Projekt z programowania - Aplikacja do nauki języka angielskiego "'PolishYourEnglish"'

Kamil Chmielak, Bartłomiej Fijołek

31.01.2024

Spis treści

1. Temat projektu

Tematem projektu było stworzenie w c++ aplikacji „PolishYourEnglish", której celem jest pomoc w nauce słownicta języka angielskiego. Aplikacja działa na zasadzie popularnej metody nauki - „fiszek".

2. Wykorzystane narzędzia do stworzenia aplikacji PolishYourEnglish

2.1. Język programowania

Projekt napisany został w języku C++.

2.2. Interfejs graficzny

Interfejs graficzny stworzony został za pomocą wieloplatformowej biblioteki wxWidgets w wersji 3.0.2 służącej do tworzenia aplikacji okienkowych.

2.3. IDE

Jako środowiska programistycznego użyliśmy Code::Blocks – wieloplatformowego, zintegrowanego środowiska programistycznego na licencji GNU w wersji 20.03.

3. Instrukcja użytkowania

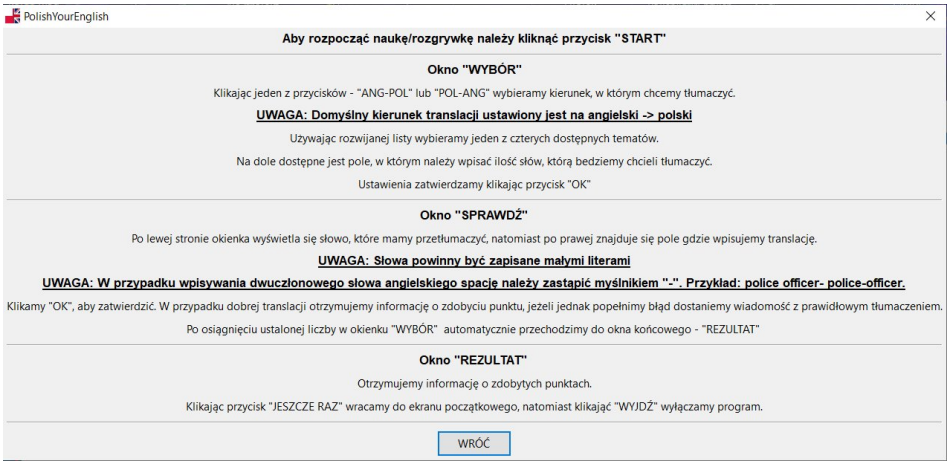
3.1. Okno Główne

Po uruchomieniu programu wyskakuje okno główne aplikacji (rysunek 1) „PolishYourEnglish". Aby rozpocząć wciskamy przycisk „start", możemy również wcisnąć przycisk „instrukcja", gdzie użytkownik również otrzyma instrukcję obsługi oraz zasady działania aplikacji.

Rysunek 1: Okno programu PolishYourEnglish

3.2. Okno Instrukcja

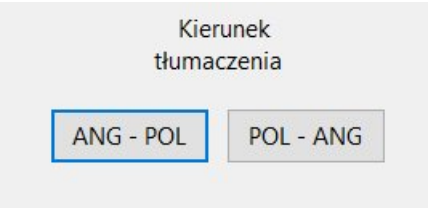
Użytkownikowi wyświetla sie krótka instrukcja obsługi (rysunek 2). Aby wrócić do menu głównego aplikacji należy kliknąć przycisk „wróć".



Rysunek 2: Okno programu PolishYourEnglish - instrukcja

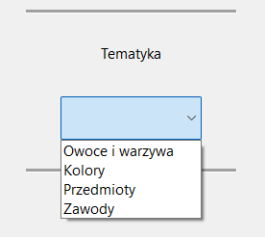
3.3. Okno Wybor

Korzystając z jednego z dwóch dostępnych przycisków (rysunek 3) pod tekstem „Kierunek tłumaczenia" wybieramy w jaki sposób będziemy chcieli uczyć się słówek. Klikając przycisk „ANG - POL" wyświetlać się będą słowa w języku angielskim, natomiast użytkownik będzie zobligowany do wpisania słowa w języku polskim. Analogicznie działa przycisk „POL - ANG" wyświetlać będą się słowa polskie, natomiast użytkownik wpisywać będzie tłumaczenie tego słowa w języku angielskim. Domyślnie kierunek ustawiony jest na „angielski - polski".



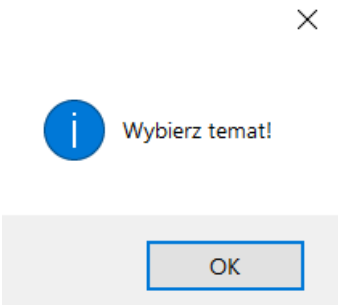
Rysunek 3: Okno programu PolishYourEnglish - przyciski do wyboru kierunku

Rozwijana lista (rysunek 4) pozwala na wybór jednego z czterech tematów słówek, których użytkownik będzie chciał się nauczyć. Dostępne tematy to: „Owoce i warzywa", „Kolory", „Przedmioty", „Zawody".



Rysunek 4: Okno programu PolishYourEnglish - rozwijana lista do wyboru tematu

W przypadku nie wybrania żadnego, wyskoczy komunikat (rysunek 5) o treści „Wybierz temat!”.



Rysunek 5: Okno programu PolishYourEnglish - rozwijana lista brak wyboru

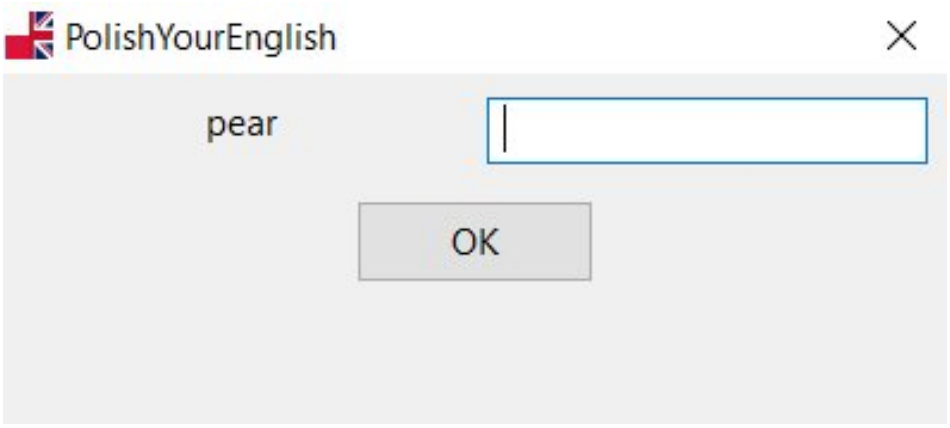
Ostatnim elemenetem jest wybór liczby słów (rysunek 6), którą użytkownik będzie tłumaczył. Należy wybrać strzałkami obok lub wpisać ręcznie zakres od 1 do 50, w przypadku liczby większej niż 50 program automatycznie ustawi liczbę 50, analogicznie z liczbą mniejszą niż 1 - aplikacja ustawi liczbę 1.

Wszystkie wybory zatwierdzamy klikając przycisk „OK”.

Rysunek 6: Okno programu PolishYourEnglish - wybór liczby fiszek

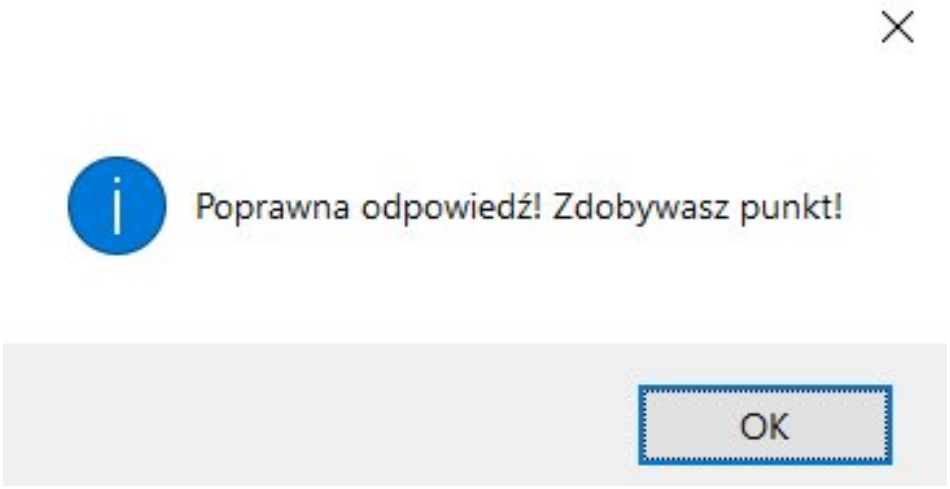
3.4. Okno Sprawdź

W tej części wyświetlają się „fiszki” ze słowami w zależności od wybranej tematyki oraz kierunku tłumaczenia (rysunek 7). Z lewej stronie okienka wyświetla się słowo, które należy przetłumaczyć, natomiast po prawej stronie znajduje się miejsce do wpisania translacji przez użytkownika. Tłumaczenie słowa należy wpisać małymi literami, natomiast w przypadku wpisywania dwuczłonowego słowa angielskiego znak spacji należy zastąpić myślnikiem „-”.

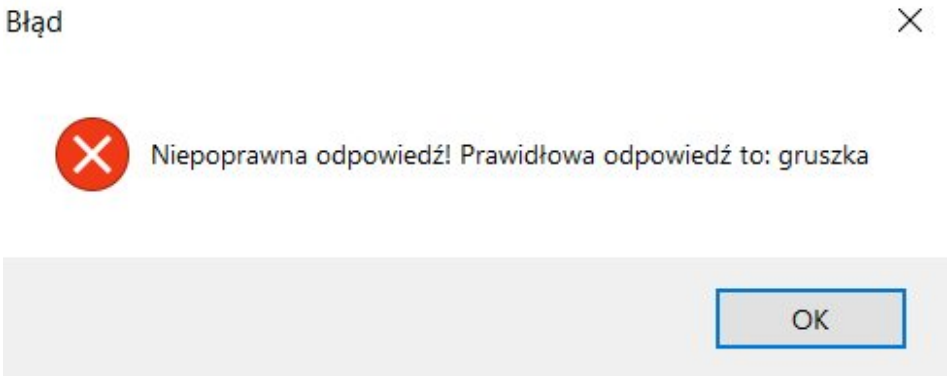


Rysunek 7: Okno programu PolishYourEnglish - okno sprawdź - fiszka

Translację zatwierdzamy klikając przycisk „OK”. W wyskakującym okienku otrzymujemy odpowiedź, w przypadku poprawnej odpowiedzi (rysunek 8) komunikat jest następujący - „Poprawna odpowiedź! Zdobywasz punkt!”, natomiast w przypadku odpowiedzi błędnej (rysunek 9) użytkownik otrzymuje wiadomość z poprawnym tłumaczeniem o treści - „Niepoprawna odpowiedź! Prawdiłowa odpowiedź to:...”



Rysunek 8: Okno programu PolishYourEnglish - okno sprawdź - prawidłowa odpowiedź

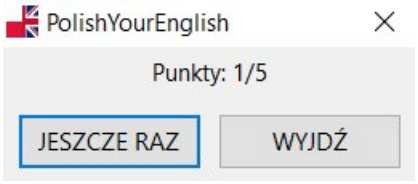


Rysunek 9: Okno programu PolishYourEnglish - okno sprawdź - zła odpowiedź

Po osiągnięciu ustalonej wcześniej liczby słówek, aplikacja przechodzi automatycznie do okienka z rezultatem naszej nauki/rozgrywki.

3.5. Okno Rezultat

Otrzymujemy informację o ilości poprawnych translacji (jedna translacja - jeden punkt) oraz ilości maksymalnej ilość punktów (rysunek 10). Użytkownik może rozpocząć naukę klikając przycisk „JESZCZE RAZ", który przekierowuje nas do okna głównego lub zakończyć działanie aplikacji klikając przycisk „WYJDŹ".



Rysunek 10: Okno programu PolishYourEnglish - okno z rezultatem końcowym

4. Działanie Aplikacji

4.1. klasa Translate

Konstruktor klasy Translate

Konstruktor tworzy obiekt przchowujący dwie tablice słów - pierwszą, w której znajdują się słowa w języku polskim oraz drugą, gdzie w odpiwadających indeksach znajduje się tłumaczenie słowa na język angielski.

```
// Konstruktor dwuargumentowy
 Translator(std::string* polish, std::string* english, int arraySize) {
     size = arraySize;
     polishWords = new std::string[size];
     englishWords = new std::string[size];

     // Inicjalizacja tablic za pomoca dostarczonych danych
     for (int i = 0; i < size; ++i) {
         polishWords[i] = polish[i];
         englishWords[i] = english[i];
     }
 }
```

Destruktor klasy Translate

```
// Destruktor do zwalniania zaalokowanej pamieci
  ~Translator() {
      delete[] polishWords;
      delete[] englishWords;
  }
```

4.2. Plik wybor.cpp

Przycisk ANG-POL oraz POL-ANG

Klikając na przycisk ustawiamy zmiennej typu int wybranyKierunek wartość 1 lub 2 w zależności co kliknął użytkownik, zmienna ta jest odpowiedzialna za kierunek translacji, dzięki niej odpala się odpowiednia instrukcja w późniejszym etapie.

```
void Wybor::OnButton1Click(wxCommandEvent& event)
{
    // ustawia kierunek tlumaczenia Ang-Pol
    wybranyKierunek = 1;
}

void Wybor::OnButton2Click(wxCommandEvent& event)
{
    // ustawia kierunek tlumaczenia Pol-Ang
    wybranyKierunek = 2;
}
```

Przycisk OK

Klikając przycisk „OK” z okna wyboru zostaje pobrany temat słówek i zapisany w zmiennej wxString wybranyTemat. Następnie pobierana jest liczba słówek podana przez użytkownika w „SpinCtrl1” oraz zapisywana w zmiennej int value, gdzie kolejno przypisujemy ją do zmiennej Liczbaslowek.

Instrukcją warunkową sprawdzamy, który z tematów został wybrany i w zależności od tego ustawiamy zmienną typu int Tematslowek. Ostatecznie przechodzimy do następnego okna.

```
void Wybor::OnButton3Click(wxCommandEvent& event)
{

    //Pobranie tematu slowek
    wxString wybranyTemat = Choice1->GetStringSelection();

   //Pobranie ilosci slowek do translacji
      int value = SpinCtrl1->GetValue();
      Liczbaslowek = value;


            if (wybranyTemat == "Owoce i warzywa") {
                Tematslowek = 1;

                    // Zamknij bieżące okno dialogowe
                    this->EndModal(wxID_CANCEL);
                    // Otwórz nowe okno
                    Sprawdz dlg(0);
                    dlg.ShowModal();

            } else if (wybranyTemat == "Kolory") {
                Tematslowek = 2;

                // Zamknij bieżące okno dialogowe
                    this->EndModal(wxID_CANCEL);
                    // Otwórz nowe okno
                    Sprawdz dlg(0);
                    dlg.ShowModal();
```

```
        } else if (wybranyTemat == "Przedmioty") {
            Tematslowek = 3;

            // Zamknij bieżące okno dialogowe
                this->EndModal(wxID_CANCEL);
                // Otwórz nowe okno
                Sprawdz dlg(0);
                dlg.ShowModal();
        }else if (wybranyTemat == "Zawody") {
            Tematslowek = 4;

            // Zamknij bieżące okno dialogowe
                this->EndModal(wxID_CANCEL);
                // Otwórz nowe okno
                Sprawdz dlg(0);
                dlg.ShowModal();
        }else{
            wxMessageBox("Wybierz␣temat!", "");
    }


}
```

4.3. Plik sprawdz.cpp

Tworzenie konstruktora

W zależności jaki wcześniej wybrany był temat słówek używając konstrukcji warunkowej tworzymy konstruktor z odpowiednią tablicą słówek.

```
//Stworznie konstruktora
        if(Tematslowek==1){
            translator = new Translator(polish1, english1, 50);
        } else if(Tematslowek==2){
            translator = new Translator(polish2, english2, 50);
        } else if(Tematslowek==3){
            translator = new Translator(polish3, english3, 50);
        } else if(Tematslowek==4){
            translator = new Translator(polish4, english4, 50);
        }
```

Tablice słówek

Tablice znajdują się w konstruktorze okienka „sprawdź”.

```
    std::string polish1[] = {
    "marchewka", "ziemniak", "cebula", "pomidor", "kapusta",
    "brokuł", "fasola", "pietruszka", "chrzan", "seler",
    "por", "kukurydza", "kalafior", "bakłażan", "ogórek",
    "szparagi", "dynia", "kabaczek", "czosnek", "cukinia",
    "szpinak", "papryka", "kalarepa", "burak", "rzodkiewka",
    "słonecznik", "róża", "banan", "jagoda", "truskawka",
    "malina", "aronia", "śliwka", "morela", "gruszka",
    "jabłko", "wiśnia", "ananas", "awokado", "sałata",
    "koperek", "melon", "bób", "szczypiorek","czereśnia",
```

```
    "karczoch", "jarmuż","arbuz","granat","grejpfrut"};

    std::string english1[] = {
    "carrot", "potato", "onion", "tomato", "cabbage",
    "broccoli", "bean", "parsnip", "horseradish", "celery",
    "leek", "corn", "cauliflower", "eggplant", "cucumber",
    "asparagus", "pumpkin", "zucchini", "garlic", "courgette",
    "spinach", "pepper", "kohlrabi", "beetroot", "radish",
    "sunflower", "rose", "banana", "blueberry", "strawberry",
    "raspberry", "chokeberry", "plum", "apricot", "pear",
    "apple", "cherry", "pineapple", "avocado", "lettuce",
    "dill", "melon", "broad␣bean", "chive","cherry",
    "artichoke", "kale","watermelon","pomegranate","grapefruit"};

    std::string polish2[] = {
    "czerwony", "zielony", "niebieski", "żółty", "fioletowy",
    "różowy", "pomarańczowy", "brązowy", "szary", "biały",
    "czarny", "srebrny", "złoty", "fioletowy", "indygo",
    "turkusowy", "magenta", "turkusowy", "oliwkowy", "kasztanowy",
    "malinowy", "bursztynowy", "lawendowy", "antracytowy", "karmazynowy",
    "jagodowy", "miętowy", "musztardowy", "lazurowy", "cytrynowy",
    "granatowy", "fuksjowy", "orzechowy", "szmaragdowy", "chabrowy",
    "błękitny", "truskawkowy", "perłowy", "morski", "ciemnoniebieski",
    "fuksjowy", "wiśniowy", "kobaltowy", "amarantowy", "tytanowy",
    "hebanowy", "szafirowy", "kasztanowy", "siwobłękitny", "ochrowy"
    };

    std::string english2[] = {
    "red", "green", "blue", "yellow", "purple",
    "pink", "orange", "brown", "grey", "white",
    "black", "silver", "gold", "violet", "indigo",
    "turquoise", "magenta", "turquoise", "olive", "chestnut",
    "raspberry", "amber", "lavender", "anthracite", "crimson",
    "berry", "mint", "mustard", "azure", "lemon",
    "navy", "fuchsia", "walnut", "emerald", "cerulean",
    "cyan", "strawberry", "pearl", "marine", "dark-blue",
    "fuchsia", "cherry", "cobalt", "amaranth", "titanium",
    "ebony", "sapphire", "brownish", "grayish-blue", "ochre"};


    std::string polish3[] = {
    "komputer", "telefon", "książka", "zeszyt", "długopis",
    "laptop", "telewizor", "aparat", "klawiatura", "mysz",
    "okulary", "torba", "kalendarz", "biurko", "zeszytówka",
    "flamastry", "obraz", "krzesło", "poduszka", "szafa",
    "światło", "butelka", "okno", "drzwi", "kubek",
    "słuchawki", "portfel", "mapa", "globus", "dywan",
    "lampa", "łóżko", "księga", "lupa", "dźwięk",
    "klucz", "zegar", "telewizor", "aparat␣fotograficzny", "obrazek",
    "tablet", "stolik", "świeczka", "ręcznik", "łazienka",
    "sztućce", "widelec", "łyżka", "nożyczki", "kuchenka"
};

    std::string english3[] = {
    "computer", "phone", "book", "notebook", "pen",
    "laptop", "television", "camera", "keyboard", "mouse",
```

```
    "glasses", "bag", "calendar", "desk", "notebook",
    "markers", "picture", "chair", "pillow", "wardrobe",
    "light", "bottle", "window", "door", "cup",
    "headphones", "wallet", "map", "globe", "carpet",
    "lamp", "bed", "book", "magnifying glass", "sound",
    "key", "clock", "television", "camera", "picture",
    "tablet", "table", "candle", "towel", "bathroom",
    "cutlery", "fork", "spoon", "scissors", "stove"
};

    std::string polish4[] = {
    "lekarz", "nauczyciel", "inżynier", "kelner", "programista",
    "psycholog", "dziennikarz", "mechanik", "architekt", "artysta",
    "fotograf", "piłkarz", "aktor", "muzyk", "kucharz",
    "prawnik", "strażak", "piekarz", "policjant", "ogrodnik",
    "księgowy", "listonosz", "florysta", "dostawca", "kierowca",
    "elektryk", "dyrektor", "barman", "tłumacz", "konsultant",
    "fryzjer", "sprzątacz", "dyplomata", "geodeta", "rybak",
    "sportowiec", "reżyser", "operator", "ratownik", "pielegniarz",
    "tancerz", "spawacz", "eksplorator", "rolnik", "astronauta",
    "stewardesa", "listonosz", "projektant", "socjolog","chemik"
};


    std::string english4[] = {
    "doctor", "teacher", "engineer", "waiter", "programmer",
    "psychologist", "journalist", "mechanic", "architect", "artist",
    "photographer", "footballer", "actor", "musician", "chef",
    "lawyer", "firefighter", "baker", "police-officer", "gardener",
    "accountant", "postman", "florist", "delivery-person", "driver",
    "electrician", "director", "bartender", "translator", "consultant",
    "hairdresser", "cleaner", "diplomat", "surveyor", "fisherman",
    "athlete", "director", "operator", "lifeguard", "nurse",
    "dancer", "welder", "explorer", "farmer", "astronaut",
    "flight attendant", "postman", "designer", "sociologist","chemist"
    };
```

Pierwsze słówko oraz licznik

W konstruktorze okienka sprawdź.cpp znajduje się również fragment, który losuje pierwsze słowo a następnie w zależności od wybranego wcześniej kierunku (wybranyKierunek) ustawia nazwę przyciskowi „StaticText1”, gdzie wyświetlać się będą słowa do translacji. Ustawia się również zmiennej typu int Licznik wartość 1, jest ona odpowiedzialna za odpowiednią ilość wyświetlanych „fiszek” , dzięki inkrementacji do momentu, gdzie Licznik będzie równy zmiennej Liczbaslowek.

```
        //ustawianie pierwszego słówka do translacji
    randomIndex = rand() % 50;

switch (wybranyKierunek) {
    case 1:
        StaticText1->SetLabel
        (wxString::FromUTF8(translator->GetSlowoAng(randomIndex).c_str()));
        break;
    case 2:
        StaticText1->SetLabel
        (wxString::FromUTF8(translator->GetSlowoPol(randomIndex).c_str()));
```

```
        break;
    }
    Licznik=1;
```

Przycisk OK

Program pobiera wpisane słowo od użytkownika z kontrolki „TexTCtrl1", następnie w zależności od wcześniej wybranego kierunku korzystając z przełącznika egzekwujemy jedną z instrukcji. W poniższym przypadku (case 1 - translacja angielsko-polska) wykonujemy konwersję tekstu między kodowaniem znaków, czego wynikiem jest utworzona zmienna wxString tekstZPolskimiZnakami zawierający tekst podany przez użytkownika w formie szerokiego znaku, który może obsługiwać polskie znaki diakrytyczne. Następnie zamieniamy również słowo do porównania z typu string na typ wxString, w celu możliwości porównania ich w instrukcji warunkowej. Przed porównaniem w instrukcji warunkowej usuwamy znaki spacji oraz ustawiamy małe litery w słowie podanym przez użytkownika (zmienna tekstZPolskimiZnakami). Ostatecznie dostajemy wiadomość z rezultatem oraz inkrementujemy licznik.

```
void Sprawdz::OnButton1Click(wxCommandEvent& event)
{
    wxString tekst = TextCtrl1->GetValue();

switch (wybranyKierunek) {
case 1: {
     // Kod dla case 1

    wxMBConvUTF8 converter;
    wxWCharBuffer wbuf = converter.cMB2WC(tekst.ToUTF8());
    wxString tekstZPolskimiZnakami(wbuf);

    // Dzięki temu możemy porównać polskie znaki
    wxString slowoDoPorownania =
    wxString::FromUTF8(translator->GetSlowoPol(randomIndex).c_str());

    // Usuwanie spacji ze słowa podanego przez użytkownika
    tekstZPolskimiZnakami.Replace("␣", "", true);
    tekstZPolskimiZnakami.MakeLower();

    // Sprawdzenie czy uzytkownik dobrze przetlumaczyl
    if (tekstZPolskimiZnakami == slowoDoPorownania) {
        wxMessageBox
        (wxString::FromUTF8("Poprawna␣odpowiedź!␣Zdobywasz␣punkt!"),
        wxT(""), wxOK | wxICON_INFORMATION, this);
        Punkty++;
        } else {
        wxMessageBox(wxString::Format
        (wxT("Niepoprawna␣odpowiedź!␣Prawidłowa␣odpowiedź␣to:␣%s"),
        wxString::FromUTF8(translator->GetSlowoPol(randomIndex).c_str())),
        wxT("Błąd"), wxOK | wxICON_ERROR, this);
    }

        // Inkrementacja Licznik i sprawdzenie warunku
        Licznik++;

        break;
    }
```

```
}
```

W przypadku numer 2 - tłumaczenie z polskiego na angielski, pobrany tekst ze zmiennej wxString zamieniamy na typ string, następnie usuwamy znaki spacji oraz zamieniamy wszystkie podane litery na małą nominacje. Następnie porównujemy wpisane słowo do słowa wyświetlanego oraz w zależności od odpowiedzi instrukcją warunkową wyświetlamy wiadomość z odpowiednim komunikatem. Na sam koniec inktementujemy zmienną Licznik.

```
    case 2: {
    // Kod dla case 2

        std::string tekstStd = std::string(tekst.ToStdString());

    // Usuwanie spacji ze slowa podanego przez uzytkownika
    tekstStd.erase(std::remove_if
    (tekstStd.begin(), tekstStd.end(), ::isspace), tekstStd.end());

    // Zamień tekstStd na małe litery
    std::transform(tekstStd.begin(), tekstStd.end(), tekstStd.begin(),
        [](unsigned char c) { return std::tolower(c); });

    if (tekstStd == translator->GetSlowoAng(randomIndex)) {
        wxMessageBox(wxString::FromUTF8("Poprawna odpowiedź! Zdobywasz punkt!"),
        wxT(""), wxOK |wxICON_INFORMATION, this);
            Punkty++;
        } else {
        wxMessageBox(wxString::Format
        (wxT("Niepoprawna odpowiedź! Prawidłowa odpowiedź to: %s"),
        wxString::FromUTF8(translator->GetSlowoAng(randomIndex).c_str())),
        wxT("Błąd"), wxOK | wxICON_ERROR, this);
        }

        // Inkrementacja Licznik i sprawdzenie warunku
        Licznik++;

        break;
    }
```

Dalsza część kodu przycisku odpowiedzialna jest za sprawdzenie czy „fiszka” wyświetliła się odpowiednią ilość razy (Licznik ¡= Liczbaslowek), jeśli tak obecne okno się zamyka i przechodzimy do okna rezultat, natomiast w przeciwnym razie losuje kolejną liczbę (kolejne słowo do translacji) oraz przypisuje ją do zmiennej randomIndex. Następnie program ustawia StaticText1 odpowiedzialny za wyświetlanie słowa do translacji, a także czyści miejsce do wpisywania słowa przez użytkownika.

```
// Warunek poza instrukcją switch
if (Licznik <= Liczbaslowek) {
    // Wspólny kod dla obu przypadków
    randomIndex = rand() % 50;

    switch(wybranyKierunek){
case 1:
        StaticText1->SetWindowStyle(wxALIGN_CENTRE | wxST_NO_AUTORESIZE);
    StaticText1->SetLabel(wxString::FromUTF8
    (translator->GetSlowoAng(randomIndex).c_str()));
    StaticText1->SetWindowStyle(wxALIGN_CENTRE | wxST_NO_AUTORESIZE);

    // Czyszczenie miejsca do wpisania
    TextCtrl1->SetValue("");
```

```
    break;
case 2:
        StaticText1->SetWindowStyle(wxALIGN_CENTRE | wxST_NO_AUTORESIZE);
    StaticText1->SetLabel(wxString::FromUTF8
    (translator->GetSlowoPol(randomIndex).c_str()));
    StaticText1->SetWindowStyle(wxALIGN_CENTRE | wxST_NO_AUTORESIZE);

    // Czyszczenie miejsca do wpisania
    TextCtrl1->SetValue("");
    }

} else {
    // Zamknij bieżące okno dialogowe
    this->EndModal(wxID_CANCEL);

    // Otwórz nowe okno
    Rezultat dlg(0);
    dlg.Center();
    dlg.ShowModal();
}
```

4.4. Plik rezultat.cpp

Wyświetlanie punktów

Pobieramy informacje o ilości dobrych odpowiedzi ze zmiennej Punkty, która była inkrementowana w pliku sprawdz.cpp w przypadku dobrej odpowiedzi oraz pobieramy maksymalną ilość odpowiedzi - Liczbaslowek, a następnie wyświetlamy je.

```
 wxString etykieta = wxString::Format("Punkty: %d/%ld", Punkty, Liczbaslowek);
        StaticText1->SetLabel(etykieta);
```

Przyciski JESZCZE RAZ i WYJDŹ

Przyciski pozwalające rozpocząć rozgrywkę od nowa lub zamknąć aplikacje.

```
void Rezultat::OnButton1Click(wxCommandEvent& event)
{
            // Zamknij bieżące okno dialogowe
                    this->EndModal(wxID_CANCEL);
                    // Otwórz nowe okno
                    PolishYourEnglishDialog dlg(0);
                    dlg.ShowModal();
}

void Rezultat::OnButton2Click(wxCommandEvent& event)
{
 Close();
}
```

5. Wyzwania podczas tworzenia programu

Podczas tworzenia aplikacji do nauki słówek z angielskiego w wxWidgets napotkaliśmy kilka wyzwań. Pierwszym z nich było skuteczne pobieranie i konwersja słówek wprowadzanych przez użytkownika na odpowiednie zmienne, uwzględniając polskie znaki w kodowaniu UTF-8. Odpowiednia obsługa kodowania znaków była kluczowa dla poprawnej funkcji programu, a zarządzanie tym procesem wymagało precyzyjnego kodu. Kolejnym wyzwaniem było efektywne zarządzanie dużym rozmiarem projektu, zwłaszcza kontrolowanie zmiennych między różnymi plikami źródłowymi (.cpp). Skoordynowanie i utrzymanie spójności danych pomiędzy plikami było kluczowe dla prawidłowego działania aplikacji.Dodatkowo, konieczność dynamicznego tworzenia konstruktora w zależności od wyborów użytkownika wprowadziła kolejne wyzwanie. Dostosowanie konstruktora do różnych opcji użytkownika wymagało elastycznego podejścia i umiejętności obsługi skomplikowanych warunków logicznych.

6. Podział pracy

- Interfejs i grafika - Kamil (70%), Bartłomiej (30%)
- Pomysł na implementację - Kamil (60%), Bartłomiej (40%)
- Programowanie Metod - Kamil (70%), Bartłomiej (30%)
- Testowanie - Kamil (30%), Bartłomiej (70%)
- Stworzenie instalatora - Kamil (10%), Bartłomiej (90%)
- Dokumentacja - Kamil (80%), Bartłomiej (20%)
- Poprawki, szlifowanie projektu Kamil (50%), Bartłomiej (50%)