Michał Kopczyński, 184817
Maksymilian Terebus, 181595
KSD ACiR 1

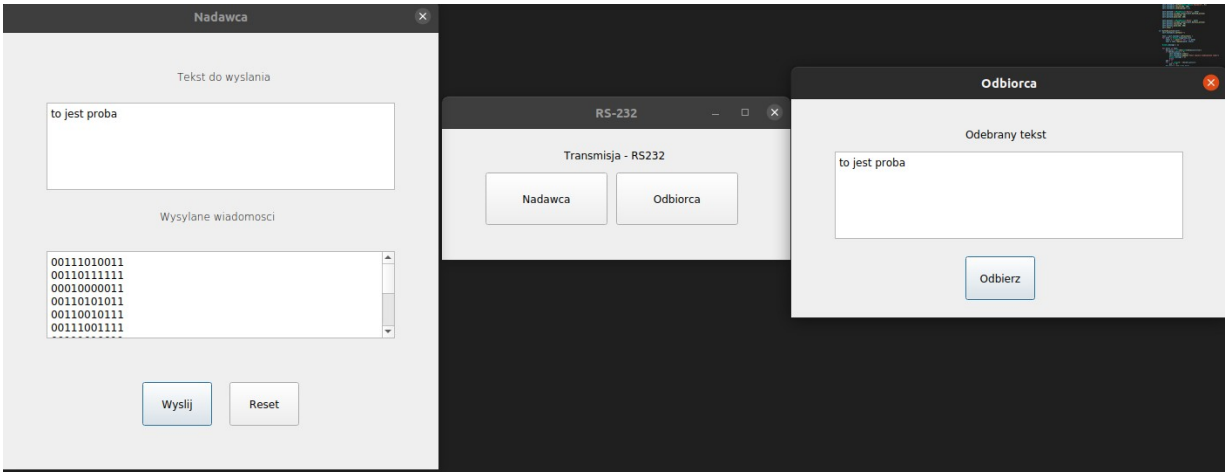
Sprawozdanie – Zadanie 2
Architektura Systemów Komputerowych

Opis zadania oraz założenia szczegółowe

Głównym celem zadania było stworzenie aplikacji symulującej transmisję szeregową zgodną ze standardem RS232.

Aplikacja jest napisana w języku Python, z wykorzystaniem biblioteki PyQt5. Program składa się z trzech okien – okna menu, okno nadawcy i okno odbiorcy. Użytkownik może wpisać swój tekst w edytowalne pole tekstowe w oknie nadawcy. Gdy tekst jest gotowy można go wysłać klikając w przycisk wyślij. Po wciśnięciu będzie dostępny podgląd łańcuchów (bit startu, bity znaku oraz bity stopy) wysyłanych do odbiorcy. Łańcuchy te tworzone są w następujący sposób:

Tekst jest pobierany z edytowalnego pola tekstowego i szukane są w nim przekleństwa, które wcześniej były pobrane z pliku „grubiaństwa.txt” i zapisane do listy. Przekleństwa są zamieniane na ciąg znaków w postaci „** […]”. Następnie pobierane są wszystkie znaki zawarte we wpisanym, wyczyszczonym już tekście. Konwertowane są na łańcuch zawierający liczbę w kodzie binarnym reprezentującą kod ASCII. Do tak stworzonego łańcucha dodawany jest na początek bit startu oraz na końcu bity stopu. Na końcu łańcuchy te są zapisywane w liście, która z kolei jest przechowywana w głównej klasie programu. Odbiornik jest w stanie tą listę przeczytać, gdyż wspomniana lista jest statycznym elementem klasy Window. Następnie każdą wiadomość jest poprawnie rozszyfrowywana, co będzie omówione w następnym rozdziale.



Rys. Podgląd programu

Opis przyjętych założeń programowych

Klasy programu

Każde okno dostępne w programie jest instancją klasy. Główną klasą jest klasa „Window”, która zawiera menu do wyboru strony komunikacji. W tej klasie zawarta jest również statyczna lista, na której oparta jest cała komunikacja. Do listy można mieć dostęp z całego programu, w szczególności z klasy nadajnika oraz odbiornika.

Filtrowanie przekleństw

Słowa z pliku są pobierane do listy:

```
vulgarism_list = [line.rstrip(_) for line in open('grubianstwa.txt')]
```

Tekst z pola edytowalnego jest konwertowany do prostego tekstu w postaci łańcucha znaków i zastępowane są w nim wulgaryzmy łańcuchem znaków w postaci „** [..]”

```
text = self.textedit.toPlainText()
for word in Window.vulgarism_list:
    stars = "".join("*" for _ in word)
    text = text.replace(word, stars)
```

Szyfrowanie tekstu

Każda litera tekstu jest konwertowana tak jak wspomniano w pierwszym rozdziale. Gdy zwiera on niedozwolone znaki, nie można takiego tekstu wysłać. Gdy uzyskany binarny kod ascii znaku jest zbyt krótki dodawane są do niego zera. Następnie łańcuch otaczany jest ramką i zapisywany w liście Window.message.

```
Window.message = []

for ascii in text:
    bin_ascii = str(bin(int(ord(ascii))))[2:]
    print(bin_ascii)
    if len(bin_ascii) > 7:
        self.textedit2.clear()
        self.textedit2.append("Tekst zawiera niedozwolone znaki")
        Window.message = []
        break
    pad = ""
    for _ in range(8 - len(bin_ascii)):
        pad += "0"
    bin_ascii = pad + bin_ascii
    bin_frame = "0" + bin_ascii + "11"
    Window.message.append(bin_frame)
    self.textedit2.append(bin_frame)

self.textedit2.setEnabled(True)
```

Odszyfrowywanie tekstu

Ramka (bit startu i bity stopu) są usuwane, następnie łańcuch reprezentujący liczbę binarną jest konwertowany na liczbę w kodzie dziesiętnym. Ten kod odpowiada pewnemu znakowi w

tablicy ASCII i znak ten jest dodawany do łańcucha wyświetlanego w osobnym oknie odbiorcy

```
def button0_action(self):
    self.textedit.setDisabled(False)
    message = ""
    if Window.message is not []:
        for frame in Window.message:
            ascii_bin = frame[1:-2]
            ascii_dec = 0
            power = 7
            for bit in ascii_bin:
                ascii_dec += int(bit) * (2 ** power)
                power -= 1
            message += chr(ascii_dec)
        self.textedit.setText(message)
    else:
        self.textedit.setText("Blad komunikacji lub nic nie wyslano")
```

Uzyskane wyniki oraz wady i zalety programy

W naszej opinii program spełnia postawione zadanie symulacji transmisji szeregowej w standardzie RS232.

Ważną zaletą programu jest efektywne filtrowanie przekleństw. W całym tekście są znajdywane charakterystyczne dla przekleństw człony wyrazów. W ten sposób możliwe jest filtrowanie wielu kombinacji tego samego przekleństwa z użyciem niedużej bazy danych nt. grubiaństw. Program jest zwięzły i czytelny a GUI jest intuicyjne i uporządkowane.

Wadą programu może być fakt nieprzesyłania polskich znaków.