłudze interfejs użytkownika oraz miarodajność testów reakcji.
Oprócz tego, zaletą jest również możliwość podejść próbnych do testu, zanim rozpocznie się finalne badanie.
Warto również wspomnieć o tym, że aplikacja jest łatwo skalowalna, to znaczy możliwość dodania nowej gry nie stanowi większego wyzwania, ponieważ możemy skorzystać z szablonu, który został przez nas napisany.
Natomiast wadami programu są:

- Lekkie opóźnienie uruchomienia pliku audio
- Redundancja w kodzie, kod jest do zoptymalizowania