

KalkulatorONP

Wygenerowano przez Doxygen 1.8.17

1 Indeks struktur danych	1
1.1 Struktury danych	1
2 Indeks plików	3
2.1 Lista plików	3
3 Dokumentacja struktur danych	5
3.1 Dokumentacja struktury Onp	5
3.1.1 Opis szczegółowy	5
3.1.2 Dokumentacja pól	5
3.1.2.1 pNext	5
3.1.2.2 pPrev	6
3.1.2.3 type	6
3.1.2.4 value	6
3.2 Dokumentacja struktury Pair	6
3.2.1 Opis szczegółowy	6
3.2.2 Dokumentacja pól	6
3.2.2.1 pOnp	7
3.2.2.2 pStack	7
3.3 Dokumentacja struktury Stack	7
3.3.1 Opis szczegółowy	7
3.3.2 Dokumentacja pól	7
3.3.2.1 pPrev	7
3.3.2.2 priority	8
3.3.2.3 type	8
4 Dokumentacja plików	9
4.1 Dokumentacja pliku Calculator.c	9
4.1.1 Dokumentacja definicji	9
4.1.1.1 _CRT_SECURE_NO_DEPRECATED	9
4.1.2 Dokumentacja funkcji	10
4.1.2.1 calculator()	10
4.1.2.2 FreeMemory()	10
4.1.2.3 getMultiResult()	10
4.1.2.4 getSingleResult()	11
4.2 Dokumentacja pliku Calculator.h	11
4.2.1 Dokumentacja funkcji	11
4.2.1.1 calculator()	11
4.2.1.2 FreeMemory()	12
4.2.1.3 getMultiResult()	12
4.2.1.4 getSingleResult()	13
4.3 Dokumentacja pliku Constans.h	13
4.3.1 Dokumentacja definicji	13

4.3.1.1	END_ARRAY	13
4.3.1.2	INPUT_SIZE	13
4.3.1.3	MULTI_ARGUMENTS	14
4.3.1.4	STACK_SIZE	14
4.3.1.5	UNDEFINED	14
4.4	Dokumentacja pliku ConvertFunctions.c	14
4.4.1	Dokumentacja definicji	14
4.4.1.1	_CRT_SECURE_NO_DEPRECATED	14
4.4.2	Dokumentacja funkcji	15
4.4.2.1	charToDigit()	15
4.4.2.2	getFraction()	15
4.4.2.3	getOperation()	15
4.4.2.4	getPriority()	16
4.4.2.5	getType()	16
4.5	Dokumentacja pliku ConvertFunctions.h	16
4.5.1	Dokumentacja funkcji	17
4.5.1.1	charToDigit()	17
4.5.1.2	getFraction()	17
4.5.1.3	getOperation()	18
4.5.1.4	getPriority()	18
4.5.1.5	getType()	18
4.6	Dokumentacja pliku Debug/KalkulatorONP.vcxproj.FileListAbsolute.txt	19
4.7	Dokumentacja pliku FileFunctions.c	19
4.7.1	Dokumentacja definicji	19
4.7.1.1	_CRT_SECURE_NO_DEPRECATED	19
4.7.2	Dokumentacja funkcji	19
4.7.2.1	addOnpToFile()	19
4.7.2.2	addResultToFile()	20
4.7.2.3	readFile()	20
4.8	Dokumentacja pliku FileFunctions.h	20
4.8.1	Dokumentacja funkcji	21
4.8.1.1	addOnpToFile()	21
4.8.1.2	addResultToFile()	21
4.8.1.3	readFile()	21
4.9	Dokumentacja pliku history.txt	22
4.10	Dokumentacja pliku Main.c	22
4.10.1	Dokumentacja definicji	22
4.10.1.1	_CRT_SECURE_NO_DEPRECATED	22
4.10.2	Dokumentacja funkcji	22
4.10.2.1	main()	23
4.11	Dokumentacja pliku OnpFunctions.c	23
4.11.1	Dokumentacja definicji	23

4.11.1.1 <code>_CRT_SECURE_NO_DEPRECATED</code>	23
4.11.2 Dokumentacja funkcji	23
4.11.2.1 <code>addToOnp()</code>	24
4.11.2.2 <code>addToPair()</code>	25
4.11.2.3 <code>findOnpHead()</code>	25
4.11.2.4 <code>freeOnp()</code>	26
4.11.2.5 <code>onpBuilder()</code>	26
4.11.2.6 <code>setState()</code>	26
4.11.2.7 <code>stateControl()</code>	27
4.12 Dokumentacja pliku <code>OnpFunctions.h</code>	27
4.12.1 Dokumentacja funkcji	27
4.12.1.1 <code>addToOnp()</code>	27
4.12.1.2 <code>addToPair()</code>	28
4.12.1.3 <code>findOnpHead()</code>	28
4.12.1.4 <code>freeOnp()</code>	29
4.12.1.5 <code>onpBuilder()</code>	29
4.12.1.6 <code>setState()</code>	29
4.12.1.7 <code>stateControl()</code>	30
4.13 Dokumentacja pliku <code>StackFunctions.c</code>	30
4.13.1 Dokumentacja definicji	31
4.13.1.1 <code>_CRT_SECURE_NO_DEPRECATED</code>	31
4.13.2 Dokumentacja funkcji	31
4.13.2.1 <code>addToStack()</code>	31
4.13.2.2 <code>emptyStack()</code>	31
4.13.2.3 <code>modyfyStack()</code>	32
4.13.2.4 <code>removeFromStack()</code>	32
4.13.2.5 <code>stackAction()</code>	32
4.14 Dokumentacja pliku <code>StackFunctions.h</code>	33
4.14.1 Dokumentacja funkcji	33
4.14.1.1 <code>addToStack()</code>	33
4.14.1.2 <code>emptyStack()</code>	34
4.14.1.3 <code>modyfyStack()</code>	34
4.14.1.4 <code>removeFromStack()</code>	35
4.14.1.5 <code>stackAction()</code>	35
4.15 Dokumentacja pliku <code>Structures.h</code>	35
4.15.1 Dokumentacja typów wyliczanych	36
4.15.1.1 <code>Type</code>	36
4.16 Dokumentacja pliku <code>Test.txt</code>	36
4.17 Dokumentacja pliku <code>UiFunctions.c</code>	36
4.17.1 Dokumentacja definicji	37
4.17.1.1 <code>_CRT_SECURE_NO_DEPRECATED</code>	37
4.17.2 Dokumentacja funkcji	37

4.17.2.1 divideByZero()	37
4.17.2.2 getUserChoice()	38
4.17.2.3 getUserInput()	38
4.17.2.4 manual()	38
4.17.2.5 printOnp()	38
4.18 Dokumentacja pliku UiFunctions.h	39
4.18.1 Dokumentacja funkcji	39
4.18.1.1 divideByZero()	39
4.18.1.2 getUserChoice()	39
4.18.1.3 getUserInput()	40
4.18.1.4 manual()	40
4.18.1.5 printOnp()	40
Indeks	41

Rozdział 1

Indeks struktur danych

1.1 Struktury danych

Tutaj znajdują się struktury danych wraz z ich krótkimi opisami:

Onp	5
Pair	6
Stack	7

Rozdział 2

Indeks plików

2.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

Calculator.c	9
Calculator.h	11
Constans.h	13
ConvertFunctions.c	14
ConvertFunctions.h	16
FileFunctions.c	19
FileFunctions.h	20
Main.c	22
OnpFunctions.c	23
OnpFunctions.h	27
StackFunctions.c	30
StackFunctions.h	33
Structures.h	35
UiFunctions.c	36
UiFunctions.h	39

Rozdział 3

Dokumentacja struktur danych

3.1 Dokumentacja struktury Onp

```
#include <Structures.h>
```

Pola danych

- double `value`
wartość elementu ONP
- enum `Type type`
typ znaku w równaniu
- struct `Onp * pPrev`
adres poprzedniego elementu równania
- struct `Onp * pNext`
adres następnego elementu równania

3.1.1 Opis szczegółowy

Struktura danych reprezentująca równanie przedstawione w odwrotnej notacji polskiej.

3.1.2 Dokumentacja pól

3.1.2.1 pNext

```
struct Onp* pNext
```

adres następnego elementu równania

3.1.2.2 pPrev

```
struct Onp* pPrev
```

adres poprzedniego elementu równania

3.1.2.3 type

```
enum Type type
```

typ znaku w równaniu

3.1.2.4 value

```
double value
```

wartość elementu ONP

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Structures.h](#)

3.2 Dokumentacja struktury Pair

```
#include <Structures.h>
```

Pola danych

- struct [Stack](#) * [pStack](#)
adres głowy stosu
- struct [Onp](#) * [pOnp](#)
adres ogona przekształconego równania

3.2.1 Opis szczegółowy

Struktura danych reprezentująca aktualny stan stosu i przekształconego równania

3.2.2 Dokumentacja pól

3.2.2.1 pOnp

```
struct Onp* pOnp
```

adres ogona przekształconego równania

3.2.2.2 pStack

```
struct Stack* pStack
```

adres głowy stosu

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Structures.h](#)

3.3 Dokumentacja struktury Stack

```
#include <Structures.h>
```

Pola danych

- double [priority](#)
prioritytet elementu na stosie
- enum [Type](#) type
typ znaku na stosie
- struct [Stack](#) * [pPrev](#)
adres poprzedniego elementu stosu

3.3.1 Opis szczegółowy

Stuktura danych reprezentująca stos, na którym składowne są wartości i znaki przy tworzeniu odwrotnej notacji polskiej

3.3.2 Dokumentacja pól

3.3.2.1 pPrev

```
struct Stack* pPrev
```

adres poprzedniego elementu stosu

3.3.2.2 prioryty

`double prioryty`

priorytet elementu na stosie

3.3.2.3 type

`enum Type type`

typ znaku na stosie

Dokumentacja dla tej struktury została wygenerowana z pliku:

- [Structures.h](#)

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku Calculator.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "Constans.h"
#include "Structures.h"
#include "ConvertFunctions.h"
#include "StackFunctions.h"
#include "Calculator.h"
```

Definicje

- `#define _CRT_SECURE_NO_DEPRECATED`

Funkcje

- double `calculator` (struct `Onp` *pHead)
- double `getMultiResult` (enum `Type` type, double head, double prev)
- double `getSingleResult` (enum `Type` type, double head)
- void `FreeMemory` (struct `Onp` *pHead, char *input)

4.1.1 Dokumentacja definicji

4.1.1.1 _CRT_SECURE_NO_DEPRECATED

```
#define _CRT_SECURE_NO_DEPRECATED
```

4.1.2 Dokumentacja funkcji

4.1.2.1 calculator()

```
double calculator (
    struct Onp * pHead )
```

Funkcja służy do obliczania wyniku równania przekonwertowanego na ONP

Parametry

<i>pHead</i>	adres pierwszego elementu listy jednokierunkowej przekonwertowanego na ONP równania
--------------	---

Zwraca

wynik równania

4.1.2.2 FreeMemory()

```
void FreeMemory (
    struct Onp * pHead,
    char * input )
```

Funkcja zwalnia pamięć po wykonaniu wszystkich operacji

Parametry

<i>pHead</i>	adres pierwszego elementu listy jednokierunkowej przekonwertowanego na ONP równania
<i>input</i>	adres łańcucha znaków zawierającego otrzymane od użytkownika równanie

4.1.2.3 getMultiResult()

```
double getMultiResult (
    enum Type type,
    double head,
    double prev )
```

Funkcja służy do obliczania wyniku pojedynczej operacji składającej się z dwóch liczb i operatora dwuelementowego

Parametry

<i>type</i>	typ operatora dwuelementowego
<i>head</i>	pierwsza liczba (bliższa pierwszego elementu stosu - znaku)
<i>prev</i>	druga liczba (liczba poprzedzająca pierwszą)

Zwraca

wynik operacji

4.1.2.4 getSingleResult()

```
double getSingleResult (
    enum Type type,
    double head )
```

Funkcja służy do obliczania wyniku pojedynczej operacji składającej się z liczby i operatora jednoelementowego

Parametry

<i>type</i>	typ operatora jednoelementowego
<i>head</i>	liczba najbliższa pierwszego elementu stosu - znaku

Zwraca

wynik operacji

4.2 Dokumentacja pliku Calculator.h

```
#include <stdio.h>
#include <stdlib.h>
```

Funkcje

- double `calculator` (struct `Onp` *pHead)
- double `getMultiResult` (enum `Type` type, double head, double prev)
- double `getSingleResult` (enum `Type` type, double head)
- void `FreeMemory` (struct `Onp` *pHead, char *input)

4.2.1 Dokumentacja funkcji

4.2.1.1 calculator()

```
double calculator (
    struct Onp * pHead )
```

Funkcja służy do obliczania wyniku równania przekonwertowanego na ONP

Parametry

<i>pHead</i>	adres pierwszego elementu listy jednokierunkowej przekonwertowanego na ONP równania
--------------	---

Zwraca

wynik równania

4.2.1.2 FreeMemory()

```
void FreeMemory (
    struct Onp * pHead,
    char * input )
```

Funkcja zwalnia pamięć po wykonaniu wszystkich operacji

Parametry

<i>pHead</i>	adres pierwszego elementu listy jednokierunkowej przekonwertowanego na ONP równania
<i>input</i>	adres łańcucha znaków zawierającego otrzymane od użytkownika równanie

4.2.1.3 getMultiResult()

```
double getMultiResult (
    enum Type type,
    double head,
    double prev )
```

Funkcja służy do obliczania wyniku pojedynczej operacji składającej się z dwóch liczb i operatora dwuelementowego

Parametry

<i>type</i>	typ operatora dwuelementowego
<i>head</i>	pierwsza liczba (bliższa pierwszego elementu stosu - znaku)
<i>prev</i>	druga liczba (liczba poprzedzająca pierwszą)

Zwraca

wynik operacji

4.2.1.4 getSingleResult()

```
double getSingleResult (
    enum Type type,
    double head )
```

Funkcja służy do obliczania wyniku pojedynczej operacji składającej się z liczby i operatora jednoelementowego

Parametry

<i>type</i>	typ operatora jednoelementowego
<i>head</i>	liczba najbliższa pierwszego elementu stosu - znaku

Zwraca

wynik operacji

4.3 Dokumentacja pliku Constans.h

```
#include <stdio.h>
#include <stdlib.h>
```

Definicje

- #define UNDEFINED -1
- #define MULTI_ARGUMENTS power
- #define INPUT_SIZE 512
- #define STACK_SIZE 64
- #define END_ARRAY '/0'

4.3.1 Dokumentacja definicji

4.3.1.1 END_ARRAY

```
#define END_ARRAY '/0'
```

4.3.1.2 INPUT_SIZE

```
#define INPUT_SIZE 512
```

4.3.1.3 MULTI_ARGUMENTS

```
#define MULTI_ARGUMENTS power
```

4.3.1.4 STACK_SIZE

```
#define STACK_SIZE 64
```

4.3.1.5 UNDEFINED

```
#define UNDEFINED -1
```

4.4 Dokumentacja pliku ConvertFunctions.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "Constans.h"
#include "Structures.h"
#include "ConvertFunctions.h"
```

Definicje

- `#define _CRT_SECURE_NO_DEPRECATED`

Funkcje

- enum `Type` `getType` (char input)
- int `getPriority` (enum `Type` type)
- char * `getOperation` (enum `Type` type)
- int `charToDigit` (char character)
- double `getFraction` (double `number`)

4.4.1 Dokumentacja definicji

4.4.1.1 _CRT_SECURE_NO_DEPRECATED

```
#define _CRT_SECURE_NO_DEPRECATED
```

4.4.2 Dokumentacja funkcji

4.4.2.1 charToDigit()

```
int charToDigit (
    char character )
```

Funkcja zamienia cyfrę zapisaną jako znak typu char na jej wartość typu int

Parametry

<i>character</i>	cyfra zapisaną jako znak
------------------	--------------------------

Zwraca

wartość tej liczby

4.4.2.2 getFraction()

```
double getFraction (
    double number )
```

Funkcja służy do tworzenia części ułamkowej liczby zmiennoprzecinkowej

Parametry

<i>number</i>	licznba występująca po znku '.' lub ','
---------------	---

Zwraca

ułamek dziesiętny

4.4.2.3 getOperation()

```
char* getOperation (
    enum Type type )
```

Funkcja zwraca oznaczenie danego typu znaku do wypisania na ekran lub do pliku

Parametry

<i>type</i>	typ znaku
-------------	-----------

Zwraca

oznaczenie danego typu znaku

4.4.2.4 getPriority()

```
int getPriority (
    enum Type type )
```

Funkcja zwraca priorytet podanego typu znaku

Parametry

<i>type</i>	typ znaku
-------------	-----------

Zwraca

prorytet przypisany danemu znakowi

4.4.2.5 getType()

```
enum Type getType (
    char input )
```

Funkcja zwraca typ podanego znaku

Parametry

<i>input</i>	pojedynczy znak równania
--------------	--------------------------

Zwraca

typ tego znaku

4.5 Dokumentacja pliku ConvertFunctions.h

```
#include <stdio.h>
#include <stdlib.h>
```

Funkcje

- enum `Type` `getType` (char input)
- int `getPriority` (enum `Type` type)
- char * `getOperation` (enum `Type` type)
- int `charToDigit` (char character)
- double `getFraction` (double `number`)

4.5.1 Dokumentacja funkcji

4.5.1.1 `charToDigit()`

```
int charToDigit (  
    char character )
```

Funkcja zamienia cyfrę zapisaną jako znak typu char na jej wartość typu int

Parametry

<code>character</code>	cyfra zapisana jako znak
------------------------	--------------------------

Zwraca

wartość tej liczby

4.5.1.2 `getFraction()`

```
double getFraction (  
    double number )
```

Funkcja służy do tworzenia części ułamkowej liczby zmiennoprzecinkowej

Parametry

<code>number</code>	licznba występująca po znku '.' lub ','
---------------------	---

Zwraca

ułamek dziesiętny

4.5.1.3 `getOperation()`

```
char* getOperation (
    enum Type type )
```

Funkcja zwraca oznaczenie danego typu znaku do wypisania na ekran lub do pliku

Parametry

<i>type</i>	typ znaku
-------------	-----------

Zwraca

oznaczenie danego typu znaku

4.5.1.4 `getPriority()`

```
int getPriority (
    enum Type type )
```

Funkcja zwraca priorytet podanego typu znaku

Parametry

<i>type</i>	typ znaku
-------------	-----------

Zwraca

prorytet przypisany danemu znakowi

4.5.1.5 `getType()`

```
enum Type getType (
    char input )
```

Funkcja zwraca typ podanego znaku

Parametry

<i>input</i>	pojedynczy znak równania
--------------	--------------------------

Zwraca

typ tego znaku

4.6 Dokumentacja pliku Debug/KalkulatorONP.vcxproj.FileListAbsolute.txt

4.7 Dokumentacja pliku FileFunctions.c

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "Constans.h"
#include "Structures.h"
#include "FileFunctions.h"
#include "UiFunctions.h"
#include "ConvertFunctions.h"
```

Definicje

- `#define _CRT_SECURE_NO_DEPRECATED`

Funkcje

- `char * readFile (char *argv1)`
- `void addResultToFile (double result)`
- `void addOnpToFile (struct Onp *pHead)`

4.7.1 Dokumentacja definicji

4.7.1.1 _CRT_SECURE_NO_DEPRECATED

```
#define _CRT_SECURE_NO_DEPRECATED
```

4.7.2 Dokumentacja funkcji

4.7.2.1 addOnpToFile()

```
void addOnpToFile (
    struct Onp * pHead )
```

Funkcja zapisuje kolejne elementu równania w odwrotnej notacji polskiej do pliku

Parametry

<i>pHead</i>	wskaźnik na pierwszy element równania
--------------	---------------------------------------

4.7.2.2 addResultToFile()

```
void addResultToFile (
    double result )
```

Funkcja zapisuje wynik równania do pliku

Parametry

<i>result</i>	wynik rónania
---------------	---------------

4.7.2.3 readFile()

```
char* readFile (
    char * argv1 )
```

Funkcja odczytuje równanie podane w pliku (podanym z lini poleceń)

Parametry

<i>argv1</i>	adres nazwy pliku do odczytu
--------------	------------------------------

Zwraca

równanie podne w pliku

4.8 Dokumentacja pliku FileFunctions.h

```
#include <stdio.h>
#include <stdlib.h>
```

Funkcje

- char * [readFile](#) (char *argv1)
- void [addResultToFile](#) (double result)
- void [addOnpToFile](#) (struct [Onp](#) *pHead)

4.8.1 Dokumentacja funkcji

4.8.1.1 addOnpToFile()

```
void addOnpToFile (
    struct Onp * pHead )
```

Funkcja zapisuje kolejny elementu równania w odwrotnej notacji polskiej do pliku

Parametry

<i>pHead</i>	wskaźnik na pierwszy element równania
--------------	---------------------------------------

4.8.1.2 addResultToFile()

```
void addResultToFile (
    double result )
```

Funkcja zapisuje wynik równania do pliku

Parametry

<i>result</i>	wynik równania
---------------	----------------

4.8.1.3 readFile()

```
char* readFile (
    char * argv1 )
```

Funkcja odczytuje równanie podane w pliku (podanym z lini poleceń)

Parametry

<i>argv1</i>	adres nazwy pliku do odczytu
--------------	------------------------------

Zwraca

równanie podne w pliku

4.9 Dokumentacja pliku history.txt

4.10 Dokumentacja pliku Main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <assert.h>
#include <vld.h>
#include <stdbool.h>
#include "Constans.h"
#include "Structures.h"
#include "FileFunctions.h"
#include "UiFunctions.h"
#include "ConvertFunctions.h"
#include "StackFunctions.h"
#include "OnpFunctions.h"
#include "Calculator.h"
```

Definicje

- `#define` [_CRT_SECURE_NO_DEPRECATED](#)

Funkcje

- `int` [main](#) (`int argc`, `char *argv[]`)

4.10.1 Dokumentacja definicji

4.10.1.1 `_CRT_SECURE_NO_DEPRECATED`

```
#define _CRT_SECURE_NO_DEPRECATED
```

4.10.2 Dokumentacja funkcji

4.10.2.1 main()

```
int main (
    int argc,
    char * argv[] )
```

4.11 Dokumentacja pliku OnpFunctions.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <assert.h>
#include <stdbool.h>
#include "Constans.h"
#include "Structures.h"
#include "ConvertFunctions.h"
#include "StackFunctions.h"
#include "OnpFunctions.h"
```

Definicje

- `#define _CRT_SECURE_NO_DEPRECATED`

Funkcje

- `struct Onp * onpBuilder (double result, char *input, bool rewrite)`
- `struct Onp * addToOnp (struct Onp *pTail, enum Type type, double value)`
- `struct Pair * addToPair (struct Onp *pTail, struct Stack *pStack)`
- `struct Onp * findOnpHead (struct Onp *pTail)`
- `bool setState (enum Type type)`
- `struct Onp * stateControl (struct Onp *pTail, bool toClear, bool toStop)`
- `void freeOnp (struct Onp *pHead)`

4.11.1 Dokumentacja definicji

4.11.1.1 `_CRT_SECURE_NO_DEPRECATED`

```
#define \_CRT\_SECURE\_NO\_DEPRECATED
```

4.11.2 Dokumentacja funkcji

4.11.2.1 addToOnp()

```
struct Onp* addToOnp (  
    struct Onp * tail,  
    enum Type type,  
    double value )
```

Funkcja służy do dodawania liczby lub znaku do przekonwertowanego na ONP równania

Parametry

<i>tail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>type</i>	typ dodawanego znaku lub liczby
<i>value</i>	wartość dodawanego znaku lub liczby

Zwraca

adres ostatniego elementu przekonwertowanego na ONP równania

4.11.2.2 addToPair()

```
struct Pair* addToPair (
    struct Onp * pTail,
    struct Stack * pStack )
```

Funkcja służy uaktualniania stanu ostatniego elementu przekonwertowanego na ONP równania i stosu

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>pStack</i>	adres pierwszego elementu stosu

Zwraca

adres aktualnej struktury **Pair**

4.11.2.3 findOnpHead()

```
struct Onp* findOnpHead (
    struct Onp * pTail )
```

Funkcja znajduje pierwszy element listy jednokierunkowej

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
--------------	--

Zwraca

adres pierwszego elementu przekonwertowanego na ONP równania

4.11.2.4 freeOnp()

```
void freeOnp (
    struct Onp * pHead )
```

Funkcja usuwa od początku listę jednokierunkową przekonwertowanego na ONP równania

Parametry

<i>pHead</i>	wskaźnik na pierwszy element listy – po usunięciu listy jest równy nullptr.
--------------	---

4.11.2.5 onpBuilder()

```
struct Onp* onpBuilder (
    double result,
    char * input,
    bool rewrite )
```

Funkcja służy do przekonwertowania równania ze standardowej notacji do odwrotnej notacji polskiej

Parametry

<i>result</i>	wynik poprzedniego równania
<i>input</i>	adres łańcucha znaków zawierającego otrzymane od użytkownika równanie
<i>rewrite</i>	warunek przepisania wyniku poprzedniego równania do nowego

Zwraca

adres pierwszego elementu przekonwertowanego na ONP równania

4.11.2.6 setState()

```
bool setState (
    enum Type type )
```

Funkcja ustala czy kalkulator powinien zakończyć działanie lub wyczyścić wyświetlacz

Parametry

<i>type</i>	typ napotkanego znaku
-------------	-----------------------

Zwraca

true albo false w zależności co zostanie ustalone

4.11.2.7 stateControl()

```
struct Onp* stateControl (
    struct Onp * pTail,
    bool toClear,
    bool toStop )
```

Funkcja sprawdza czy kalkulator spełnia warunki do kontynuowania pracy w niezmienionym stanie

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>toClear</i>	warunek usunięcia z wyświetlacza wszystkich danych
<i>toStop</i>	warunek zakończenia pracy kalkulatora

Zwraca

adres pierwszego elementu przekonwertowanego na ONP równania

4.12 Dokumentacja pliku OnpFunctions.h

```
#include <stdio.h>
#include <stdlib.h>
```

Funkcje

- struct Onp * onpBuilder (double result, char *input, bool rewrite)
- struct Onp * addToOnp (struct Onp *tail, enum Type type, double value)
- struct Pair * addToPair (struct Onp *pTail, struct Stack *pStack)
- struct Onp * findOnpHead (struct Onp *pTail)
- bool setState (enum Type type)
- struct Onp * stateControl (struct Onp *pTail, bool toClear, bool toStop)
- void freeOnp (struct Onp *pHead)

4.12.1 Dokumentacja funkcji

4.12.1.1 addToOnp()

```
struct Onp* addToOnp (
    struct Onp * tail,
    enum Type type,
    double value )
```

Funkcja służy do dodawania liczby lub znaku do przekonwertowanego na ONP równania

Parametry

<i>tail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>type</i>	typ dodawanego znaku lub liczby
<i>value</i>	wartość dodawanego znaku lub liczby

Zwraca

adres ostatniego elementu przekonwertowanego na ONP równania

4.12.1.2 addToPair()

```
struct Pair* addToPair (
    struct Onp * pTail,
    struct Stack * pStack )
```

Funkcja służy uaktualniania stanu ostatniego elementu przekonwertowanego na ONP równania i stosu

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>pStack</i>	adres pierwszego elementu stosu

Zwraca

adres aktualnej struktury `Pair`

4.12.1.3 findOnpHead()

```
struct Onp* findOnpHead (
    struct Onp * pTail )
```

Funkcja znajduje pierwszy element listy jednokierunkowej

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
--------------	--

Zwraca

adres pierwszego elementu przekonwertowanego na ONP równania

4.12.1.4 freeOnp()

```
void freeOnp (
    struct Onp * pHead )
```

Funkcja usuwa od początku listę jednokierunkową przekonwertowanego na ONP równania

Parametry

<i>pHead</i>	wskaźnik na pierwszy element listy – po usunięciu listy jest równy nullptr.
--------------	---

4.12.1.5 onpBuilder()

```
struct Onp* onpBuilder (
    double result,
    char * input,
    bool rewrite )
```

Funkcja służy do przekonwertowania równania ze standardowej notacji do odwrotnej notacji polskiej

Parametry

<i>result</i>	wynik poprzedniego równania
<i>input</i>	adres łańcucha znaków zawierającego otrzymane od użytkownika równanie
<i>rewrite</i>	warunek przepisania wyniku poprzedniego równania do nowego

Zwraca

adres pierwszego elementu przekonwertowanego na ONP równania

4.12.1.6 setState()

```
bool setState (
    enum Type type )
```

Funkcja ustala czy kalkulator powinien zakończyć działanie lub wyczyścić wyświetlacz

Parametry

<i>type</i>	typ napotkanego znaku
-------------	-----------------------

Zwraca

true albo false w zależności co zostanie ustalone

4.12.1.7 stateControl()

```
struct Onp* stateControl (
    struct Onp * pTail,
    bool toClear,
    bool toStop )
```

Funkcja sprawdza czy kalkulator spełnia warunki do kontynuowania pracy w niezmienionym stanie

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>toClear</i>	warunek usunięcia z wyświetlacza wszystkich danych
<i>toStop</i>	warunek zakończenia pracy kalkulatora

Zwraca

adres pierwszego elementu przekonwertowanego na ONP równania

4.13 Dokumentacja pliku StackFunctions.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>
#include <assert.h>
#include "Constans.h"
#include "Structures.h"
#include "ConvertFunctions.h"
#include "StackFunctions.h"
#include "OnpFunctions.h"
```

Definicje

- `#define _CRT_SECURE_NO_DEPRECATED`

Funkcje

- `struct Pair * stackAction (struct Stack *pStack, struct Onp *pTail, enum Type type)`
- `struct Pair * modifyStack (struct Stack *pStack, struct Onp *pTail, enum Type type)`
- `struct Stack * addToStack (struct Stack *pStack, enum Type type, double priority)`
- `struct Onp * emptyStack (struct Onp *pTail, struct Stack *pStack)`
- `struct Stack * removeFromStack (struct Stack *pStack)`

4.13.1 Dokumentacja definicji

4.13.1.1 _CRT_SECURE_NO_DEPRECATED

```
#define _CRT_SECURE_NO_DEPRECATED
```

4.13.2 Dokumentacja funkcji

4.13.2.1 addToStack()

```
struct Stack* addToStack (
    struct Stack * pStack,
    enum Type type,
    double priority )
```

Funkcja służy do dodawania znaku na początek stosu

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
<i>type</i>	typ dodawanego znaku
<i>priority</i>	prorytet przypisany dodawemu znakowi

Zwraca

adres pierwszego elementu stosu

4.13.2.2 emptyStack()

```
struct Onp* emptyStack (
    struct Onp * pTail,
    struct Stack * pStack )
```

Funkcja służy do przenoszenia znków, które pozostały na stosie po odczytaniu wszystkich wprowadzonych znaków, na koniec przekonwertowanego na ONP równania

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>pStack</i>	adres pierwszego elementu stosu

Zwraca

nowy adres ostatniego elementu przekonwertowanego na ONP równania

4.13.2.3 modifyStack()

```
struct Pair* modifyStack (
    struct Stack * pStack,
    struct Onp * pTail,
    enum Type type )
```

Funkcja skupia w sobie usuwanie danych ze stosu, dodawanie ich do przekonwertowanego na ONP równania oraz dodawanie nowych danych na stos

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>type</i>	typ danego znaku lub liczby

Zwraca

adres aktualnej struktury [Pair](#)

4.13.2.4 removeFromStack()

```
struct Stack* removeFromStack (
    struct Stack * pStack )
```

Funkcja służy do usuwania znaku z początku stosu

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
---------------	---------------------------------

Zwraca

adres aktualnego pierwszego elementu stosu po usunięciu poprzedniego

4.13.2.5 stackAction()

```
struct Pair* stackAction (
    struct Stack * pStack,
```

```
struct Onp * pTail,
enum Type type )
```

Funkcja skupia w sobie wszystkie operacje związane ze stosem dla danego typu znaku (dokładniej opisane w innych funkcjach)

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>type</i>	typ danego znaku lub liczby

Zwraca

adres aktualnej struktury [Pair](#)

4.14 Dokumentacja pliku StackFunctions.h

```
#include <stdio.h>
#include <stdlib.h>
```

Funkcje

- struct [Pair](#) * [stackAction](#) (struct [Stack](#) *pStack, struct [Onp](#) *pTail, enum [Type](#) type)
- struct [Pair](#) * [modifyStack](#) (struct [Stack](#) *pStack, struct [Onp](#) *pTail, enum [Type](#) type)
- struct [Stack](#) * [addToStack](#) (struct [Stack](#) *pStack, enum [Type](#) type, double priority)
- struct [Onp](#) * [emptyStack](#) (struct [Onp](#) *pTail, struct [Stack](#) *pStack)
- struct [Stack](#) * [removeFromStack](#) (struct [Stack](#) *pStack)

4.14.1 Dokumentacja funkcji

4.14.1.1 addToStack()

```
struct Stack* addToStack (
    struct Stack * pStack,
    enum Type type,
    double priority )
```

Funkcja służy do dodawania znaku na początek stosu

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
<i>type</i>	typ dodawanego znaku
<i>priority</i>	prorytet przypisany dodawemu znakowi

Zwraca

adres pierwszego elementu stosu

4.14.1.2 emptyStack()

```
struct Onp* emptyStack (
    struct Onp * pTail,
    struct Stack * pStack )
```

Funkcja służy do przenoszenia znków, które pozostały na stosie po odczytaniu wszystkich wprowadzonych znaków, na koniec przekonwertowanego na ONP równania

Parametry

<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>pStack</i>	adres pierwszego elementu stosu

Zwraca

nowy adres ostatniego elementu przekonwertowanego na ONP równania

4.14.1.3 modyfyStack()

```
struct Pair* modyfyStack (
    struct Stack * pStack,
    struct Onp * pTail,
    enum Type type )
```

Funkcja skupia w sobie usuwanie danych ze stosu, dodawanie ich do przekonwertowanego na ONP równania oraz dodawanie nowych danych na stos

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>type</i>	typ danego znaku lub liczby

Zwraca

adres aktualnej struktury **Pair**

4.14.1.4 removeFromStack()

```
struct Stack* removeFromStack (
    struct Stack * pStack )
```

Funkcja służy do usuwania znaku z początku stosu

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
---------------	---------------------------------

Zwraca

adres aktualnego pierwszego elementu stosu po usunięciu poprzedniego

4.14.1.5 stackAction()

```
struct Pair* stackAction (
    struct Stack * pStack,
    struct Onp * pTail,
    enum Type type )
```

Funkcja skupia w sobie wszystkie operacje związane ze stosem dla danego typu znaku (dokładniej opisane w innych funkcjach)

Parametry

<i>pStack</i>	adres pierwszego elementu stosu
<i>pTail</i>	adres ostatniego elementu przekonwertowanego na ONP równania
<i>type</i>	typ danego znaku lub liczby

Zwraca

adres aktualnej struktury [Pair](#)

4.15 Dokumentacja pliku Structures.h

```
#include <stdio.h>
#include <stdlib.h>
```

Struktury danych

- struct [Onp](#)
- struct [Stack](#)
- struct [Pair](#)

Wyliczenia

- enum `Type` {
 `number`, `dot`, `begin`, `end`,
 `plus`, `minus`, `mulipy`, `divide`,
 `power`, `percentage`, `sinus`, `cosinus`,
 `logarithm`, `negation`, `clear`, `done`,
 `unn` }

4.15.1 Dokumentacja typów wyliczanych

4.15.1.1 Type

enum `Type`

Typy znaków występujących w równaniu

Wartości wyliczeń

number	
dot	
begin	
end	
plus	
minus	
mulipy	
divide	
power	
percentage	
sinus	
cosinus	
logarithm	
negation	
clear	
done	
unn	

4.16 Dokumentacja pliku Test.txt

4.17 Dokumentacja pliku UiFunctions.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

```
#include "Constans.h"
#include "Structures.h"
#include "FileFunctions.h"
#include "UiFunctions.h"
#include "ConvertFunctions.h"
```

Definicje

- `#define _CRT_SECURE_NO_DEPRECATED`

Funkcje

- `char * getUserChoice (char *argv)`
- `char * getUserInput ()`
- `void printOnp (struct Onp *pHead)`
- `void manual ()`
- `void divideByZero (char *input)`

4.17.1 Dokumentacja definicji

4.17.1.1 `_CRT_SECURE_NO_DEPRECATED`

```
#define _CRT_SECURE_NO_DEPRECATED
```

4.17.2 Dokumentacja funkcji

4.17.2.1 `divideByZero()`

```
void divideByZero (
    char * input )
```

Funkcja sprawdza czy użytkownik próbuje dzielić przez 0

Parametry

<i>input</i>	podane równanie
--------------	-----------------

4.17.2.2 `getUserChoice()`

```
char* getUserChoice (
    char * argv )
```

Funkcja otrzymuje od użytkownika informację czy w wierszu linii poleceń podał nazwę pliku czy równanie

Parametry

<code>argv</code>	adres łańcucha znaków (otrzymanego równania lub nazwy pliku)
-------------------	--

Zwraca

otrzymane od użytkownika lub z pliku równanie

4.17.2.3 `getUserInput()`

```
char* getUserInput ( )
```

Funkcja otrzymuje równanie od użytkownika.

Zwraca

adres łańcucha znaków (otrzymanego równania)

4.17.2.4 `manual()`

```
void manual ( )
```

Funkcja wyświetla instrukcję dla użytkownika

4.17.2.5 `printOnp()`

```
void printOnp (
    struct Onp * pHead )
```

Funkcja wyświetla równanie przekonwertowane na odwrotną notację polską

Parametry

<code>pHead</code>	wskaźnik pierwszy element przekonwertowanego równania
--------------------	---

4.18 Dokumentacja pliku UiFunctions.h

```
#include <stdio.h>
#include <stdlib.h>
```

Funkcje

- char * [getUserChoice](#) (char *argv)
- char * [getUserInput](#) ()
- void [printOnp](#) (struct [Onp](#) *pHead)
- void [manual](#) ()
- void [divideByZero](#) (char *input)

4.18.1 Dokumentacja funkcji

4.18.1.1 [divideByZero\(\)](#)

```
void divideByZero (
    char * input )
```

Funkcja sprawdza czy użytkownik próbuje dzielić przez 0

Parametry

<i>input</i>	podane równanie
--------------	-----------------

4.18.1.2 [getUserChoice\(\)](#)

```
char* getUserChoice (
    char * argv )
```

Funkcja otrzymuje od użytkownika informację czy w wierszu linii poleceń podał nazwę pliku czy równanie

Parametry

<i>argv</i>	adres łańcucha znaków (otrzymanego równania lub nazwy pliku)
-------------	--

Zwraca

otrzymane od użytkownika lub z pliku równanie

4.18.1.3 `getUserInput()`

```
char* getUserInput ( )
```

Funkcja otrzymuje równanie od użytkownika.

Zwraca

adres łańcucha znaków (otrzymanego równania)

4.18.1.4 `manual()`

```
void manual ( )
```

Funkcja wyświetla instrukcję dla użytkownika

4.18.1.5 `printOnp()`

```
void printOnp (
    struct Onp * pHead )
```

Funkcja wyświetla równanie przekonwertowane na odwrotną notację polską

Parametry

<i>pHead</i>	wskaźnik pierwszy element przekonwertowanego równania
--------------	---

Indeks

- [_CRT_SECURE_NO_DEPRECATED](#)
 - [Calculator.c, 9](#)
 - [ConvertFunctions.c, 14](#)
 - [FileFunctions.c, 19](#)
 - [Main.c, 22](#)
 - [OnpFunctions.c, 23](#)
 - [StackFunctions.c, 31](#)
 - [UiFunctions.c, 37](#)
- [addOnpToFile](#)
 - [FileFunctions.c, 19](#)
 - [FileFunctions.h, 21](#)
- [addResultToFile](#)
 - [FileFunctions.c, 20](#)
 - [FileFunctions.h, 21](#)
- [addToOnp](#)
 - [OnpFunctions.c, 23](#)
 - [OnpFunctions.h, 27](#)
- [addToPair](#)
 - [OnpFunctions.c, 25](#)
 - [OnpFunctions.h, 28](#)
- [addToStack](#)
 - [StackFunctions.c, 31](#)
 - [StackFunctions.h, 33](#)
- [begin](#)
 - [Structures.h, 36](#)
- [calculator](#)
 - [Calculator.c, 10](#)
 - [Calculator.h, 11](#)
- [Calculator.c, 9](#)
 - [_CRT_SECURE_NO_DEPRECATED, 9](#)
 - [calculator, 10](#)
 - [FreeMemory, 10](#)
 - [getMultiResult, 10](#)
 - [getSingleResult, 11](#)
- [Calculator.h, 11](#)
 - [calculator, 11](#)
 - [FreeMemory, 12](#)
 - [getMultiResult, 12](#)
 - [getSingleResult, 12](#)
- [charToDigit](#)
 - [ConvertFunctions.c, 15](#)
 - [ConvertFunctions.h, 17](#)
- [clear](#)
 - [Structures.h, 36](#)
- [Constans.h, 13](#)
 - [END_ARRAY, 13](#)
 - [INPUT_SIZE, 13](#)
- [MULTI_ARGUMENTS, 13](#)
- [STACK_SIZE, 14](#)
- [UNDEFINED, 14](#)
- [ConvertFunctions.c, 14](#)
 - [_CRT_SECURE_NO_DEPRECATED, 14](#)
 - [charToDigit, 15](#)
 - [getFraction, 15](#)
 - [getOperation, 15](#)
 - [getPriority, 16](#)
 - [getType, 16](#)
- [ConvertFunctions.h, 16](#)
 - [charToDigit, 17](#)
 - [getFraction, 17](#)
 - [getOperation, 17](#)
 - [getPriority, 18](#)
 - [getType, 18](#)
- [cosinus](#)
 - [Structures.h, 36](#)
- [Debug/KalkulatorONP.vcxproj.FileListAbsolute.txt, 19](#)
- [divide](#)
 - [Structures.h, 36](#)
- [divideByZero](#)
 - [UiFunctions.c, 37](#)
 - [UiFunctions.h, 39](#)
- [done](#)
 - [Structures.h, 36](#)
- [dot](#)
 - [Structures.h, 36](#)
- [emptyStack](#)
 - [StackFunctions.c, 31](#)
 - [StackFunctions.h, 34](#)
- [end](#)
 - [Structures.h, 36](#)
- [END_ARRAY](#)
 - [Constans.h, 13](#)
- [FileFunctions.c, 19](#)
 - [_CRT_SECURE_NO_DEPRECATED, 19](#)
 - [addOnpToFile, 19](#)
 - [addResultToFile, 20](#)
 - [readFile, 20](#)
- [FileFunctions.h, 20](#)
 - [addOnpToFile, 21](#)
 - [addResultToFile, 21](#)
 - [readFile, 21](#)
- [findOnpHead](#)
 - [OnpFunctions.c, 25](#)
 - [OnpFunctions.h, 28](#)

- FreeMemory
 - Calculator.c, [10](#)
 - Calculator.h, [12](#)
- freeOnp
 - OnpFunctions.c, [25](#)
 - OnpFunctions.h, [28](#)
- getFraction
 - ConvertFunctions.c, [15](#)
 - ConvertFunctions.h, [17](#)
- getMultiResult
 - Calculator.c, [10](#)
 - Calculator.h, [12](#)
- getOperation
 - ConvertFunctions.c, [15](#)
 - ConvertFunctions.h, [17](#)
- getPriority
 - ConvertFunctions.c, [16](#)
 - ConvertFunctions.h, [18](#)
- getSingleResult
 - Calculator.c, [11](#)
 - Calculator.h, [12](#)
- getType
 - ConvertFunctions.c, [16](#)
 - ConvertFunctions.h, [18](#)
- getUserChoice
 - UiFunctions.c, [37](#)
 - UiFunctions.h, [39](#)
- getUserInput
 - UiFunctions.c, [38](#)
 - UiFunctions.h, [39](#)
- history.txt, [22](#)
- INPUT_SIZE
 - Constans.h, [13](#)
- logarithm
 - Structures.h, [36](#)
- main
 - Main.c, [22](#)
- Main.c, [22](#)
 - _CRT_SECURE_NO_DEPRECATED, [22](#)
 - main, [22](#)
- manual
 - UiFunctions.c, [38](#)
 - UiFunctions.h, [40](#)
- minus
 - Structures.h, [36](#)
- modifyStack
 - StackFunctions.c, [32](#)
 - StackFunctions.h, [34](#)
- muliply
 - Structures.h, [36](#)
- MULTI_ARGUMENTS
 - Constans.h, [13](#)
- negation
 - Structures.h, [36](#)
- number
 - Structures.h, [36](#)
- Onp, [5](#)
 - pNext, [5](#)
 - pPrev, [5](#)
 - type, [6](#)
 - value, [6](#)
- onpBuilder
 - OnpFunctions.c, [26](#)
 - OnpFunctions.h, [29](#)
- OnpFunctions.c, [23](#)
 - _CRT_SECURE_NO_DEPRECATED, [23](#)
 - addToOnp, [23](#)
 - addToPair, [25](#)
 - findOnpHead, [25](#)
 - freeOnp, [25](#)
 - onpBuilder, [26](#)
 - setState, [26](#)
 - stateControl, [27](#)
- OnpFunctions.h, [27](#)
 - addToOnp, [27](#)
 - addToPair, [28](#)
 - findOnpHead, [28](#)
 - freeOnp, [28](#)
 - onpBuilder, [29](#)
 - setState, [29](#)
 - stateControl, [30](#)
- Pair, [6](#)
 - pOnp, [6](#)
 - pStack, [7](#)
- percentage
 - Structures.h, [36](#)
- plus
 - Structures.h, [36](#)
- pNext
 - Onp, [5](#)
- pOnp
 - Pair, [6](#)
- power
 - Structures.h, [36](#)
- pprev
 - Onp, [5](#)
 - Stack, [7](#)
- printOnp
 - UiFunctions.c, [38](#)
 - UiFunctions.h, [40](#)
- priority
 - Stack, [7](#)
- pStack
 - Pair, [7](#)
- readFile
 - FileFunctions.c, [20](#)
 - FileFunctions.h, [21](#)
- removeFromStack
 - StackFunctions.c, [32](#)
 - StackFunctions.h, [34](#)

setState
 OnpFunctions.c, [26](#)
 OnpFunctions.h, [29](#)
sinus
 Structures.h, [36](#)
Stack, [7](#)
 pPrev, [7](#)
 priority, [7](#)
 type, [8](#)
STACK_SIZE
 Constans.h, [14](#)
stackAction
 StackFunctions.c, [32](#)
 StackFunctions.h, [35](#)
StackFunctions.c, [30](#)
 _CRT_SECURE_NO_DEPRECATED, [31](#)
 addToStack, [31](#)
 emptyStack, [31](#)
 modifyStack, [32](#)
 removeFromStack, [32](#)
 stackAction, [32](#)
StackFunctions.h, [33](#)
 addToStack, [33](#)
 emptyStack, [34](#)
 modifyStack, [34](#)
 removeFromStack, [34](#)
 stackAction, [35](#)
stateControl
 OnpFunctions.c, [27](#)
 OnpFunctions.h, [30](#)
Structures.h, [35](#)
 begin, [36](#)
 clear, [36](#)
 cosinus, [36](#)
 divide, [36](#)
 done, [36](#)
 dot, [36](#)
 end, [36](#)
 logarithm, [36](#)
 minus, [36](#)
 mulipy, [36](#)
 negation, [36](#)
 number, [36](#)
 percentage, [36](#)
 plus, [36](#)
 power, [36](#)
 sinus, [36](#)
 Type, [36](#)
 unn, [36](#)

Test.txt, [36](#)
Type
 Structures.h, [36](#)
type
 Onp, [6](#)
 Stack, [8](#)

UiFunctions.c, [36](#)
 _CRT_SECURE_NO_DEPRECATED, [37](#)

 divideByZero, [37](#)
 getUserChoice, [37](#)
 getUserInput, [38](#)
 manual, [38](#)
 printOnp, [38](#)
UiFunctions.h, [39](#)
 divideByZero, [39](#)
 getUserChoice, [39](#)
 getUserInput, [39](#)
 manual, [40](#)
 printOnp, [40](#)
UNDEFINED
 Constans.h, [14](#)
unn
 Structures.h, [36](#)

value
 Onp, [6](#)