KalkulatorONP

Wygenerowano przez Doxygen 1.8.17

Rozdział 1

Indeks struktur danych

1.1 Struktury danych

Tutaj znajdują się struktury danych wraz z ich krótkimi opisami:

Rozdział 2

Indeks plików

2.1 Lista plików

Tutaj znajduje się lista wszystkich plików z ich krótkimi opisami:

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 następniego elementu równania

3.1.1 Opis szczegółowy

Stuktura danych reprezentująca równanie przedstawione w odwrotnej notacji polskiej.

3.1.2 Dokumentacja pól

3.1.2.1 pNext

`struct Onp* pNext`

adres następniego 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

Stuktura 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 prioryty
  priorytet 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_DEPRECATE

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_DEPRECATE

```
#define _CRT_SECURE_NO_DEPRECATE
```

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

pHead	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

pHead	adres pierwszego elementu listy jednokierunkowej przekonwertowanego na ONP równania
input	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 pojedyńczej operacji składającej sie z dwóch liczb i operatora dwuelementowego

Parametry

type	typ operatora dwuelementowego
head	pierwsza liczba (bliższa pierwszego elementu stosu - znaku)
prev	druga liczba (liczba poprzedzajaca pierwszą)

Zwraca

wynik operacji

4.1.2.4 getSingleResult()

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

Funkcja służy do obliczania wyniku pojedyńczej operacji składającej sie z liczby i operatora jednoelementowego

Parametry

type	typ operatora jednoelementowego
head	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

pHead	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

pHead	adres pierwszego elementu listy jednokierunkowej przekonwertowanego na ONP równania
input	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 pojedyńczej operacji składającej sie z dwóch liczb i operatora dwuelementowego

Parametry

type	typ operatora dwuelementowego
head	pierwsza liczba (bliższa pierwszego elementu stosu - znaku)
prev	druga liczba (liczba poprzedzajaca pierwszą)

Zwraca

wynik operacji

4.2.1.4 getSingleResult()

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

Funkcja służy do obliczania wyniku pojedyńczej operacji składającej sie z liczby i operatora jednoelementowego

Parametry

type	typ operatora jednoelementowego
head	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_DEPRECATE

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_DEPRECATE

```
#define _CRT_SECURE_NO_DEPRECATE
```

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

character	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

number	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

type	typ znaku

Zwraca

oznaczenie danego typu znaku

4.4.2.4 getPriority()

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

Funkcja zwraca priorytet podanego typu znaku

Parametry

type	typ znaku

Zwraca

prorytet przypisany danemu znakowi

4.4.2.5 getType()

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

Funkcja zwraca typ podanego znaku

Parametry

input	pojedyńczy 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

character	cyfra zapisaną jako znak

Zwraca

wartość tej liczby

4.5.1.2 getFraction()

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

Funkcja służy do tworzenia części ułamkowej liczby zmiennoprzecinkowej

Parametry

number	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

type	typ znaku

Zwraca

oznaczenie danego typu znaku

4.5.1.4 getPriority()

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

Funkcja zwraca priorytet podanego typu znaku

Parametry

type	typ znaku

Zwraca

prorytet przypisany danemu znakowi

4.5.1.5 getType()

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

Funkcja zwraca typ podanego znaku

Parametry

input	pojedyńczy 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_DEPRECATE

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_DEPRECATE

```
#define _CRT_SECURE_NO_DEPRECATE
```

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

pHead	wskaźnik na pierszy element równania

4.7.2.2 addResultToFile()

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

Funkcja zapisuje wynik równania do pliku

Parametry

result	wynik rónania

4.7.2.3 readFile()

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

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

Parametry

argv1	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 kolejne elementu równania w odwrotnej notacji polskiej do pliku

Parametry

pHead	wskaźnik na pierszy element równania

4.8.1.2 addResultToFile()

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

Funkcja zapisuje wynik równania do pliku

Parametry

result	wynik rónania

4.8.1.3 readFile()

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

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

Parametry

argv1	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_DEPRECATE

Funkcje

- int main (int argc, char ∗argv[ ])

4.10.1 Dokumentacja definicji

4.10.1.1 _CRT_SECURE_NO_DEPRECATE

```
#define _CRT_SECURE_NO_DEPRECATE
```

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_DEPRECATE

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_DEPRECATE

```
#define _CRT_SECURE_NO_DEPRECATE
```

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

tail	adres ostatniego elementu przekonwertowanego na ONP równania
type	typ dodawnaego znaku lub liczby
value	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

pTail	adres ostatniego elementu przekonwertowanego na ONP równania
pStack	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

pTail	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

pHead	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

result	wynik poprzedniego równania
input	adres łańcucha znaków zawierającego otrzymane od urzytkownika równanie
rewrite	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

type	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

pTail	adres ostatniego elementu przekonwertowanego na ONP równania
toClear	warunek usunięcia z wyświetlacza wszystkich danych
toStop	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

tail	adres ostatniego elementu przekonwertowanego na ONP równania
type	typ dodawnaego znaku lub liczby
value	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

pTail	adres ostatniego elementu przekonwertowanego na ONP równania
pStack	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

pTail	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

pHead	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

result	wynik poprzedniego równania
input	adres łańcucha znaków zawierającego otrzymane od urzytkownika równanie
rewrite	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

type	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

pTail	adres ostatniego elementu przekonwertowanego na ONP równania
toClear	warunek usunięcia z wyświetlacza wszystkich danych
toStop	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_DEPRECATE

Funkcje

- struct Pair * stackAction (struct Stack *pStack, struct Onp *pTail, enum Type type)
- struct Pair * modyfyStack (struct Stack *pStack, struct Onp *pTail, enum Type type)
- struct Stack * addToStack (struct Stack *pStack, enum Type type, double prioryty)
- 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_DEPRECATE

```
#define _CRT_SECURE_NO_DEPRECATE
```

4.13.2 Dokumentacja funkcji

4.13.2.1 addToStack()

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

Funkcja służy do dodawania znaku na początek stosu

Parametry

pStack	adres pierwszego elementu stosu
type	typ dodawanego znaku
prioryty	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

pTail	adres ostatniego elementu przekonwertowanego na ONP równania
pStack	adres pierwszego elementu stosu

Zwraca

nowy adres ostatniego elementu przekonwertowanego na ONP równania

4.13.2.3 modyfyStack()

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

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

Parametry

pStack	adres pierwszego elementu stosu
pTail	adres ostatniego elementu przekonwertowanego na ONP równania
type	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 poczatku stosu

Parametry

pStack	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

pStack	adres pierwszego elementu stosu
pTail	adres ostatniego elementu przekonwertowanego na ONP równania
type	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 * modyfyStack (struct Stack *pStack, struct Onp *pTail, enum Type type)
- struct Stack * addToStack (struct Stack *pStack, enum Type type, double prioryty)
- 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 prioryty )
```

Funkcja służy do dodawania znaku na początek stosu

Parametry

pStack	adres pierwszego elementu stosu
type	typ dodawanego znaku
prioryty	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

pTail	adres ostatniego elementu przekonwertowanego na ONP równania
pStack	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, dodawnie ich do przekonwertowanego na ONP równania oraz dodawanie nowych danych na stos

Parametry

pStack	adres pierwszego elementu stosu
pTail	adres ostatniego elementu przekonwertowanego na ONP równania
type	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 poczatku stosu

Parametry

pStack	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

pStack	adres pierwszego elementu stosu
pTail	adres ostatniego elementu przekonwertowanego na ONP równania
type	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_DEPRECATE

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_DEPRECATE

```
#define _CRT_SECURE_NO_DEPRECATE
```

4.17.2 Dokumentacja funkcji

4.17.2.1 divideByZero()

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

Funkcja sprawdza czy użytkownik próbuje dzielć przez 0

Parametry

input	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

argv	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 notacje polską

Parametry

pHead	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 dzielć przez 0

Parametry

input	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

argv	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 notacje polską

Parametry

pHead	wskaźnik pierwszy element przekonwertowanego równania

Indeks