**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

Kalkulator

dla elektroników

|  |  |  |
| --- | --- | --- |
| REV | DATA | ZMIANY |
| 1.1 | 24.06.2023 | Miłosz Sokół (miloszsokol@student.agh.edu.pl) |
|  |  |  |
|  |  |  |
|  |  |  |

Autor: Miłosz Sokół

Akademia Górniczo-Hutnicza

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**Spis treści:**

1. Wstęp 4
2. Podstawowe założenia projektu 5
3. Funkcjonalność / działanie kalkulatora 6
   1. Systemy liczbowe 7
   2. Obliczanie wartości prądu, napięcia i rezystancji 8
   3. Obliczanie rezystora LED 9
   4. Dzielnik prądu oraz napięcia 10
4. Analiza problemu 11
5. Projekt techniczny 12
   1. Uwagi wstępne 12
   2. Przeliczaniepomiędzy systemami liczbowymi / klasa Conversion 13
   3. Podstawowe obliczenia dla elektronika / klasa Calculations 15
   4. Zabezpieczenia 17
6. Opis realizacji 19
7. Podręcznik użytkownika 20

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**Lista oznaczeń**

|  |  |
| --- | --- |
| HEX | Szesnastkowy |
| DEC | Dziesiętny |
| OCT | Ósemkowy |
| BIN | Dwójkowy(binarny) |
| LED | Light-Emitting Diode |

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

1. **Wstęp**

Niniejszy dokument jest opracowaniem projektu „Kalkulator dla elektroników”. Celem tego systemu jest ułatwienie pracy dla elektronika bądź informatyka poprzez oferowanie urozmaiconych funkcji, które umożliwia wykonywanie podstawowych operacji, które są niezbędne przy pracy elektronika.

Projekt jest głownie skierowany do osób rozpoczynających swoja przygodę z elektronika/informatyka ponieważ jego zakres pracy obejmuje podstawa funkcjonalność niezbędna do prawidłowego poznania podstaw podczas nauki.

Oferuje miedzy innymi możliwość przeliczania systemów liczbowych jak i obliczanie wartości prądu oraz napiec dla prostych dzielników napięcia oraz prądu. Są to podstawowe zagadnienie jakie może wykonać osoba rozpoczynająca swoje przygody z elektroniką.

Może być pomocny do konstrukcji prostych obwodów zawierających takie elementy jak rezystory oraz LEDy. Usprawni to projektowanie swoich pierwszych układów, dzięki którym szybko zdobędzie się pożądana wiedze, która posłuży w przyszłości do rozbudowy swoich projektów jak i budowy tych trudniejszych/ambitniejszych.

Sam kalkulator został wykonany w dosyć prosty sposób tak aby każdy kto chce lub ma możliwość mógł dostosować go do swoich naukowych potrzeb. Jego interfejs jest prosty i czytelny co ułatwi szybkie poruszanie się po jego funkcjach bez potrzeby „przeklikiwania” tysiąca opcji

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

1. **Podstawowe założenia projektu**

Głównym założeniem projektu było zaprojektowanie, napisanie kalkulatora w języku c++.

Cały kalkulator miał być prosty i przejrzysty oraz oferować podstawowe opcje przydatne w pracy elektronika takie jak:

* Konwersja systemów liczbowych(np. z HEX na DEC lub BIN na OCT)
* Wykonywanie podstawowych operacji służących do obliczeń rezystancji, prądu lub napięcia
* Obliczanie spadków napiec na rezystorach na tak zwanym dzielniku napięcia
* Obliczanie prądów na rezystorach na tak zwanym dzielniku prądu
* Wyznaczanie wartości rezystora dla LED w zależności od napięcia zasilania, maksymalnego prądu oraz koloru diody

Cały kalkulator miał być napisany w konsoli aby nie komplikować zbytnio projektu oraz stworzyć go prostym w obsłudze i dać szerokie możliwości do przyszłej rozbudowy.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

1. **Funkcjonalność/działanie kalkulatora**

**Kalku**lator oferuje wszystkie opcje które zostały przedstawione w założeniach.

Jego MENU wygląda następująco:

"1. Przeliczanie systemów liczbowych"

"2. Obliczanie podstawowych wartości prądu, napięcia i rezystancji

"3. Obliczanie wartości rezystora dla LED"

"4. Dzielnik napięcia oraz prądu"

"5. Wyjście z kalkulatora"

A schemat graficzny wygląda następująco:

Obraz zawierający tekst, zrzut ekranu, Czcionka, diagram

Opis wygenerowany automatycznie

Użytkownik poprzez wpisanie odpowiedniej cyfry ma możliwość wybrania odpowiedniej opcji kalkulatora.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

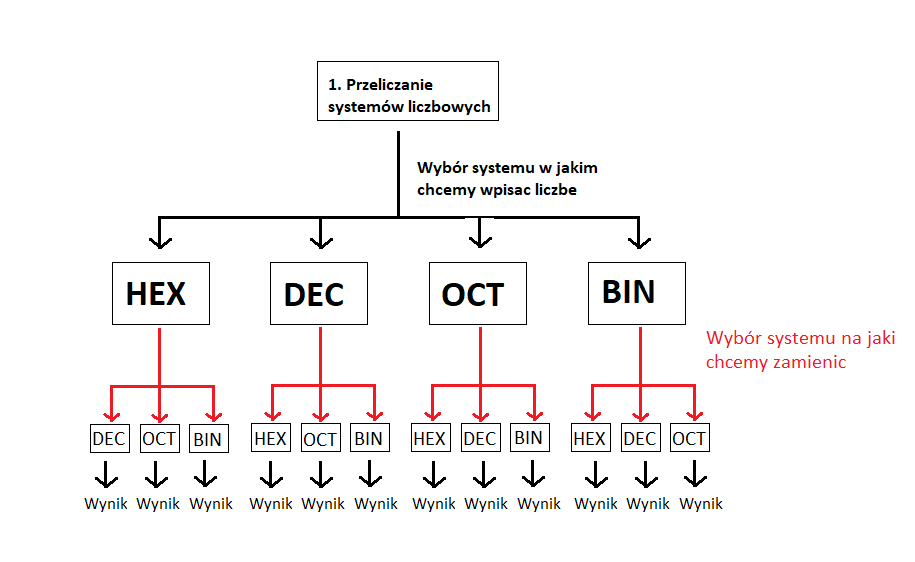
**Technology**

**3.1 Systemy liczbowe**

Najważniejsza z nich jest możliwość przeliczania liczb z jednego systemu liczbowego na drugi. Jest to pierwsza opcja w MENU kalkulatora zaraz po uruchomieniu

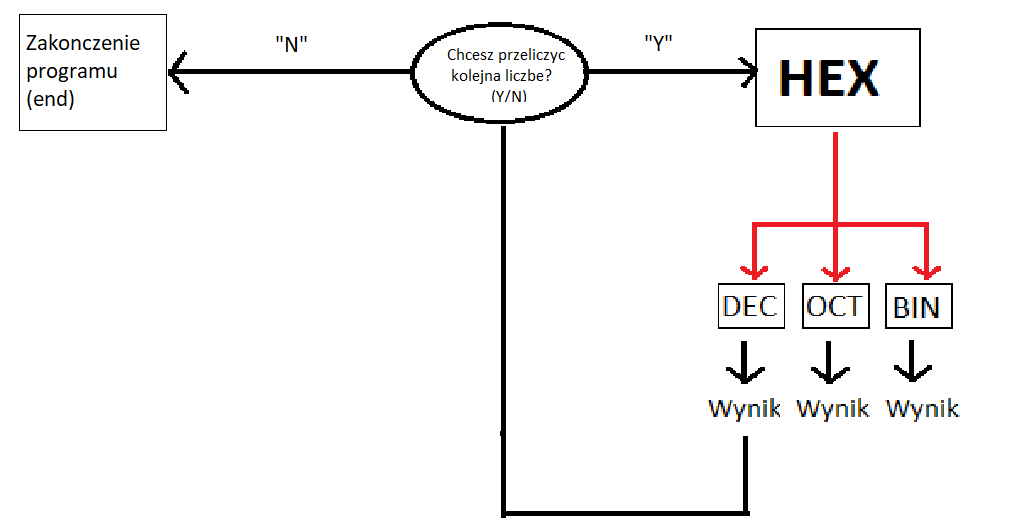
Kalkulator daje nam możliwość wybrania systemu w jakim posiadamy liczbę oraz przeliczenia jej na dowolny inny (dokładniej mówiąc na system HEX, DEC, OCT lub BIN)

Zaprojektowany system wygląda następująco:



Po wpisaniu liczby i wybraniu systemu, w którym chcemy otrzymać liczbę otrzymujemy wynik.

Teraz pojawi się komunikat, który zapyta czy chcemy przeliczyć jeszcze raz kolejna liczbę albo wyjść z kalkulatora:



**Technical Report** **24.06.2023r**

**AGH University of Science and**

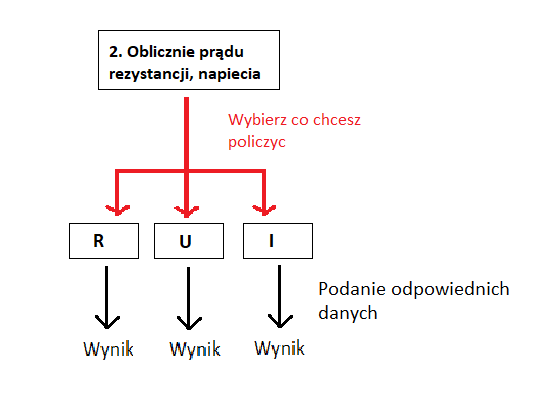
**Technology**

**3.2 Obliczanie prądu, rezystancji i napiecie**

Możliwość wyliczenia rezystancji, prądu lub napięcia jest druga opcja w MENU kalkulatora.

Wystarczy wybrać odpowiednia opcje (1 z 3), która pozwoli nam obliczyć prąd, napięcie oraz rezystancje zgodnie ze wzorem U = R \* I, gdzie:

* U – napięcie
* R- rezystancja
* I -prąd



**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**3.3 Obliczanie rezystora dla LED**

Jedna z ważniejszych umiejętności przy nauce elektroniki jest umiejętność obliczania rezystora dla LEDa.

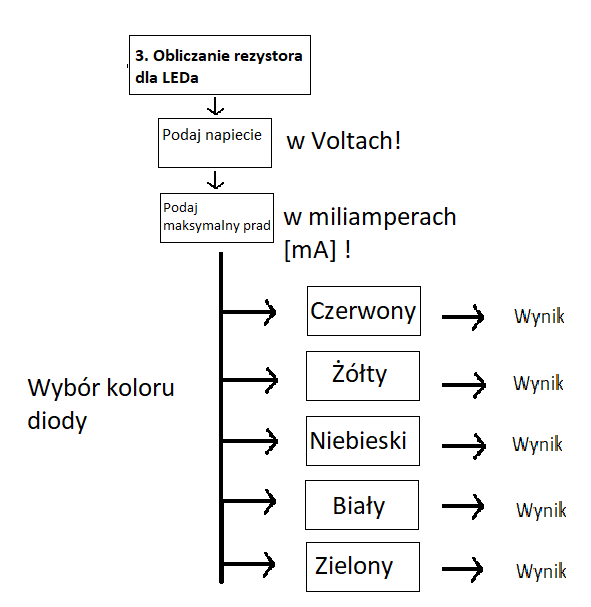
Obraz zawierający symbol, Czcionka, linia, biały

Opis wygenerowany automatycznie symbol LED

Nie zabezpieczona przez rezystor dioda, przez która popłynie nieznany prąd często ulega uszkodzeniu, dlatego kalkulator daje szeroka możliwość szybkiego obliczanie wartości rezystora w zależności od napięcia zasilania, maksymalnego prądu jaki chcemy przepuścić przez układ (LEDy nie powinny być traktowane większym prądem jak 30mA) oraz od koloru diody (kalkulator pozwala wybierać spośród 5 podstawowych kolorów dla LEDów).

Dla różnych kolorów diody maja różne spadki napięcia, kalkulator to uwzględnia.

Jest to bardzo przydatna funkcja dla osób rozpoczynających swoją przygodę z elektroniką.



Wynik zostanie nam zwrócony w omach i będzie to wartość rezystora jaka powinniśmy zastosować.

Należy pamiętać aby prąd podawać w mA ponieważ tylko wtedy otrzymamy poprawny wynik!

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

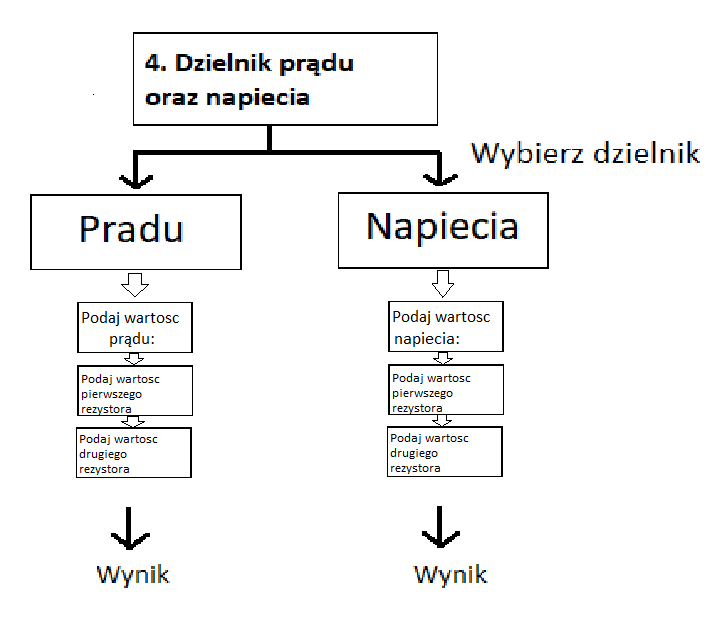
**3.4 Dzielnik prądu oraz napiecia**

Ostatnia opcja jest możliwość szybkiego otrzymania spadków napiec lub prądów na poszczególnych rezystorach za pomocą tak zwanych dzielników napięcia oraz prądu.

Dzięki gotowym funkcja możemy od razu otrzymać wyniki jedyne co trzeba podać to oczywiste dane takie jak:

**Dla dzielnika napięcia** – wartość napięcia zasilania oraz rezystancja obydwu rezystorów

**Dla dzielnika prądu** – wartość całkowita prądu jaki popłynie przez układ oraz również rezystancje obydwu rezystorów



Wynik zostanie zwrócony w przypadku dzielnika napięcia w woltach[V] a w przypadku dzielnika prądu w amperach[A]

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

1. **Analiza problemu**

Pierwszym zagadnieniem, problemem był sam kalkulator, co będzie się w nim znajdować i jakie będzie posiadał funkcje. Pewnym było iż na pewno będzie posiadał możliwość przeliczania pomiędzy systemami liczbowymi. Był to priorytet. Następnie do opisu funkcji zostały dodane opcje które zostały opisane w pierwszych akapitach. Ich wdrożenie nie było większym problemem, zostały opracowane w sposób prosty i czytelny.

Z racji iż kalkulator miał mieć możliwość przeliczania pomiędzy systemami stanowiło to największy problem.

Opracowałem założenie, które wydaje się dosyć logiczne oraz przemyślane. Zamiast projektować przeliczanie każdego systemu w każdy zostało wykonane:

Przeliczanie z systemu dziesiętnego na pozostałe czyli z DEC na HEX, BIN oraz OCT, a następnie zostały opracowane funkcje które obliczały kolejno z HEX, OCT , BIN na DEC a dopiero po obliczeniu na DEC na pozostałe.

Osiągnięte zostało dzięki temu uproszczenie, zamiast np. przeliczać liczbę z systemu HEX na BIN i pisać do tego osobna funkcje zostało to uzyskane przeliczając najpierw HEX na DEC a dopiero potem dzięki już istniejącej funkcji z DEC na BIN.

Dzięki takiemu rozwiązaniu zamiast dużej liczby w funkcji gdzie każda przelicza w każda kod wymagał napisanie jedynie czterech funkcji gdzie trzy pozostałe korzystały z tej jednej.

Interfejs użytkownika tez stanowił mały problem. Ostatecznie został zaimplementowany użytkownik w tak zwanej konsoli gdzie został w czytelny i jasny sposób opracowany.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**5. Projekt techniczny**

**5.1 Uwagi wstępne**

Cały projekt składa się z 3 plików i są to odpowiednio:

* plik main.cpp - (Main\_Kalkulator\_V1.cpp)
* plik .h - (Kalkulator\_MS.h)
* plik .cpp - (Kalkulator\_MS.cpp)

Plik main zawiera cały interfejs użytkownika wraz z zadeklarowanymi zmiennymi oraz stworzonymi obiektami

Plik: Kalkulator\_MS.h jest lista wszystkich klas oraz metod w nich użytych a plik: Kalkulator\_MS.cpp zawiera wszystkie metody jakie zostały użyte w projekcie.

Cały projekt został zrealizowany z użyciem dwóch klas:

* class Conversion
* class Calculations

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**5.2 Przeliczanie pomiędzy systemami liczbowymi / klasa Conversion**

Klasa Conversion zawiera w sobie 5 metod odpowiedzialnych tylko i wyłącznie za przeliczanie pomiędzy systemami liczbowymi a są to kolejno:

* string baseToSystem(int number, int base);
* string binToSystem(const string& bin, int system);
* string hexToSystem(const string& hex, int system);
* string octToSystem(const string& oct, int system);
* bool iteration();

baseToSystem jest odpowiedzialna za przeliczanie z systemu (DEC) na (HEX, OCT, BIN). Jej argumentami są „number” czyli liczba wprowadzona przez użytkownika do przeliczenia oraz „base” czyli system na który chcemy przeliczyć.

Jej składnia wygląda następująco:

string Conversion::baseToSystem(int number, int base)

{

string outcome;

while (number > 0)

{

int remainder = number % base;

char digit;

if (remainder < 10)

digit = '0' + remainder;

else

digit = 'A' + remainder - 10;

outcome.insert(outcome.begin(), digit);

number /= base;

}

return outcome;

}

*//należy mieć na uwadze ze w przyszłości kod może ulec lekkim zmianom*

pętla while będzie się wykonywać dopóki wartość number jest większa od 0, dzieli ona liczbę przez podstawę systemu base i zapisuje resztę w postaci cyfry już w odpowiednim systemie liczbowym.

W każdej pętli obliczana jest reszta z dzielenia i zapisywana do zmiennej remainder

Ważna sprawa jest instrukcja warunkowa if, gdzie jeśli nasza reszta z dzielenia jest mniejsza od 10 cyfra zamieniana jest na odpowiadający znak w systemie dziesiętnym i zapisywana do zmiennej char a jeśli jest większa od 10 to jest dodawana wartość ‘A’ dzięki temu dla systemu szesnastkowego otrzymamy kolejne odpowiadające im wartości A,B,C,D,E,F

Funkcja insert odwraca nam kolejne wartości przypisywane do zmiennej digit dzięki czemu możemy wyświetlić liczbę w odpowiedniej kolejności

Na końcu number jest dzielona przez base co przesuwa kolejna liczbę do analizy

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

Pozostałe funkcje różnią się składnią od baseToSystem ponieważ miedzy innymi ich liczba podawana jest w postaci znaków „string” a nie w postaci liczby „int” jak to jest dla funkcji która przelicza z DEC na pozostałe.

Opisywanie każdej funkcji nie ma sensu, warto zwrócić uwagę iż każda z nich korzysta z wyżej wymienionej funkcji i każda z nich zwraca ta właśnie funkcje.

Funkcja bool iteration(); została dodana do kodu aby umożliwić ponowne przeliczanie liczby jeśli użytkownik o to poprosi, zasada działania została przedstawiona na stronie 7,

Jej składnia wygląda następująco:

bool Conversion::iteration()

{

char choice;

cout << "Czy chcesz przeliczyc liczbe ponownie w innym systemie? (Y,y / N,n): ";

cout << "" << endl;

cin >> choice;

if (choice == 'Y' || choice == 'y')

{

return true;

}

else if (choice == 'N' || choice == 'n')

{

return false;

}

else

{

cout << "Niepoprawny wybor. Proszę sprobowac ponownie." << endl;

return iteration();

}

}

Jest to prosta funkcja która zwraca true lub false w zależności od tego co wybierze użytkownik dzięki czemu mamy możliwość ponownego przeliczenia jednostki.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**5.3 Podstawowe obliczenia dla elektronika**

**/ klasa Calculations**

Klasa Calculations zawiera w sobie:

1. double resistance(double V, double I);
2. double current(double V, double R);
3. double voltage(double R, double I);
4. double resistanceLED(double V, double I, string color);
5. vector <double> voltageDivider(double R1, double R2, double V);
6. vector <double> currentDivider(double R1, double R2, double I);
7. void readResistorSecurity(const string& nameR, double& R);

gdzie pierwsze trzy metody służą do prostych obliczeń prądu, napięcia i rezystancji:

double Calculations::current(double V, double R)

{

return V / R;

}

*//przykładowa metoda „current” do obliczania prądu*

a pozostałe realizują trochę bardziej rozbudowane zadania:

Metody voltageDivider oraz currentDivider jako jedyne zwracają wektor, ponieważ są to funkcje, które obliczają prądy oraz spadki napiec na 2 rezystorach i jest potrzeba zwracania 2 wartości

vector <double> Calculations::voltageDivider(double R1, double R2, double V)

{

vector <double> voltage;

V1 = (R1 / (R1 + R2)) \* V;

voltage.push\_back(V1);

V2 = (R2 / (R1 + R2)) \* V;

voltage.push\_back(V2);

return voltage;

}

*//przykładowa metoda „voltageDivider” która zwraca spadki napiec obliczone na poszczególnych //rezystorach podanych wcześniej przez użytkownika*

Inicjalizowane są w nich wektory a następnie obliczane poszczególne wartości które ma zwrócić funkcja. Następnie za pomocą metody push\_back są dodawane do wektora a następnie zwracane do funkcji.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

Metoda resistanceLED służy do obliczenia rezystora ograniczającego prąd dla diody.

Wyróżniająca ja cecha jest to iż zmienna drop przyjmuje wartości od tego, jaki kolor diody wybierze użytkownik. Dla poszczególnych kolorów spadek napięcia jest różny a funkcja uwzględnia ten najbardziej optymalny wyliczony ze średniej dla każdego koloru.

double Calculations::resistanceLED(double V, double I, string color)

{

double drop{}; //sredni spadek napiecie, uzelezniony od koloru LEDa

double IA; // prad w Amperach

IA = I / 1000;

if (V <= 0)

{

cout << "Niepoprawana wartosc napiecia zasilania" << endl;

return 0.0;

}

if (color == "red")

drop = 2.0;

else if (color == "blue")

drop = 3.8;

else if (color == "yellow")

drop = 2.2;

else if (color == "green")

drop = 3.0;

else if (color == "white")

drop = 3.3;

else

cout << "Niepoprawny kolor diody";

if (V <= drop)

{

cout << "Napiecie zasilania jest za małe, nie moze byc mniejsze niz spadek napiecia na diodzie" << endl;

return 0.0;

}

double diodeV = V - drop;

double resistanceLed = diodeV / IA;

return resistanceLed;

}

Warto zauważyć iż funkcja przelicza wartosc prądu podanego w mA, który podaje użytkownik na prad w Amperach

Funkcja jest w stanie obliczyć wartosc rezystora dla piecu kolorów diody lecz w bardzo prosty sposób można to rozbudować dla większej liczby jak i również dla diod takich jak np. IR

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**5.4 Zabezpieczenia**

Program został uzbrojony w kilka zabezpieczeń, które pozwalają w bezpieczny sposób poruszać się w menu kalkulatora oraz zabezpiecza go w przypadku podania złych wartości przy zapytaniach

W przypadku funkcji warunkowych if, else if, else użytkownik w prosty sposób może zaradzić jeśli zostanie wybrana opcja, która zostać nie powinna.

Jeśli program prosi użytkownika o podanie liczby, na której podstawie wybierze odpowiednia opcje to można po prostu w ostatniej części instrukcji warunkowej else zastosować :

int calcChoice{};

cin >> calcChoice;

if (calcChoice == 1) …

else if (calcChoice == 2) …

else if (calcChoice == 3) …

else

{

cout << "Podano nieprawidlowy wybor, sprobuj jeszcze raz";

cout << "" << endl;

}

gdzie każda wartosc która nie będzie pasowała do ustalonych wcześniej else if zostanie przekierowana do else a następnie zostanie wystawiony powyższy komunikat.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**5.4 Zabezpieczenia cd.**

Bardziej problematyczny okazał się problem przy wczytywaniu danych. Jeśli program prosił użytkownika o liczbe a została mu podana wartosc string np. litera, program się wysypywał.

W większości miejsc, gdzie potrzebne było poprawne wczytywanie danych zostało użyte:

*//na przykładzie metody readResistorSecurity z klasy Calculations*

void Calculations::readResistorSecurity(const string& nameR, double& R)

{

while (true)

{

cout << "Podaj wartosc rezystora " << nameR << " (Om) : ";

if (cin >> R)

{

if (R > 0)

break;

else cout << "Rezystancja nie moze byc mniejsza badz rowna 0" << endl;

}

else

{

cout << "Niepoprawna wartosc rezystancji, sprobuj ponownie" << endl;

cout << "" << endl;

cin.clear();

cin.ignore(std::numeric\_limits<std::streamsize>::max(), '\n');

}

}

}

Metoda ta sprawdza poprawność podawanych danych przy funkcji dla dzielników napiec gdzie program prosi o podanie wartości rezystorów. Jeśli zostanie podanego cos innego niż liczba wyskoczy błąd: "Niepoprawna wartosc rezystancji, sprobuj ponownie"

Pętla while będzie się wykonywała, aż nie zostanie spełniony warunek przerwania pętli.

W pierwszym if następuje sprawdzenie czy podana wartosc jest dokładna, jeśli tak to przechodzimy do następnego kroku, a jeśli nie to wykonywana jest instrukcja z else.

Wyświetlana jest informacja o błędnie wprowadzonej wartości a następnie czyszczony jest strumień n wejściowy cin za pomocą cin.clear() aby usunąć dane z bufora. Funkcja jest dodana cin.ignore dodatkowo aby zignorować pozostałości w buforze wejściowym.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

1. **Opis realizacji**

Do wykonania projektu używałem komputera HP Pavilion 15

* CPU: Ryzen 5600H
* GPU: RTX 3050TI
* RAM: 32GB 3200MHz
* SSD: 512Gb

Cały program został napisany z wykorzystaniem MicrosoftVIisualStudio 2022

na systemie operacyjnym Windows 10 Home

Program działa bez zarzutu, operacje są przeprowadzane płynnie.

Podczas testów zostały wykryte błędy programu, które nie wpływają na jego prace pod warunkiem prawidłowej obsługi programu

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

**7. Podręcznik użytkownika**

Aby program działał prawidłowo należy go używać w prawidłowy sposób i wykonując komendy podawane przez system.

Główne MENU:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Aby wybrać odpowiednia funkcje kalkulatora należy wybrać liczbe od „1 do 5”, gdzie wybranie „5” kończy działanie programu.

Po wybraniu pierwszej opcji pojawi się widok:

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Aby wybrać odpowiednia opcje należy wpisywać ‘HEX’ lub ‘BIN’ z dużych liter tak jak jest podane w poleceniu.

Obraz zawierający tekst, zrzut ekranu, Czcionka

Opis wygenerowany automatycznie

Przy wpisywaniu liczby w zapisie szesnastkowym można korzystać z małych liter jak i również przy opcji wyboru ponownego przeliczenia.

**Technical Report** **24.06.2023r**

**AGH University of Science and**

**Technology**

Podczas pracy pożytkowałem się różnymi źródłami wiedzy, głownie internetem oraz dziełami naukowymi a głownie książka Pana Bogusława Cyganka „Programowanie w języku C++ Wprowadzenie dla inżynierów”