AGH University of Science and Technology

Języki Programowania Obiektowego

Dokumentacja projektowa

LICZNIK STATYSTYK TEKSTU

Autor: Maciej Kapica
Prowadzący: Dr hab. inż. Bogusław Cyganek

15 stycznia 2023

Spis treści

Lista oznaczeń

Poniższa tabela zawiera zestawienie użytych w dokumentacji skrótów oraz ich rozwinięć.

Skrót	Znaczenie
GUI	Graphical User Interface
SDK	Software Development Kit
OOD	Object-Oriented Design
OOP	Object-Oriented Programming
QT	Zestaw przenośnych bibliotek i narzędzi programistycznych
MinGW	Minimalist GNU for Windows
GCC	GNU Compiler Collection

1 Wstęp

Niniejsza dokumentacja dotyczy opracowania systemu zliczającego statystyki plików tekstowych. Celem projektu było stworzenie aplikacji pozwalającej użytkownikowi na zebranie danych o plikach znajdujących się w wybranym przez niego folderze oraz folderach podrzędnych.W tym dokumencie przedstawione zostały opisy działania i funkcjonalności stworzonej aplikacji. W kolejnych rozdziałach zaprezentowano szczegołowe informacje na temat charakterystyki funkcjonalności programu, jego wymagań systemowych oraz sposobu jego realizacji.

2 Wymagania systemowe

W poniższym rozdziale przedstawione zostały wymagania co do projektu systemu jakim jest aplikacja do analizy plików tekstowych. W poniższych podrozdziałach zestawione zostały wstepne założenia projektowe oraz zgodnie z nimi wykorzystane środki służące do realizacji projektu.

2.1 Założenia projektowe

Głównym wymaganiem było aby aplikacja była stworzona do użytku w systemie Windows 10 w języku C++ oraz posiadać GUI. Zatem projekt oparty został o następujące założenia:

- stworzenie syntetycznego opisu działania aplikacji oraz wymaganych funkcjonalności
- opracowanie architektury programu oraz sposobów jego komunikacji z użytkownikiem
- wybór odpowiednich bibliotek dla tworzenia GUI
- wybór adekwatnego dla celów SDK
- stworzenie aplikacji
- przeprowadzenie testów funkcjonalności programu
- opracowanie pełnej dokumentacji projektowej oraz instrukcji obsługi

2.2 Wykorzystane środki

Zgodnie z przedstawionymi wyżej założeniami do realizacji projektu wykorzystany został język C++ oraz środowisko programistyczne *QT*. GUI aplikacji oparte zostało o biblioteki tego środowiska, co za tym idzie korzysta ze stworzonych w nim klas oraz funkcji. Do edycji kodu, debuggingu, kompilacji oraz testów wykorzystana została aplikacja *Qt Creator 9.0.1 (Community)*, natomiast do organizacji plików programu oraz kontroli kompilacji zostało wykorzystane narzędzie *qmake*.

3 Opis funkcjonalności

Zgodnie z założeniami aplikacja posiada zaprogramowane w niej następujące funkcjonalności:

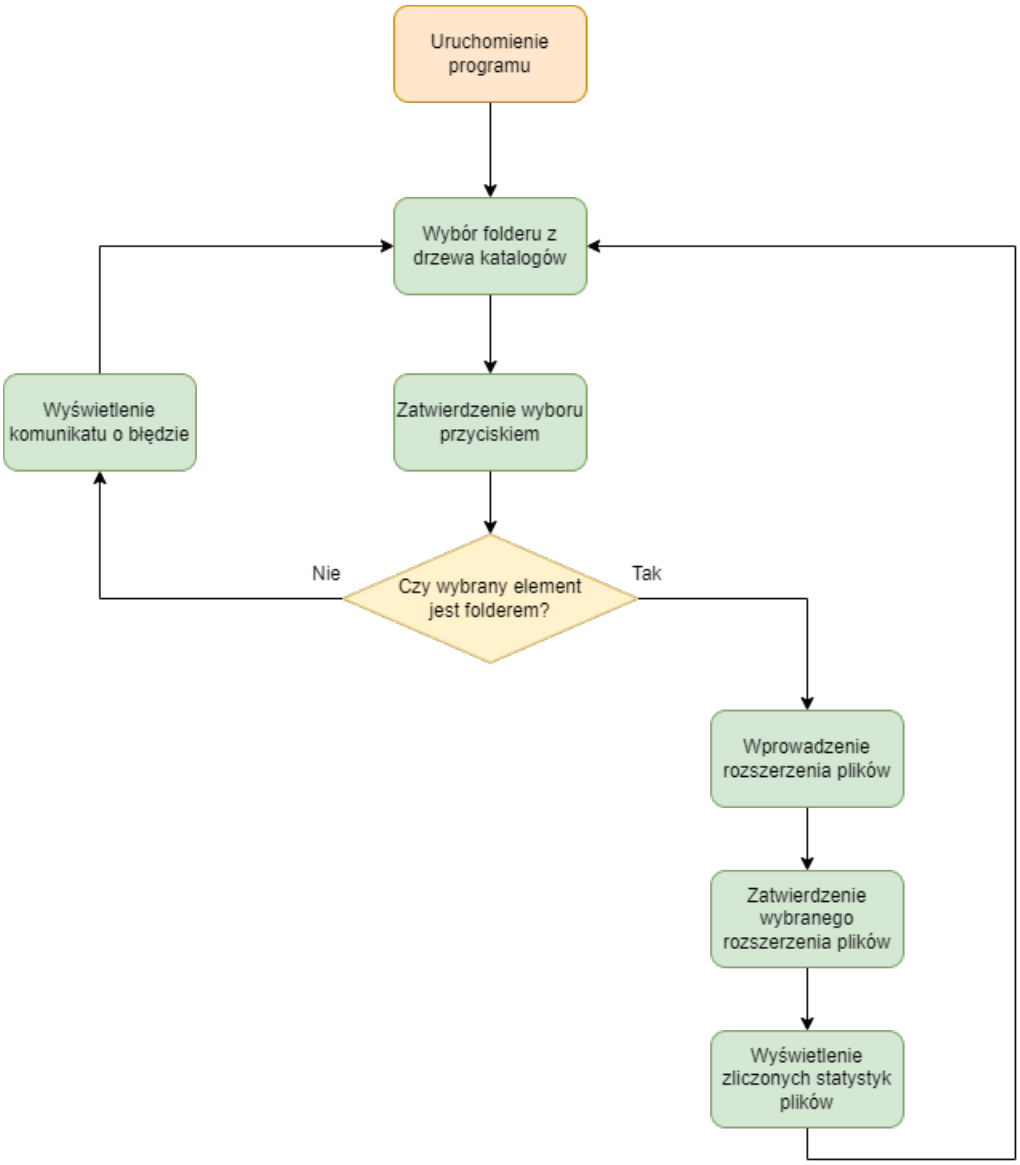
- komunikacja z użytkownikiem poprzez GUI
- wybór katalogu, który ma zostać poddany analizie
- wybór rozszerzenia plików, dla jakich mają być zliczone statystyki
- liczenie statystyk plików tekstowych takich jak: liczba wyrazów, liczba znaków, ilość lini
- sumowanie statystyk dla wszystkich plików znajdujących się w wybranym folderze oraz w folderach podrzędnych
- prezentacja zliczonych statystyk użytkownikowi w osobnym oknie dialogowym

4 Projekt techniczny

W niniejszym rozdziale szczegółowo przedstawione zostały aspekty techniczne realizacji projektu. W poniższych podrozdziałach zaprezentowane zostały schematy oraz tabele dotyczące działania aplikacji w oparciu o założenia projektowe, wykorzystane środki oraz wymagane funkcjonalności.

4.1 Diagram UML działania aplikacji

Na poniższej grafice przedstawiony został schemat ogólny działania aplikacji.



4.2 Klasy i obiekty

W niżej przedstawionych tabelach zestawione zostały klasy oraz metody biblioteki QT wykorzystane w projekcie. W oparciu o to zbudowana została aplikacja, dzięki czemu w czytelny posób mogła zostać ona oparta o GUI.

Hierarchia klas wykorzystanych w tym programie jest następująca:
QApplication jest klasą podstawową dla wszystkich aplikacji napisanych z użyciem biblioteki QT. QTreeView, QFileSystemModel, QPushButton i QVBoxLayout są klasami pochodnymi z QWidget. QFileSystemModel jest modelem dla systemu plików, który jest używany przez QTreeView do pobierania informacji o plikach i katalogach. W tym programie QApplication jest jedynym obiektem tej klasy i jest on wykorzystywany przez całą aplikację.
QTreeView jest skojarzony z modelem dla systemu plików (QFileSystemModel) przez ustawienie odpowiedniego modelu dla widoku. QPushButton jest skojarzony z funkcją analizującą przez podłączenie sygnału "clicked"do odpowiedniej funkcji. Natomiast QVBoxLayout jest skojarzony z widokiem drzewa (QTreeView) oraz przyciskiem (QPushButton) poprzez dodanie ich do layoutu okna aplikacji.

Tabela klas oraz relacji między nimi:

Klasa	**Opis**	**Relacje z innymi klasami**
QApplication	Klasa podstawowa dla wszystkich aplikacji napisanych z użyciem biblioteki QT.	Jeden obiekt jest wykorzystywany przez całą aplikację.
QTreeView	Klasa widoku drzewa, używana do wyświetlania i przeglądania katalogów.	Skojarzona z modelem dla systemu plików (QFileSystemModel) przez ustawienie odpowiedniego modelu dla widoku.
QFileSystemModel	Model dla systemu plików, używany do pobierania informacji o plikach i katalogach.	Skojarzona z klasą QTreeView.
QPushButton	Klasa przycisku, używana do uruchomienia analizy folderu	Skojarzona z funkcją analizującą przez podłączenie sygnału "clicked"do odpowiedniej funkcji.
QVBoxLayout	Klasa layoutu, używana do układania widoków w oknie aplikacji	Skojarzona z widokiem drzewa (QTreeView) oraz przyciskiem (QPushButton) poprzez dodanie ich do layoutu.

Tabela wykorzystanych metod:

Klasa	**Metoda**	**Opis**
QApplication	exec()	Uruchamia główną pętlę aplikacji.
	setAttribute()	Ustawia atrybut dla aplikacji.
	processEvents()	Przetwarza zdarzenia systemowe.
QTreeView	setModel()	Ustawia model dla widoku drzewa.
	setRootIndex()	Ustawia indeks katalogu głównego.
	currentIndex()	Zwraca indeks aktualnie wybranego elementu.
QFileSystemModel	setRootPath()	Ustawia ścieżkę katalogu głównego.
	index()	Zwraca indeks dla danej ścieżki.
	isDir()	Sprawdza czy dany indeks jest katalogiem.
	getFiles()	Zwraca listę plików dla danego indeksu.
	filePath()	Zwraca pełną ścieżkę dla danego indeksu.

5 Opis realizacji

W tym rozdziale został przedstawiony szczegółowy opis środowiska programistycznego, wykorzystanych narzędzi oraz samej implementacji architektury programu. W kolejnych podrozdziałach tematy te zostały rozwinięte.

5.1 Środowisko programistyczne

Qt wykorzystane do stworzenia aplikacji to biblioteka narzędzi i frameworki programistyczne zapewniające możliwość tworzenia interfejsu graficznego użytkownika, a także innych funkcjonalności dla aplikacji konsolowych, internetowych, mobilnych, bazodanowych, itp.

Środowisko programistyczne QT składa się z kilku elementów:

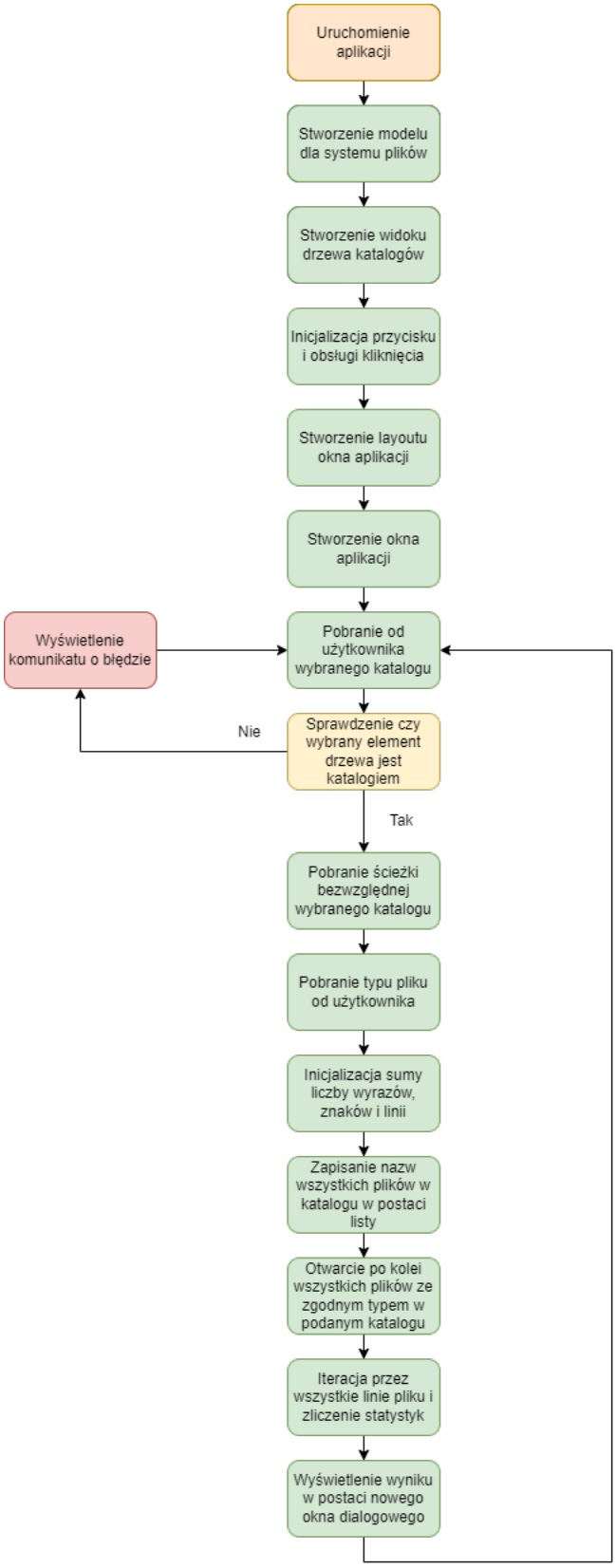
- Biblioteka QT - główna biblioteka zawierająca klasy i metody do tworzenia aplikacji z interfejsem graficznym oraz innymi funkcjonalnościami.
- Qt Creator - narzędzie do tworzenia aplikacji z interfejsem graficznym. Qt Creator jest środowiskiem programistycznym opartym na Eclipse, które zawiera edytor kodu, narzędzia do debugowania, tworzenie interfejsu graficznego oraz narzędzia do tworzenia i uruchamiania projektów.
- Qt Designer - narzędzie do projektowania interfejsu graficznego, pozwala na tworzenie i edycję okien, przycisków, menu, itp.

5.2 Kompilator

Do kompilacji stworzonego kodu aplikacji wykorzystany został MinGW. Jest to darmowa implementacja kompilatora GCC dla systemów Windows. Jest on przeznaczony do kompilowania programów napisanych w języku C, C++, Fortran i Ada. MinGW jest przenośnym i darmowym zestawem narzędzi, które pozwala na tworzenie natywnych aplikacji Windows bez potrzeby użycia Cygwin. MinGW zawiera kompilator GCC, narzędzia do budowy aplikacji, biblioteki, narzędzia do debugowania i inne narzędzia potrzebne do kompilacji i uruchamiania aplikacji. Kompilator MinGW jest kompatybilny z większością popularnych systemów operacyjnych Windows, w tym Windows XP, Windows 7, Windows 8 i Windows 10.

5.3 Schemat działania programu

Na poniższym diagramie został przedstawiony schemat w jaki sposób program realizuje założenia projektu.



6 Instrukcja obsługi

Z racji, że projekt został stworzony w środowisku QT, aby móc uruchomić aplikację wymagany jest program *Qt Creator 9.0.1 (Community).*

Aby uruchomić projekt należy:

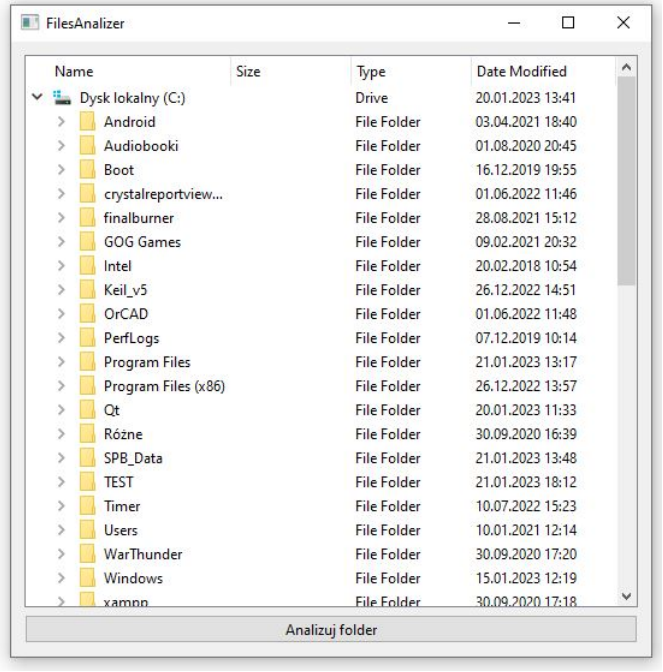
- Sklonować repozytorium na dysk lokalny
- Uruchomić program Qt Creator
- Z menu programu wybrać opcję Open Project
- Wybrać w oknie dialogowym plik .../TextAnalizer/FilesAnalizer/FilesAnalizer.pro
- Otworzy się wtedy cała struktura projektu
- Następnie wybrać z menu: Budowanie -> Uruchom (lub skrót klawiszowy CTR + R)
- Uruchomiona zostanie aplikacja TextAnalizer

6.1 Obsługa programu

W tym podrozdziale zostanie przedstawiona obsługa aplikacji krok po kroku wraz załączonymi zrzutami ekranu.

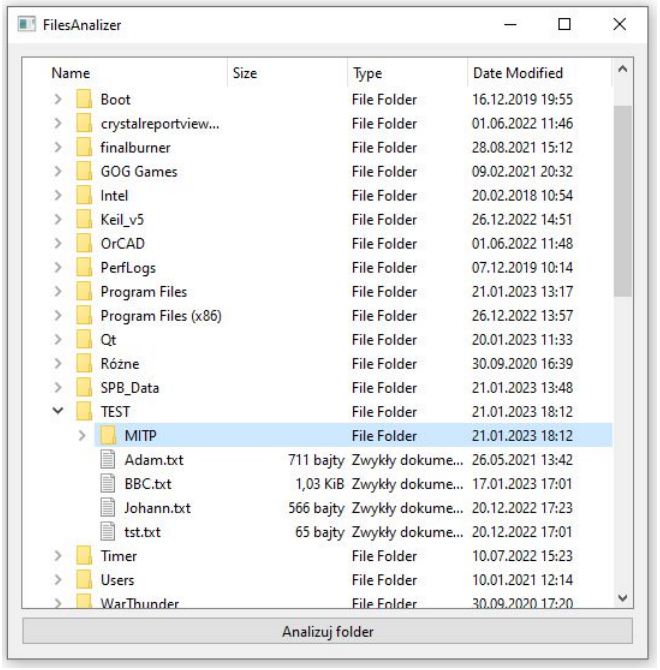
6.1.1 Okno główne

Po uruchomieniu aplikacji pojawi się okno główne wyświetlające drzewo katalogów komputeraz lokalnego



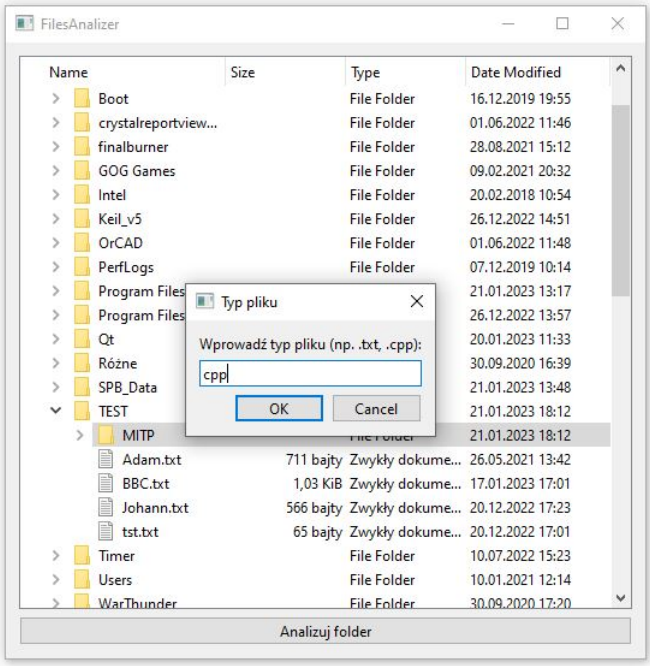
6.1.2 Wybór katalogu

Następnie użytkownik ma mozliwość przeglądania drzewa katalogów oraz wybór konkretnego folderu do analizy poprzez pojedyncze kliknięcie na niego lewym przyciskiem myszy.



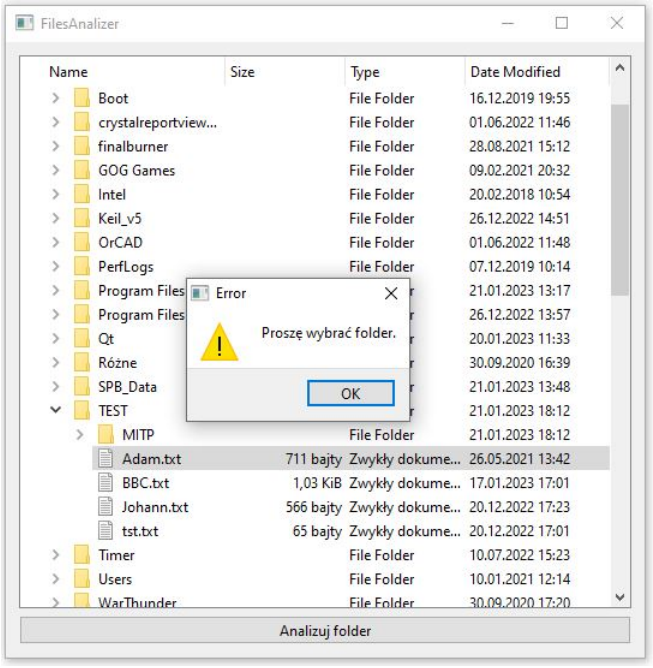
6.1.3 Zatwierdzenie wyboru oraz wprowadzenie rozszerzenia plików

Następnie użytkownik może zatwierdzić wybór katalogu poprzez kliknięcie przycisku *Analizuj folder*. Po wykonaniu tej czynności na ekranie pokaże się nowe okno dialogowe, w którym użytkonik może wprowadzić roszerzenie plików, których statystyki chce zobaczyć.



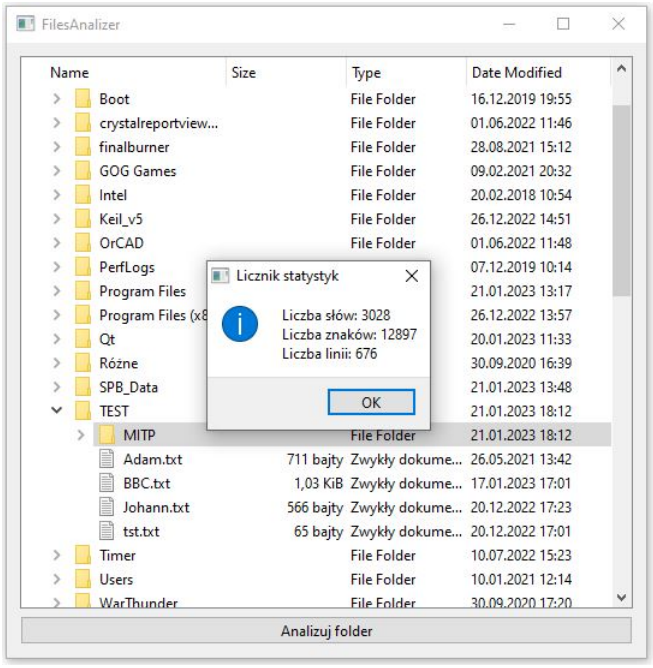
6.1.4 Wybór pliku zamiast katalogu

W razie wyboru przez użytkownika pliku zamiast katalogu program poinformuje go o tym wyświetlając error w nowym oknie dialogowym.



6.1.5 Prezentacja statystyk

Gdy użytkownik wybrał folder, wprowadził żądane rozszerzenie plików do analizy i zatwierdził to przyciskiem *OK* wyświetlone zostaną zsumowane statystyki plików o podanym typie w wybranym katalogu oraz w katalogach podrzędnych.



7 GitHub

Repozytorium projektu zostało umieszczone w serwisie GitHub.
Można je znaleźć pod linkiem: https://github.com/maciejkapica/JPO$_{PROJECT_TEXTANALIZER}$

8 Bibliografia

1. Guillaume Lazar, Robin Penea: *Mastering Qt 5: Create Stunning Cross-platform Applications Using C++ with Qt Widgets and QML with Qt Quick, 2nd Edition*, Packt Publishing 2018
2. https://wiki.qt.io
3. https://www.ibm.com/docs/en/i/7.3?topic=languages-c-c