

Vigenere

Wygenerowano przez Doxygen 1.9.3



<b>1 Indeks klas</b>	<b>1</b>
1.1 Lista klas	1
<b>2 Indeks plików</b>	<b>3</b>
2.1 Lista plików	3
<b>3 Dokumentacja klas</b>	<b>5</b>
3.1 Dokumentacja struktury Params	5
3.1.1 Opis szczegółowy	5
3.1.2 Dokumentacja atrybutów składowych	5
3.1.2.1 flaga_dzialania	5
3.1.2.2 inputFile	5
3.1.2.3 kluczFile	6
3.1.2.4 outputFile	6
<b>4 Dokumentacja plików</b>	<b>7</b>
4.1 Dokumentacja pliku C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/↔ Projekt/funkcje.cpp	7
4.1.1 Dokumentacja funkcji	8
4.1.1.1 czytaj_Parametry()	8
4.1.1.2 decyzyjator()	8
4.1.1.3 deszyfrowanie()	8
4.1.1.4 deszyfrowanie_znaku()	9
4.1.1.5 filtrowanie_tekstu()	9
4.1.1.6 komparator_klucza()	10
4.1.1.7 komparator_wartosci()	10
4.1.1.8 lamanie_kodu()	10
4.1.1.9 NWD()	11
4.1.1.10 NWDv2()	11
4.1.1.11 pomoc()	11
4.1.1.12 sortuj()	12
4.1.1.13 szyfrowanie()	12
4.1.1.14 szyfrowanie_znaku()	12
4.1.1.15 wczytaj_plik()	13
4.1.1.16 wyznacz_znak_klucza()	13
4.1.1.17 wyznaczanie_ciagow()	14
4.1.1.18 wyznaczanie_dlugosci_klucza()	14
4.1.1.19 wyznaczanie_klucza()	14
4.1.1.20 wyznaczanie_roznic()	15
4.2 Dokumentacja pliku C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/↔ Projekt/funkcje.h	15
4.2.1 Dokumentacja funkcji	16
4.2.1.1 czytaj_Parametry()	16
4.2.1.2 decyzyjator()	16

4.2.1.3 deszyfrowanie()	17
4.2.1.4 deszyfrowanie_znaku()	17
4.2.1.5 filtrowanie_tekstu()	17
4.2.1.6 komparator_klucza()	18
4.2.1.7 komparator_wartosci()	18
4.2.1.8 lamanie_kodu()	19
4.2.1.9 NWD()	19
4.2.1.10 NWDv2()	19
4.2.1.11 pomoc()	20
4.2.1.12 sortuj()	20
4.2.1.13 szyfrowanie()	20
4.2.1.14 szyfrowanie_znaku()	21
4.2.1.15 wczytaj_plik()	21
4.2.1.16 wyznacz_znak_klucza()	21
4.2.1.17 wyznaczanie_ciagow()	22
4.2.1.18 wyznaczanie_dlugosci_klucza()	22
4.2.1.19 wyznaczanie_klucza()	23
4.2.1.20 wyznaczanie_roznic()	23
4.3 funkcje.h	23
4.4 Dokumentacja pliku C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/↵ Projekt/main.cpp	24

# Rozdział 1

## Indeks klas

### 1.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

<a href="#">Params</a>	5
------------------------	---



## Rozdział 2

# Indeks plików

### 2.1 Lista plików

Tutaj znajduje się lista wszystkich udokumentowanych plików z ich krótkimi opisami:

C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/funkcje.cpp	. . .	7
C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/funkcje.h	. . .	15
C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/main.cpp	. . .	24





## Rozdział 3

# Dokumentacja klas

### 3.1 Dokumentacja struktury Params

```
#include <funkcje.h>
```

#### Atrybuty publiczne

- std::string [flaga\\_dzialania](#)
- std::string [inputFile](#)
- std::string [kluczFile](#)
- std::string [outputFile](#)

#### 3.1.1 Opis szczegółowy

Struktura zawierająca dane wejściowe

#### 3.1.2 Dokumentacja atrybutów składowych

##### 3.1.2.1 flaga\_dzialania

```
std::string Params::flaga_dzialania
```

flaga mówiąca o operacji, którą będziemy wykonywać (szyfrowanie, deszyfrowanie, łamanie kodu)

##### 3.1.2.2 inputFile

```
std::string Params::inputFile
```

nazwa pliku wejściowego

### 3.1.2.3 kluczFile

`std::string Params::kluczFile`

nazwa pliku zawierającego klucz

### 3.1.2.4 outputFile

`std::string Params::outputFile`

nazwa pliku wyjściowego

Dokumentacja dla tej struktury została wygenerowana z pliku:

- <C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/funkcje.h>

## Rozdział 4

# Dokumentacja plików

### 4.1 Dokumentacja pliku C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/funkcje.cpp

```
#include <iostream>
#include <sstream>
#include <string>
#include <fstream>
#include <vector>
#include "funkcje.h"
#include <map>
#include <unordered_map>
#include <cctype>
#include <algorithm>
```

#### Funkcje

- bool [czytaj\\_Parametry](#) (int argc, char \*\*argv, [Params](#) &params)
- void [pomoc](#) (const std::string &progName)
- void [szyfrowanie\\_znaku](#) (char &znak\_tekstu, const std::string &klucz, int &licznik\_klucza)
- void [deszyfrowanie\\_znaku](#) (std::string klucz, char &znak\_tekstu, int &licznik\_klucza)
- std::string [szyfrowanie](#) (const [Params](#) &params)
- std::string [wczytaj\\_plik](#) (std::string nazwa\_pliku)
- std::string [deszyfrowanie](#) ([Params](#) &params)
- std::vector< std::string > [wyznaczanie\\_ciagow](#) (std::string &tekst)
- int [NWD](#) (int a, int b)
- bool [komparator\\_klucza](#) (const std::pair< int, int > &a, const std::pair< int, int > &b)
- bool [komparator\\_wartosci](#) (const std::pair< int, int > &a, const std::pair< int, int > &b)
- std::vector< std::pair< int, int > > [sortuj](#) (std::map< int, int > &mapa)
- int [NWDv2](#) (std::vector< int > &liczby)
- std::vector< int > [wyznaczanie\\_roznic](#) (std::unordered\_map< std::string, std::vector< int > > &slowa)
- int [wyznaczanie\\_dlugosci\\_klucza](#) (std::vector< std::string > &ciagi)
- std::string [filtrowanie\\_tekstu](#) (std::string tekst)
- char [wyznacz\\_znak\\_klucza](#) (std::string &tekst, int dlugosc\_klucza, int poz\_klucza)
- std::string [wyznaczanie\\_klucza](#) (std::string &tekst, int dlugosc\_klucza)
- std::string [lamanie\\_kodu](#) ([Params](#) &params)
- bool [decyzjator](#) ([Params](#) &params)

### 4.1.1 Dokumentacja funkcji

#### 4.1.1.1 czytaj\_Parametry()

```
bool czytaj_Parametry (
    int argc,
    char ** argv,
    Params & params )
```

Funkcja zaczytująca parametry do programu i zapisująca je do struktury

##### Parametry

<i>argc</i>	wartości parametrów z którymi został wywołany program
<i>argv</i>	tablica tych wartości
<i>params</i>	struktura do której zapisywane są parametry

##### Zwraca

funkcja zwraca true, jeżeli program został wywołany z odpowiednimi parametrami. W przeciwnym razie zwraca false

#### 4.1.1.2 decyzyjator()

```
bool decyzyjator (
    Params & params )
```

Główna funkcja sterująca. Decyduje, którą z operacji będziemy wykonywać na podstawie parametrów wejściowych

##### Parametry

<i>params</i>	parametry wejściowe
---------------	---------------------

##### Zwraca

funkcja zwraca true jeśli była odpowiednia flaga działania i udało się wejść do odpowiedniego segmentu funkcji związanego z tą flagą. W przeciwnym razie zwraca false

#### 4.1.1.3 deszyfrowanie()

```
std::string deszyfrowanie (
    Params & params )
```

Główna funkcja deszyfrująca

## Parametry

<i>params</i>	parametry wejsciowe
---------------	---------------------

## Zwraca

funkcja zwraca odszyfrowany tekst

**4.1.1.4 deszyfrowanie\_znaku()**

```
void deszyfrowanie_znaku (
    std::string klucz,
    char & znak_tekstu,
    int & licznik_klucza )
```

Funkcja deszyfruje znak tekstu kolejnym znakiem klucza i jeśli znak dało się zdeszyfrować (tzn. a-z, A-Z), to przypisuje go wartości oryginału

## Parametry

<i>klucz</i>	klucz którym deszyfrujemy
<i>znak_tekstu</i>	kolejny znak tekstu do zdeszyfrowania
<i>licznik_klucza</i>	kolejny znak klucza, a gdy dojdziemy do jego końca, bierze się go od początku

**4.1.1.5 filtrowanie\_tekstu()**

```
std::string filtrowanie_tekstu (
    std::string tekst )
```

Funkcja wyznacza tekst składający się z samych znaków A-Z. Duże litery zostawia, małe litery zamienia na duże, a resztę znaków pomija

## Parametry

<i>tekst</i>	tekst do przefiltrowania
--------------	--------------------------

## Zwraca

funkcja zwraca przefiltrowany tekst

#### 4.1.1.6 komparator\_klucza()

```
bool komparator_klucza (
    const std::pair< int, int > & a,
    const std::pair< int, int > & b )
```

Funkcja porównująca pary na podstawie klucza

##### Parametry

<i>a</i>	
<i>b</i>	

##### Zwraca

funkcja zwraca true jeśli `a.first > b.first`, w przeciwnym razie false

#### 4.1.1.7 komparator\_wartosci()

```
bool komparator_wartosci (
    const std::pair< int, int > & a,
    const std::pair< int, int > & b )
```

Funkcja porównująca pary na podstawie wartości

##### Parametry

<i>a</i>	
<i>b</i>	

##### Zwraca

funkcja zwraca true jeśli `a.second > b.second`, w przeciwnym razie false

#### 4.1.1.8 łamanie\_kodu()

```
std::string łamanie_kodu (
    Params & params )
```

Główna funkcja do łamania szyfru

##### Parametry

<i>params</i>	parametry wejściowe
---------------	---------------------

**Zwraca**

funkcja zwraca odszyfrowany tekst

**4.1.1.9 NWD()**

```
int NWD (
    int a,
    int b )
```

Funkcja wyznacza nwd rekurencyjnie dwóch liczb metodą euklidesa

**Parametry**

<i>a</i>	pierwsza liczba
<i>b</i>	druga liczba

**Zwraca**

funkcja zwraca nwd tych dwóch liczb

**4.1.1.10 NWDv2()**

```
int NWDv2 (
    std::vector< int > & liczby )
```

Funkcja wyznacza statystycznie największą liczbę, dla której ilość powstałych nwd jest największa

**Parametry**

<i>liczby</i>	wyniki operacji nwd dla wszystkich ciągów
---------------	---

**Zwraca**

funkcja zwraca jedno statystycznie najczęstsze i największe nwd

**4.1.1.11 pomoc()**

```
void pomoc (
    const std::string & progName )
```

Funkcja wyświetla okno pomocy, gdy `czytaj_Parametry` zwróci false

## Parametry

<i>progName</i>	nazwa programu
-----------------	----------------

**4.1.1.12 sortuj()**

```
std::vector< std::pair< int, int > > sortuj (
    std::map< int, int > & mapa )
```

Funkcja zamienia mapę na wektor i sortuje go

## Parametry

<i>mapa</i>	mapa do posortowania
-------------	----------------------

## Zwraca

funkcja zwraca posortowanego wektora, utworzonego z mapy

**4.1.1.13 szyfrowanie()**

```
std::string szyfrowanie (
    const Params & params )
```

Główna funkcja szyfrująca

## Parametry

<i>params</i>	parametry wejściowe
---------------	---------------------

## Zwraca

funkcja zwraca zaszyfrowany tekst

**4.1.1.14 szyfrowanie\_znaku()**

```
void szyfrowanie_znaku (
    char & znak_tekstu,
    const std::string & klucz,
    int & licznik_klucza )
```

Funkcja szyfruje znak tekstu kolejnym znakiem klucza i jeśli znak dało się zaszyfrować (tzn. a-z, A-Z), to przypisuje go wartości oryginału



## Parametry

<i>znak_tekstu</i>	kolejny znak tekstu do zaszyfrowania
<i>klucz</i>	klucz którym szyfrujemy
<i>licznik_klucza</i>	kolejny znak klucza, a gdy dojdziemy do jego końca, bierze się go od początku

## 4.1.1.15 wczytaj\_plik()

```
std::string wczytaj_plik (
    std::string nazwa_pliku )
```

Funkcja wczytuje plik o nazwie z parametrów wejściowych

## Parametry

<i>params</i>	parametry wejściowe
---------------	---------------------

## Zwraca

funkcja zwraca tekst z tego pliku

## 4.1.1.16 wyznacz\_znak\_klucza()

```
char wyznacz_znak_klucza (
    std::string & tekst,
    int dlugosc_klucza,
    int poz_klucza )
```

Funkcja wyznacza literę klucza na podstawie co n-tej litery (n-długość klucza) i wyznacza najczęściej powtarzającą się. W języku polskim jest to "a", więc na tej podstawie obliczamy przesunięcie

## Parametry

<i>tekst</i>	tekst główny z którego czytujemy litery
<i>dlugosc_klucza</i>	długość klucza
<i>poz_klucza</i>	zmienna opisująca którego znaku klucza szukamy, co którą literę wczytujemy

## Zwraca

funkcja zwraca literę klucza

#### 4.1.1.17 wyznaczanie\_ciagow()

```
std::vector< std::string > wyznaczanie_ciagow (
    std::string & tekst )
```

Funkcja wyznacza trzyliterowe ciągi

##### Parametry

<i>tekst</i>	tekst, który będziemy analizować
--------------	----------------------------------

##### Zwraca

funkcja zwraca te ciągi

#### 4.1.1.18 wyznaczanie\_dlugosci\_klucza()

```
int wyznaczanie_dlugosci_klucza (
    std::vector< std::string > & ciagi )
```

Funkcja wyznacza długość klucza na podstawie największej ilości powtórzeń nwd dla danego ułożenia różnic odległości między powtarzającymi się ciągami, oraz zmienia tą kolejność dla dokładniejszego wyniku

##### Parametry

<i>ciagi</i>	3 literowe ciągi ułożone z kolejnych liter zaszyfrowanego tekstu
--------------	--

##### Zwraca

funkcja zwraca długość klucza

#### 4.1.1.19 wyznaczanie\_klucza()

```
std::string wyznaczanie_klucza (
    std::string & tekst,
    int dlugosc_klucza )
```

Funkcja wyznacza klucz na podstawie jego długości. Sprawdza co n-tą literę (n-długość klucza) i wyznacza najczęściej powtarzającą się. W języku polskim jest to "a", więc na tej podstawie obliczamy przesunięcie

##### Parametry

<i>tekst</i>	tekst główny z którego zczytujemy litery
<i>dlugosc_klucza</i>	długość klucza

**Zwraca**

funkcja zwraca klucz

**4.1.1.20 wyznaczanie\_roznic()**

```
std::vector< int > wyznaczanie_roznic (
    std::unordered_map< std::string, std::vector< int > > & slowa )
```

Funkcja wyznacza różnice między pierwszym i każdym kolejnym wystąpieniem poszczególnego 3-literowego ciągu

**Parametry**

<i>slowa</i>	mapa tych 3-literowych ciągów (<ciąg>, <pozycje wystąpienia>)
--------------	---

**Zwraca**

funkcja zwraca różnice pomiędzy pierwszym i każdym kolejnym wystąpieniem poszczególnego 3-literowego ciągu

## 4.2 Dokumentacja pliku C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/funkcje.h

```
#include <iostream>
#include <string>
#include <vector>
#include <cctype>
#include <unordered_map>
#include <map>
```

**Komponenty**

- struct [Params](#)

**Funkcje**

- bool [czytaj\\_Parametry](#) (int argc, char \*\*argv, [Params](#) &params)
- void [pomoc](#) (const std::string &progName)
- void [szyfrowanie\\_znaku](#) (char &znak\_tekstu, const std::string &klucz, int &licznik\_klucza)
- void [deszyfrowanie\\_znaku](#) (std::string klucz, char &znak\_tekstu, int &licznik\_klucza)
- std::string [szyfrowanie](#) (const [Params](#) &params)
- std::string [wczytaj\\_plik](#) (std::string nazwa\_pliku)
- std::string [deszyfrowanie](#) ([Params](#) &params)
- std::vector< std::string > [wyznaczanie\\_ciagow](#) (std::string &tekst)
- int [NWD](#) (int a, int b)

- int `NWDv2` (`std::vector< int > &liczby`)
- int `wyznaczanie_dlugosci_klucza` (`std::vector< std::string > &ciagi`)
- `std::string` `filtrowanie_tekstu` (`std::string tekst`)
- `std::string` `lamanie_kodu` (`Params &params`)
- bool `decyzjator` (`Params &params`)
- `std::vector< int >` `wyznaczanie_roznic` (`std::unordered_map< std::string, std::vector< int > > &slowa`)
- `std::string` `wyznaczanie_klucza` (`std::string &tekst, int dlugosc_klucza`)
- char `wyznacz_znak_klucza` (`std::string &tekst, int dlugosc_klucza, int poz_klucza`)
- bool `komparator_wartosci` (`const std::pair< int, int > &a, const std::pair< int, int > &b`)
- bool `komparator_klucza` (`const std::pair< int, int > &a, const std::pair< int, int > &b`)
- `std::vector< std::pair< int, int > >` `sortuj` (`std::map< int, int > &mapa`)

## 4.2.1 Dokumentacja funkcji

### 4.2.1.1 czytaj\_Parametry()

```
bool czytaj_Parametry (
    int argc,
    char ** argv,
    Params & params )
```

Funkcja zaczytująca parametry do programu i zapisująca je do struktury

#### Parametry

<i>argc</i>	wartości parametrów z którymi został wywołany program
<i>argv</i>	tablica tych wartości
<i>params</i>	struktura do której zapisywane są parametry

#### Zwraca

funkcja zwraca true, jeżeli program został wywołany z odpowiednimi parametrami. W przeciwnym razie zwraca false

### 4.2.1.2 decyzjator()

```
bool decyzjator (
    Params & params )
```

Główna funkcja sterująca. Decyduje, którą z operacji będziemy wykonywać na podstawie parametrów wejściowych

#### Parametry

<i>params</i>	parametry wejściowe
---------------	---------------------

**Zwraca**

funkcja zwraca true jeśli była odpowiednia flaga działania i udało się wejść do odpowiedniego segmentu funkcji związanego z tą flagą. W przeciwnym razie zwraca false

**4.2.1.3 deszyfrowanie()**

```
std::string deszyfrowanie (
    Params & params )
```

Główna funkcja deszyfrująca

**Parametry**

<i>params</i>	parametry wejściowe
---------------	---------------------

**Zwraca**

funkcja zwraca odszyfrowany tekst

**4.2.1.4 deszyfrowanie\_znaku()**

```
void deszyfrowanie_znaku (
    std::string klucz,
    char & znak_tekstu,
    int & licznik_klucza )
```

Funkcja deszyfruje znak tekstu kolejnym znakiem klucza i jeśli znak dało się zdeszyfrować (tzn. a-z, A-Z), to przypisuje go wartości oryginału

**Parametry**

<i>klucz</i>	klucz którym deszyfrujemy
<i>znak_tekstu</i>	kolejny znak tekstu do zdeszyfrowania
<i>licznik_klucza</i>	kolejny znak klucza, a gdy dojdziemy do jego końca, bierze się go od początku

**4.2.1.5 filtrowanie\_tekstu()**

```
std::string filtrowanie_tekstu (
    std::string tekst )
```

Funkcja wyznacza tekst składający się z samych znaków A-Z. Duże litery zostawia, małe litery zamienia na duże, a resztę znaków pomija

**Parametry**

<i>tekst</i>	tekst do przefiltrowania
--------------	--------------------------

**Zwraca**

funkcja zwraca przefiltrowany tekst

**4.2.1.6 komparator\_klucza()**

```
bool komparator_klucza (
    const std::pair< int, int > & a,
    const std::pair< int, int > & b )
```

Funkcja porównująca pary na podstawie klucza

**Parametry**

<i>a</i>	
<i>b</i>	

**Zwraca**

funkcja zwraca true jeśli `a.first > b.first`, w przeciwnym razie false

**4.2.1.7 komparator\_wartosci()**

```
bool komparator_wartosci (
    const std::pair< int, int > & a,
    const std::pair< int, int > & b )
```

Funkcja porównująca pary na podstawie wartości

**Parametry**

<i>a</i>	
<i>b</i>	

**Zwraca**

funkcja zwraca true jeśli `a.second > b.second`, w przeciwnym razie false

#### 4.2.1.8 łamanie\_kodu()

```
std::string łamanie_kodu (
    Params & params )
```

Główna funkcja do łamania szyfru

##### Parametry

<i>params</i>	parametry wejściowe
---------------	---------------------

##### Zwraca

funkcja zwraca odszyfrowany tekst

#### 4.2.1.9 NWD()

```
int NWD (
    int a,
    int b )
```

Funkcja wyznacza nwd rekurencyjnie dwóch liczb metodą euklidesa

##### Parametry

<i>a</i>	pierwsza liczba
<i>b</i>	druga liczba

##### Zwraca

funkcja zwraca nwd tych dwóch liczb

#### 4.2.1.10 NWDv2()

```
int NWDv2 (
    std::vector< int > & liczby )
```

Funkcja wyznacza statystycznie największą liczbę, dla której ilość powstałych nwd jest największa

##### Parametry

<i>liczby</i>	wyniki operacji nwd dla wszystkich ciągów
---------------	---

**Zwraca**

funkcja zwraca jedno statystycznie najczęstsze i największe nwd

**4.2.1.11 pomoc()**

```
void pomoc (
    const std::string & progName )
```

Funkcja wyświetla okno pomocy, gdy czytaj\_Parametry zwróci false

**Parametry**

<i>progName</i>	nazwa programu
-----------------	----------------

**4.2.1.12 sortuj()**

```
std::vector< std::pair< int, int > > sortuj (
    std::map< int, int > & mapa )
```

Funkcja zamienia mapę na wektor i sortuje go

**Parametry**

<i>mapa</i>	mapa do posortowania
-------------	----------------------

**Zwraca**

funkcja zwraca posortowanego wektora, utworzonego z mapy

**4.2.1.13 szyfrowanie()**

```
std::string szyfrowanie (
    const Params & params )
```

Główna funkcja szyfrująca

**Parametry**

<i>params</i>	parametry wejściowe
---------------	---------------------



**Zwraca**

funkcja zwraca zaszyfrowany tekst

**4.2.1.14 szyfrowanie\_znaku()**

```
void szyfrowanie_znaku (
    char & znak_tekstu,
    const std::string & klucz,
    int & licznik_klucza )
```

Funkcja szyfruje znak tekstu kolejnym znakiem klucza i jeśli znak dało się zaszyfrować (tzn. a-z, A-Z), to przypisuje go wartości oryginału

**Parametry**

<i>znak_tekstu</i>	kolejny znak tekstu do zaszyfrowania
<i>klucz</i>	klucz którym szyfrujemy
<i>licznik_klucza</i>	kolejny znak klucza, a gdy dojdziemy do jego końca, bierze się go od początku

**4.2.1.15 wczytaj\_plik()**

```
std::string wczytaj_plik (
    std::string nazwa_pliku )
```

Funkcja wczytuje plik o nazwie z parametrów wejściowych

**Parametry**

<i>params</i>	parametry wejściowe
---------------	---------------------

**Zwraca**

funkcja zwraca tekst z tego pliku

**4.2.1.16 wyznacz\_znak\_klucza()**

```
char wyznacz_znak_klucza (
    std::string & tekst,
    int dlugosc_klucza,
    int poz_klucza )
```

Funkcja wyznacza literę klucza na podstawie co n-tej litery (n-długość klucza) i wyznacza najczęściej powtarzającą się. W języku polskim jest to "a", więc na tej podstawie obliczamy przesunięcie

**Parametry**

<i>tekst</i>	tekst główny z którego zczytujemy litery
<i>dlugosc_klucza</i>	długość klucza
<i>poz_klucza</i>	zmienna opisująca którego znaku klucza szukamy, co którą literę wczytujemy

**Zwraca**

funkcja zwraca literę klucza

**4.2.1.17 wyznaczanie\_ciagow()**

```
std::vector< std::string > wyznaczanie_ciagow (
    std::string & tekst )
```

Funkcja wyznacza trzyliterowe ciągi

**Parametry**

<i>tekst</i>	tekst, który będziemy analizować
--------------	----------------------------------

**Zwraca**

funkcja zwraca te ciągi

**4.2.1.18 wyznaczanie\_dlugosci\_klucza()**

```
int wyznaczanie_dlugosci_klucza (
    std::vector< std::string > & ciagi )
```

Funkcja wyznacza długość klucza na podstawie największej ilości powtórzeń nwd dla danego ułożenia różnic odległości między powtarzającymi się ciągami, oraz zmienia tą kolejność dla dokładniejszego wyniku

**Parametry**

<i>ciagi</i>	3 literowe ciągi ułożone z kolejnych liter zaszyfrowanego tekstu
--------------	--

**Zwraca**

funkcja zwraca długość klucza

#### 4.2.1.19 wyznaczanie\_klucza()

```
std::string wyznaczanie_klucza (
    std::string & tekst,
    int dlugosc_klucza )
```

Funkcja wyznacza klucz na podstawie jego długości. Sprawdza co n-tą literę (n-długość klucza) i wyznacza najczęściej powtarzającą się. W języku polskim jest to "a", więc na tej podstawie obliczamy przesunięcie

##### Parametry

<i>tekst</i>	tekst główny z którego czytujemy litery
<i>dlugosc_klucza</i>	długość klucza

##### Zwraca

funkcja zwraca klucz

#### 4.2.1.20 wyznaczanie\_roznic()

```
std::vector< int > wyznaczanie_roznic (
    std::unordered_map< std::string, std::vector< int > > & slowa )
```

Funkcja wyznacza różnice między pierwszym i każdym kolejnym wystąpieniem poszczególnego 3-literowego ciągu

##### Parametry

<i>slowa</i>	mapa tych 3-literowych ciągów (<ciąg>, <pozycje wystapienia>)
--------------	---

##### Zwraca

funkcja zwraca różnice pomiędzy pierwszym i każdym kolejnym wystąpieniem poszczególnego 3-literowego ciągu

## 4.3 funkcje.h

[Idź do dokumentacji tego pliku.](#)

```
1
2 #pragma once
3
4 #include<iostream>
5 #include<string>
6 #include<vector>
7 #include <cctype>
8 #include <unordered_map>
9 #include <map>
10
12 struct Params
13 {
14     std::string flaga_dzialania;
15     std::string inputFile;
16     std::string kluczFile;
```

```
17     std::string outputFile;
18 };
19
26 bool czytaj_Parametry(int argc, char** argv, Params& params);
27
31 void pomoc(const std::string& progName);
32
38 void szyfrowanie_znaku(char &znak_tekstu, const std::string &klucz, int& licznik_klucza);
39
45 void deszyfrowanie_znaku(std::string klucz, char& znak_tekstu, int& licznik_klucza);
46
51 std::string szyfrowanie(const Params& params);
52
57 std::string wczytaj_plik(std::string nazwa_pliku);
58
63 std::string deszyfrowanie(Params &params);
64
69 std::vector<std::string> wyznaczanie_ciagow(std::string &tekst);
70
76 int NWD(int a, int b);
77
82 int NWDv2(std::vector<int>& liczby);
83
88 int wyznaczanie_dlugosci_klucza(std::vector<std::string>&ciagi);
89
94 std::string filtrowanie_tekstu(std::string tekst);
95
100 std::string lamanie_kodu(Params &params);
101
106 bool decyzyjator( Params& params);
107
112 std::vector<int>wyznaczanie_roznic(std::unordered_map<std::string, std::vector<int> &slowa);
113
119 std::string wyznaczanie_klucza(std::string &tekst, int dlugosc_klucza);
120
127 char wyznacz_znak_klucza(std::string &tekst, int dlugosc_klucza, int poz_klucza);
128
134 bool komparator_wartosci(const std::pair<int, int>& a, const std::pair<int, int>& b);
135
141 bool komparator_klucza(const std::pair<int, int>& a, const std::pair<int, int>& b);
142
147 std::vector<std::pair<int, int>>sortuj(std::map<int, int> &mapa);
```

## 4.4 Dokumentacja pliku C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/main.cpp

```
#include <iostream>
#include <string>
#include <fstream>
#include "funkcje.h"
```

### Funkcje

- int main (int argc, char \*\*argv)

# Indeks

C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/funkcje.cpp, [7](#)  
C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/funkcje.h, [15](#), [23](#)  
C:/Users/user/source/repos/polsl-aei-ppk-katowice/a682f497-gr22-repo/projekt/Projekt/main.cpp, [24](#)

czytaj\_Parametry  
    funkcje.cpp, [8](#)  
    funkcje.h, [16](#)

decyzjator  
    funkcje.cpp, [8](#)  
    funkcje.h, [16](#)

deszyfrowanie  
    funkcje.cpp, [8](#)  
    funkcje.h, [17](#)

deszyfrowanie\_znaku  
    funkcje.cpp, [9](#)  
    funkcje.h, [17](#)

filtrowanie\_tekstu  
    funkcje.cpp, [9](#)  
    funkcje.h, [17](#)

flaga\_dzialania  
    Params, [5](#)

funkcje.cpp  
    czytaj\_Parametry, [8](#)  
    decyzjator, [8](#)  
    deszyfrowanie, [8](#)  
    deszyfrowanie\_znaku, [9](#)  
    filtrowanie\_tekstu, [9](#)  
    komparator\_klucza, [9](#)  
    komparator\_wartosci, [10](#)  
    lamanie\_kodu, [10](#)  
    NWD, [11](#)  
    NWDv2, [11](#)  
    pomoc, [11](#)  
    sortuj, [12](#)  
    szyfrowanie, [12](#)  
    szyfrowanie\_znaku, [12](#)  
    wczytaj\_plik, [13](#)  
    wyznacz\_znak\_klucza, [13](#)  
    wyznaczanie\_ciagow, [13](#)  
    wyznaczanie\_dlugosci\_klucza, [14](#)  
    wyznaczanie\_klucza, [14](#)  
    wyznaczanie\_roznic, [15](#)

funkcje.h  
    czytaj\_Parametry, [16](#)  
    decyzjator, [16](#)  
    deszyfrowanie, [17](#)  
    deszyfrowanie\_znaku, [17](#)  
    filtrowanie\_tekstu, [17](#)  
    komparator\_klucza, [18](#)  
    komparator\_wartosci, [18](#)  
    lamanie\_kodu, [18](#)  
    NWD, [19](#)  
    NWDv2, [19](#)  
    pomoc, [20](#)  
    sortuj, [20](#)  
    szyfrowanie, [20](#)  
    szyfrowanie\_znaku, [21](#)  
    wczytaj\_plik, [21](#)  
    wyznacz\_znak\_klucza, [21](#)  
    wyznaczanie\_ciagow, [22](#)  
    wyznaczanie\_dlugosci\_klucza, [22](#)  
    wyznaczanie\_klucza, [22](#)  
    wyznaczanie\_roznic, [23](#)

inputFile  
    Params, [5](#)

kluczFile  
    Params, [5](#)

komparator\_klucza  
    funkcje.cpp, [9](#)  
    funkcje.h, [18](#)

komparator\_wartosci  
    funkcje.cpp, [10](#)  
    funkcje.h, [18](#)

lamanie\_kodu  
    funkcje.cpp, [10](#)  
    funkcje.h, [18](#)

NWD  
    funkcje.cpp, [11](#)  
    funkcje.h, [19](#)

NWDv2  
    funkcje.cpp, [11](#)  
    funkcje.h, [19](#)

outputFile  
    Params, [6](#)

Params, [5](#)  
    flaga\_dzialania, [5](#)  
    inputFile, [5](#)  
    kluczFile, [5](#)  
    outputFile, [6](#)

pomoc  
    funkcje.cpp, [11](#)

- funkcje.h, [20](#)
- sortuj
  - funkcje.cpp, [12](#)
  - funkcje.h, [20](#)
- szyfrowanie
  - funkcje.cpp, [12](#)
  - funkcje.h, [20](#)
- szyfrowanie\_znakulka
  - funkcje.cpp, [12](#)
  - funkcje.h, [21](#)
- wczytaj\_plik
  - funkcje.cpp, [13](#)
  - funkcje.h, [21](#)
- wyznacz\_znak\_klucza
  - funkcje.cpp, [13](#)
  - funkcje.h, [21](#)
- wyznaczanie\_ciagow
  - funkcje.cpp, [13](#)
  - funkcje.h, [22](#)
- wyznaczanie\_dlugosci\_klucza
  - funkcje.cpp, [14](#)
  - funkcje.h, [22](#)
- wyznaczanie\_klucza
  - funkcje.cpp, [14](#)
  - funkcje.h, [22](#)
- wyznaczanie\_roznic
  - funkcje.cpp, [15](#)
  - funkcje.h, [23](#)