

Kalkulator dla elektroników/informatyków

Z przedmiotu: Języki programowania obiektowego

Autor: Kacper Janicki

Kierunek: Elektronika i Telekomunikacja

Spis treści:

1.	WSTĘP.....	3
2.	FUNKcjONALNOŚĆ (<i>FUNCTIONALITY</i>).....	4
3.	ANALIZA PROBLEMU (<i>PROBLEM ANALYSIS</i>).....	5
4.	PROJEKT TECHNICZNY (<i>TECHNICAL DESIGN</i>).....	6
5.	OPIS REALIZACJI (<i>IMPLEMENTATION REPORT</i>).....	10
6.	OPIS WYKONANYCH TESTÓW (<i>TESTING REPORT</i>)	11
7.	PODRĘCZNIK UŻYTKOWNIKA (<i>USER'S MANUAL</i>).....	12
8.	BIBLIOGRAFIA.....	14

Lista oznaczeń:

MFC	<i>Microsoft Foundation Class Library</i>
-----	---

1.Wstęp:

Dokument dotyczy projektu kalkulatora dla elektroników/informatyków. Kalkulator ten został zaprojektowany z myślą o łatwej i szybkiej konwersji między różnymi systemami liczbowymi, takimi jak dziesiętny (decimal), binarny (binary), ósemkowy (octal) i szesnastkowy (hexadecimal). W projekcie został zaimplementowany prosty interfejs użytkownika za pomocą biblioteki MFC. Całość została wykonana w języku programowania C++ w środowisku Visual Studio 2022.

2.FUNKCJONALNOŚĆ (*FUNCTIONALITY*):

Kalkulator oferuje możliwość przeliczania liczb między czterema głównymi systemami liczbowymi: dziesiętnym, binarnym, ósemkowym i szesnastkowym. Użytkownik może wprowadzić liczbę w jednym systemie, a kalkulator natychmiast przeliczy ją na pozostałe trzy systemy. Interfejs jest prosty i intuicyjny, co pozwala na szybkie i łatwe przeliczanie liczb, obsługuje duże liczby. W razie błędnie wprowadzonych danych wyskoczy prosty i zrozumiały komunikat, dzięki któremu użytkownik będzie mógł zmienić dane na poprawne. Więcej informacji odnośnie komunikatów znajduje się w punkcie numer 7, czyli w podręczniku użytkownika (user's manual) na stronie numer 12 .

3. ANALIZA PROBLEMU (*PROBLEM ANALYSIS*):

Praca z różnymi systemami liczbowymi jest nieodłącznym elementem pracy elektroników i informatyków. Często muszą oni przeliczać liczby z jednego systemu na inny, co może być czasochłonne i narażone na błędy, zwłaszcza przy ręcznym przeliczaniu.

Problem 1: Czasochłonność

Ręczne przeliczanie liczb między różnymi systemami liczbowymi może być czasochłonne. Dla dużych liczb lub skomplikowanych konwersji, proces ten może zająć dużo czasu, który mógłby być lepiej wykorzystany na inne zadania.

Problem 2: Możliwość błędów

Ręczne przeliczanie liczb jest narażone na błędy. Nawet drobne pomyłki mogą prowadzić do znacznych błędów w wynikach.

Kalkulator został zaprojektowany, aby rozwiązać te problemy, oferując szybko, dokładną i łatwą w użyciu platformę do przeliczania liczb między różnymi systemami liczbowymi. Dzięki temu, użytkownicy mogą skupić się na swojej pracy, zamiast spędzać czas na ręcznym przeliczaniu liczb.

4.PROJEKT TECHNICZNY (TECHNICAL DESIGN):

Za pomocą programu doxygen, wygenerowano listę wszystkich występujących klas w programie oraz występujących w klasach funkcji. Program doxygen ułatwił zrozumienie struktury projektu kalkulatora.

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:


 CAboutDlg	
 CCalcForEleApp	
 CCalcForEleDlg	
 ZamianaLiczb	

Rysunek 1. Lista klas w projekcie

Diagram dla poszczególnych klas:

CAboutDlg Class Reference

Inheritance diagram for CAboutDlg:



```
graph BT; CDialogEx --> CAboutDlg
```

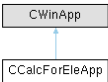
Protected Member Functions

virtual void DoDataExchange (CDataExchange *pDX)

Rysunek 2 Diagram dla CAboutDlg.

CCalcForEleApp Class Reference

Inheritance diagram for CCalcForEleApp:



```
graph BT; CWinApp --> CCalcForEleApp
```

Public Member Functions

virtual BOOL InitInstance ()

Rysunek 3Diagram dla CCalcForEleApp.

CCalcForEleDlg Class Reference

Inheritance diagram for CCalcForEleDlg:

CDialogEx

CCalcForEleDlg

Public Member Functions

CCalcForEleDlg (CWnd *pParent=NULLPTR)

afx_msg void OnCbnSelchangeCombo1 ()

afx_msg void OnEnChangeEdit4 ()

afx_msg void OnEnChangeEdit3 ()

afx_msg void OnEnChangeEdit6 ()

afx_msg void OnBnClickedButton1 ()

afx_msg void OnBnClickedButton2 ()

afx_msg void OnEnChangeEdit2 ()

Public Attributes

CString binary

CString decimal

CString octal

CString hexadecimal

Protected Member Functions

virtual void DoDataExchange (CDataExchange *pDX)

virtual BOOL OnInitDialog ()

afx_msg void OnSysCommand (UINT nID, LPARAM lParam)

afx_msg void OnPaint ()

afx_msg HCURSOR OnQueryDragIcon ()

Rysunek 4 Diagram dla CCalcForEleDlg.

ZamianaLiczb Class Reference

Public Member Functions

bool isValidDecimal (const std::string &str)

bool isValidBinary (const std::string &str)

bool isValidOctal (const std::string &str)

bool isValidHexadecimal (const std::string &str)

Static Public Member Functions

static std::string decimalToBinary (std::string num)

static std::string decimalToOctal (std::string num)

static std::string decimalToHexadecimal (std::string num)

static std::string hexadecimalToDecimal (const std::string &hex)

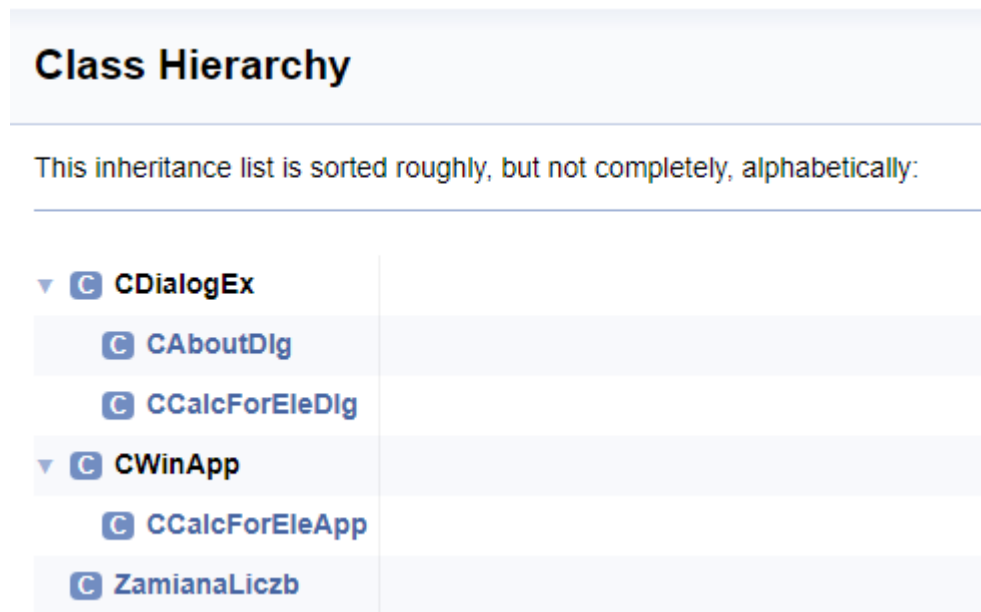
static std::string octalToDecimal (const std::string &octal)

static std::string binaryToDecimal (const std::string &binary)

Rysunek 5 Lista funkcji w klasie ZamianaLicz.

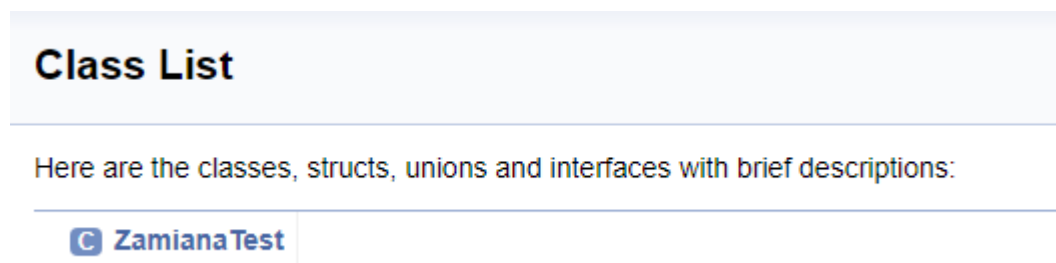
7 z 14

Hierarchia Klas w projekcie kalkulatora:



Rysunek 6 Hierarchia klas.

Została również wygenerowana lista klas oraz funkcji w niej istniejących dla google testu, sprawdzającego poprawność działania funkcji dzięki któremu kalkulator zamienia naszą liczbę w liczby w innych systemach liczbowych.



Rysunek 7 Lista klas w Gtest.

ZamianaTest Class Reference

Public Member Functions

bool	isValidDecimal	(const std::string &str)
bool	isValidBinary	(const std::string &str)
bool	isValidOctal	(const std::string &str)
bool	isValidHexadecimal	(const std::string &str)

Static Public Member Functions

static std::string	decimalToBinary	(std::string num)
static std::string	decimalToOctal	(std::string num)
static std::string	decimalToHexadecimal	(std::string num)
static std::string	hexadecimalToDecimal	(const std::string &hex)
static std::string	octalToDecimal	(const std::string &octal)
static std::string	binaryToDecimal	(const std::string &binary)

Rysunek 8 Lista funkcji w klasie z Googletestu.

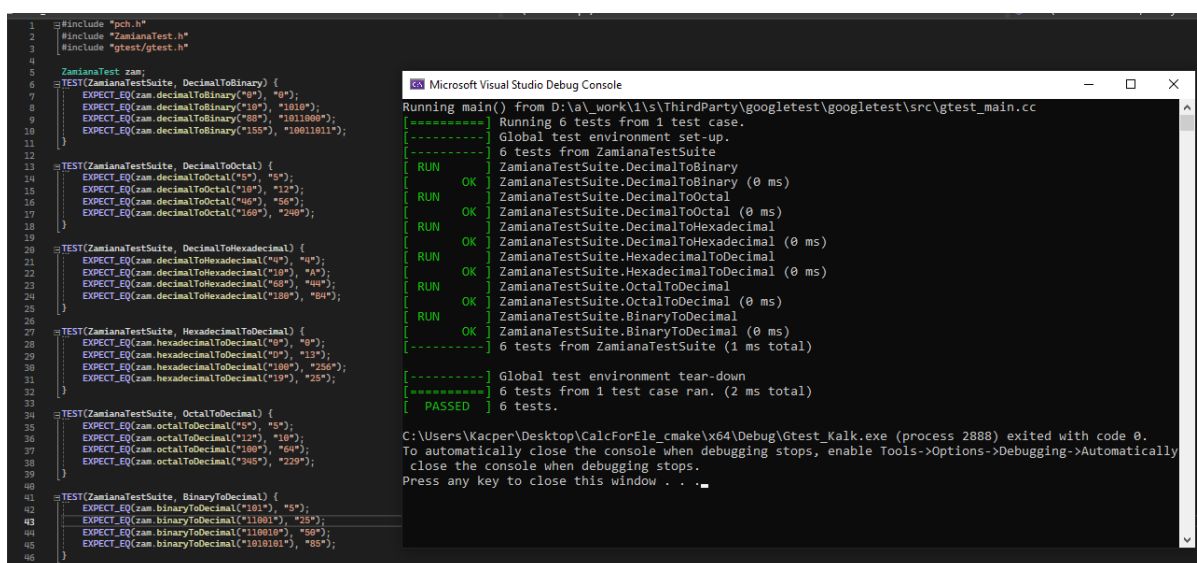
5. OPIS REALIZACJI (*IMPLEMENTATION REPORT*):

Wykorzystane narzędzia w projekcie:

- **Visual Studio 2022:** Wybrano to środowisko ze względu na jego wszechstronność i bogactwo funkcji, w tym bibliotekę MFC, co czyni go optymalnym wyborem dla tego projektu.
- **Debugger w Visual Studio 2022:** Umożliwił analizę, monitorowanie i rozwiązywanie problemów w kodzie podczas debugowania. Dzięki wyświetlanym błędom, możliwe było szybkie znalezienie i naprawienie błędów w kodzie.
- **CMake:** Utworzono plik CMakeLists, który umożliwia generowanie projektu na dowolnym komputerze. CMake jest szeroko stosowany w dużych projektach do budowania projektów na różnych systemach/platformach.
- **GitHub:** Wszystkie pliki projektu zostały umieszczone na tej platformie.
- **Google Test:** Za pomocą tego narzędzia przeprowadzono testy funkcji przeliczających liczby na inne systemy liczbowe. Pozwoliło to upewnić się, że kalkulator działa poprawnie we wszystkich przypadkach.
- **Doxygen:** Ten program umożliwił przedstawienie listy, hierarchii klas oraz funkcji występujących w tych klasach.

6.OPIS WYKONANYCH TESTÓW (*TESTING REPORT*):

Zostały przeprowadzone testy GTest, które sprawdzają poprawność działania poszczególnych funkcji konwercji liczb. Poniżej przedstawiono zrzut ekranu, na którym można zaobserwować kod wspomnianego GTestu oraz po prawej stronie wynik uruchomienia tego kodu w środowisku Visual Studio 2022. Przeprowadzone testy przebiegły w sposób pomyślny bez większych problemów. Dlatego można założyć, że funkcje przemiany liczb działają poprawnie.



```
1 #include "pch.h"
2 #include "ZamianaTest.h"
3 #include "gtest/gtest.h"
4
5 ZamianaTest zam;
6 TEST(ZamianaTestSuite, DecimalToBinary) {
7     EXPECT_EQ(zam.decimalToBinary("0"), "0");
8     EXPECT_EQ(zam.decimalToBinary("10"), "1010");
9     EXPECT_EQ(zam.decimalToBinary("80"), "1011000");
10    EXPECT_EQ(zam.decimalToBinary("155"), "10011011");
11}
12
13 TEST(ZamianaTestSuite, DecimalToOctal) {
14    EXPECT_EQ(zam.decimalToOctal("0"), "0");
15    EXPECT_EQ(zam.decimalToOctal("10"), "12");
16    EXPECT_EQ(zam.decimalToOctal("40"), "50");
17    EXPECT_EQ(zam.decimalToOctal("160"), "240");
18}
19
20 TEST(ZamianaTestSuite, DecimalToHexadecimal) {
21    EXPECT_EQ(zam.decimalToHexadecimal("0"), "0");
22    EXPECT_EQ(zam.decimalToHexadecimal("10"), "A");
23    EXPECT_EQ(zam.decimalToHexadecimal("68"), "44");
24    EXPECT_EQ(zam.decimalToHexadecimal("100"), "64");
25}
26
27 TEST(ZamianaTestSuite, HexadecimalToDecimal) {
28    EXPECT_EQ(zam.hexadecimalToDecimal("0"), "0");
29    EXPECT_EQ(zam.hexadecimalToDecimal("A"), "10");
30    EXPECT_EQ(zam.hexadecimalToDecimal("100"), "256");
31    EXPECT_EQ(zam.hexadecimalToDecimal("19"), "25");
32}
33
34 TEST(ZamianaTestSuite, OctalToDecimal) {
35    EXPECT_EQ(zam.octalToDecimal("0"), "0");
36    EXPECT_EQ(zam.octalToDecimal("10"), "8");
37    EXPECT_EQ(zam.octalToDecimal("100"), "64");
38    EXPECT_EQ(zam.octalToDecimal("345"), "229");
39}
40
41 TEST(ZamianaTestSuite, BinaryToDecimal) {
42    EXPECT_EQ(zam.binaryToDecimal("101"), "5");
43    EXPECT_EQ(zam.binaryToDecimal("11001"), "25");
44    EXPECT_EQ(zam.binaryToDecimal("1100101"), "59");
45    EXPECT_EQ(zam.binaryToDecimal("11010101"), "85");
46}
```

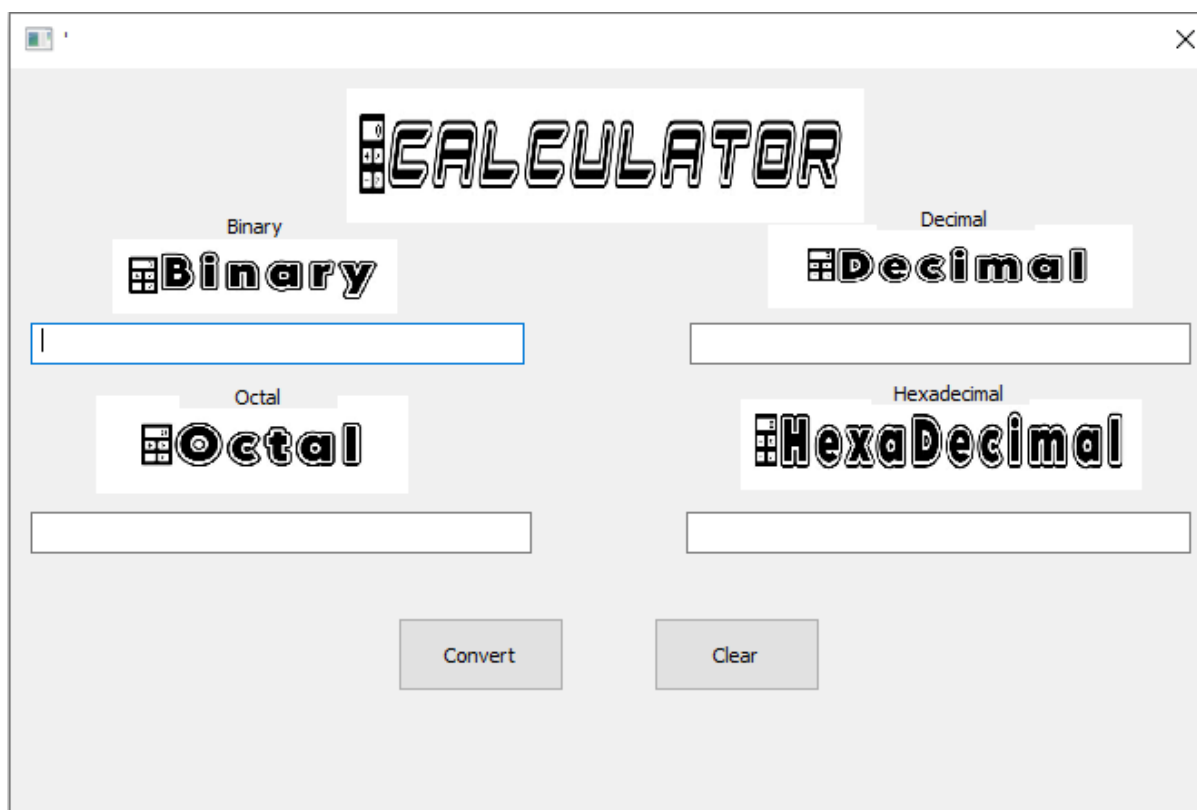
```
Running main() from D:\a_work\1\ThirdParty\googletest\googletest\src\gtest_main.cc
[*****] Running 6 tests from 1 test case.
[*****] Global test environment set-up.
[*****] 6 tests from ZamianaTestSuite
[ RUN ] ZamianaTestSuite.DecimalToBinary (0 ms)
[ OK ] ZamianaTestSuite.DecimalToBinary (0 ms)
[ RUN ] ZamianaTestSuite.DecimalToOctal (0 ms)
[ OK ] ZamianaTestSuite.DecimalToOctal (0 ms)
[ RUN ] ZamianaTestSuite.DecimalToHexadecimal (0 ms)
[ OK ] ZamianaTestSuite.DecimalToHexadecimal (0 ms)
[ RUN ] ZamianaTestSuite.HexadecimalToDecimal (0 ms)
[ OK ] ZamianaTestSuite.HexadecimalToDecimal (0 ms)
[ RUN ] ZamianaTestSuite.OctalToDecimal (0 ms)
[ OK ] ZamianaTestSuite.OctalToDecimal (0 ms)
[ RUN ] ZamianaTestSuite.BinaryToDecimal (0 ms)
[ OK ] ZamianaTestSuite.BinaryToDecimal (0 ms)
[*****] 6 tests from ZamianaTestSuite (1 ms total)
[*****] Global test environment tear-down
[*****] 6 tests from 1 test case ran. (2 ms total)
[ PASSED ] 6 tests.

C:\Users\Kacper\Desktop\CalcForEle_cmake\x64\Debug\Gtest_Kalk.exe (process 2888) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically
close the console when debugging stops.
Press any key to close this window . . .
```

Rysunek 9 Zrzut ekranu kodu oraz wyniku Google test.

7. Podręcznik użytkownika (*user's manual*):

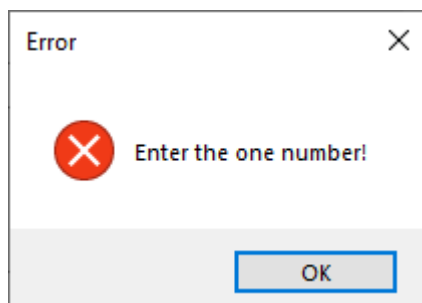
Poniżej znajduje się zrzut ekranu interfejsu użytkownika wykonanego za pomocą biblioteki MFC. Można na nim zauważyć cztery miejsca na wpisanie liczby, którą chcemy przeliczyć na pozostałe systemy liczbowe. Wystarczy wpisać znaną nam liczbę w odpowiednim miejscu, następnie kliknąć przycisk „Convert”, a kalkulator w bardzo szybkim tempie przeliczy naszą liczbę na wszystkie inne systemy. Po przeliczeniu wystarczy nacisnąć przycisk „Clear”, który ma za zadanie wyczyścić cały kalkulator, aby móc wpisać następną liczbę do przeliczenia. W razie błędnego wpisania wartości danej liczby wyskoczą komunikaty, dzięki którym będzie można poprawić naszą liczbę.



Rysunek 10 Interfejs użytkownika.

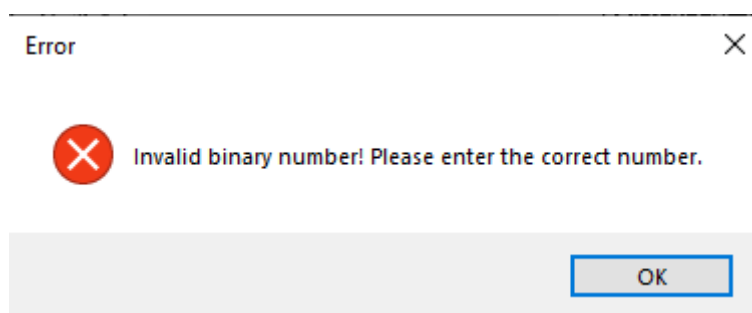
Komunikaty oraz błędy, które mogą wyskoczyć w razie złego wpisania liczby:

-Error numer 1 występuje gdy albo żadna wartość nie została wpisana albo gdy wpisano wartość do więcej niż jednego okienka.



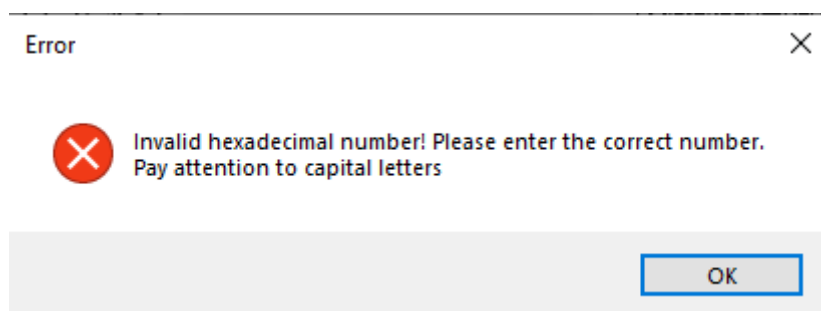
Rysunek 11 Error numer 1.

-Error numer 2 wyskakuje gdy, została wpisana zła wartość w jakieś okienko.



Rysunek 12 Error numer 2

-Error numer 3 wyskakuje gdy w okienku dla hexadecimal jest wpisana zła wartość, albo zostały użyte małe literki.



Rysunek 13 Error numer 3.

Bibliografia:

- [1] Cyganek B.: Programowanie w języku C++. Wprowadzenie dla inżynierów. PWN, 2023.
- [2] <https://www.geeksforgeeks.org/> - Strona Geeks for geeks
- [3] <https://stackoverflow.com/> - Strona Stackoverflow
- [4] <https://github.com/> - GitHub
- [5] <https://gitlab.kitware.com/cmake/community/-/wikis/FAQ#how-to-use-mfc-with-cmake>
- [6] <https://chat.openai.com> – Chat GPT