Invoice

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 udokumentowanych plików z ich krótkimi opisami:

Rozdział 3

Dokumentacja struktur danych

3.1 Dokumentacja struktury Address

`#include <Address.h>`

Pola danych

- char street [50]

  nazwa ulicy
- char homeNumber [10]

  numer budynku
- char postalCode [7]

  kod pocztowy
- char city [50]

  nazwa miasta

3.1.1 Opis szczegółowy

Struktura adresu.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Address.h

3.2 Dokumentacja struktury Invoice

`#include <Invoice.h>`

Pola danych

- char documentNumber [20]
  unikalny numer faktury
- char date [20]
  data
- char paymentDeadline [20]
  termin płatności
- float netSum
  suma netto
- float taxSum
  suma vat
- float grossSum
  suma brutto
- float paid
  zapłacona kwota
- struct Person * solder
  wskaźnik na osobę sprzedawcy
- struct Person * buyer
  wskaźnik na osobę nabywcy
- struct Ware * wHead
  wskaźnik na głowę towarów/usług
- struct Invoice * iNext
  wskaźnik na następny element listy faktur

3.2.1 Opis szczegółowy

Struktura faktury.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Invoice.h

3.3 Dokumentacja struktury Person

```
#include <Person.h>
```

Pola danych

- char companyName [50]
  nazwa firmy
- char name [50]
  imię osoby
- char surname [50]
  nazwisko osoby
- char nip [11]
  nip
- char accountNumber [27]
  numer konta
- struct Address * address
  wskaźnik na adres

3.3.1 Opis szczegółowy

Struktura osoby.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Person.h

3.4 Dokumentacja struktury Ware

```
#include <Ware.h>
```

Pola danych

- char name [150]
  nazwa
- int amount
  ilość
- float netPrice
  cena netto
- float netValue
  wartość netto
- float tax
  stawka VAT
- float taxValue
  VAT.
- float grossValue
  wartość brutto
- struct Ware * wNext
  wskaźnik na następny element listy towarów/usług

3.4.1 Opis szczegółowy

Struktura towaru/usługi.

Dokumentacja dla tej struktury została wygenerowana z pliku:

- Ware.h

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku Address.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "Address.h"
#include "../functions/utilities.h"
#include "../functions/input.h"
```

Funkcje

- struct Address ∗ createAddress ()
- void fillAddress (struct Address ∗address, char street[ ], char homeNumber[ ], char postalCode[ ], char city[ ])
- void readDataAddress (struct Address ∗address)
- void editAddress (struct Address ∗address)

4.1.1 Dokumentacja funkcji

4.1.1.1 createAddress()

```
struct Address* createAddress ( )
```

Konstruktor alokuje pamięć na strukturę adresu.

Zwraca

wskaźnik na utworzoną strukturę adresu

4.1.1.2 editAddress()

```
void editAddress (
            struct Address * address )
```

Funkcja prosi o i odczytuje nowe dane adresowe do edycji z konsoli oraz przypisuje je do adresu.

Parametry

address	wskaźnik na adres do który będzie edytowany

4.1.1.3 fillAddress()

```
void fillAddress (
            struct Address * address,
            char street[],
            char homeNumber[],
            char postalCode[],
            char city[] )
```

Funkcja wypełnia strukturę adresu danymi.

Parametry

address	wskaźnik na adres to wypełnienia
street	tablica znaków zawierająca nazwę ulicy
homeNumber	tablica znaków zawierająca numer budynku
postalCode	tablica znaków zawierająca kod pocztowy
city	tablica znaków zawierająca nazwę miasta

4.1.1.4 readDataAddress()

```
void readDataAddress (
            struct Address * address )
```

Funkcja prosi o i odczytuje dane adresowe z konsoli oraz przypisuje je do adresu.

Parametry

address	wskaźnik na adres do którego będą przekazane dane

4.2 Dokumentacja pliku Address.h

Struktury danych

- struct Address

Funkcje

- struct Address ∗ createAddress ()
- void fillAddress (struct Address ∗address, char street[ ], char homeNumber[ ], char postalCode[ ], char city[ ])
- void readDataAddress (struct Address ∗address)
- void editAddress (struct Address ∗address)

4.2.1 Dokumentacja funkcji

4.2.1.1 createAddress()

```
struct Address* createAddress ( )
```

Konstruktor alokuje pamięć na strukturę adresu.

Zwraca

wskaźnik na utworzoną strukturę adresu

4.2.1.2 editAddress()

```
void editAddress (
            struct Address * address )
```

Funkcja prosi o i odczytuje nowe dane adresowe do edycji z konsoli oraz przypisuje je do adresu.

Parametry

address	wskaźnik na adres do który będzie edytowany

4.2.1.3 fillAddress()

```
void fillAddress (
            struct Address * address,
            char street[],
            char homeNumber[],
            char postalCode[],
            char city[] )
```

Funkcja wypełnia strukturę adresu danymi.

Parametry

address	wskaźnik na adres to wypełnienia
street	tablica znaków zawierająca nazwę ulicy
homeNumber	tablica znaków zawierająca numer budynku
postalCode	tablica znaków zawierająca kod pocztowy
city	tablica znaków zawierająca nazwę miasta

4.2.1.4 readDataAddress()

```
void readDataAddress (
            struct Address * address )
```

Funkcja prosi o i odczytuje dane adresowe z konsoli oraz przypisuje je do adresu.

Parametry

address	wskaźnik na adres do którego będą przekazane dane

4.3 Dokumentacja pliku input.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "../structures/Address.h"
#include "../structures/Person.h"
#include "../structures/Ware.h"
#include "../structures/Invoice.h"
#include "../structures/InvoiceBox.h"
#include "input.h"
#include "utilities.h"
```

Funkcje

- void readDataFromFile (struct Invoice **invoiceList, const char *PATH, char *fileName)
- void readLine (char *target, int length)
- int readInteger ()
- float readNumber ()
- float readPercentage ()
- char * readDate ()
- int readSelectOption (int start, int end)
- int readYesOrNoOption (char text[ ])

4.3.1 Dokumentacja funkcji

4.3.1.1 readDataFromFile()

```
void readDataFromFile (
            struct Invoice ** invoiceList,
            const char * PATH,
            char * fileName )
```

Funkcja odczytuje dane z pliku i na ich podstawie tworzy jednokierunkową listę faktur.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę jednokierunkowej listy faktur
PATH	ścieżka do folderu z plikiem do odczytu, gdy "" to odczyt z folderu w którym jest program
fileName	nazwa pliku do odczytu (wymagane rozszerzenie - ".txt")

4.3.1.2 readDate()

```
char* readDate ( )
```

Funkcja odczytuje datę z konsoli w odpowiednim formacie (dd.mm.rrrr).w przypadku błędnych danych prosi o ponowne podanie daty.

Zwraca

odczytana data jako string

4.3.1.3 readInteger()

```
int readInteger ( )
```

Funkcja odczytuje z konsoli liczbę całkowitą.

Zwraca

odczytana liczba całkowita

4.3.1.4 readLine()

```
void readLine (
            char * target,
            int length )
```

Funkcja odczytuje dane wyprowadzone na konsolę i zapisuje je do przekazanego stringa.

Parametry

target	string do którego będą zapisane dane odczytane przez funkcję
length	maksymalna długość zapisywanych w stringu danych

4.3.1.5 readNumber()

```
float readNumber ( )
```

Funkcja odczytuje liczbę rzeczywistą z konsoli, w przypadku błędnych danych prosi o ponowne podanie liczby.

Zwraca

odczytana liczba rzeczywista

4.3.1.6 readPercentage()

```
float readPercentage ( )
```

Funkcja odczytuje liczbę rzeczywistą z konsoli, w przypadku błędnych danych prosi o ponowne podanie liczby.

Zwraca

odczytana liczba rzeczywista podana w procentach

4.3.1.7 readSelectOption()

```
int readSelectOption (
            int start,
            int end )
```

Funkcja odczytuje liczbę całkowitą, w przypadku błędnych danych prosi o ponowne podanie liczby.

Parametry

start	początek przedziału z jakiego ma być liczba
end	koniec przedziału z jakiego ma być liczba

Zwraca

odczytana liczba całkowita

4.3.1.8 readYesOrNoOption()

```
int readYesOrNoOption (
            char text[] )
```

Funkcja odczytuje wyniki wyboru [Y/n] gdzie domyślną wartością jest "Y".

Parametry

text	tekst jaki ma być wyświetlony przed poproszeniem o wybór

Zwraca

1 dla "Y" (zgody), 0 dla "n" (niezgody)

4.4 Dokumentacja pliku input.h

Funkcje

- void readDataFromFile (struct Invoice **invoiceList, const char *PATH, char *fileName)
- void readLine (char *target, int length)
- int readInteger ()
- float readNumber ()
- float readPercentage ()
- char * readDate ()
- int readSelectOption (int start, int end)
- int readYesOrNoOption (char text[])

4.4.1 Dokumentacja funkcji

4.4.1.1 readDataFromFile()

```
void readDataFromFile (
            struct Invoice ** invoiceList,
            const char * PATH,
            char * fileName )
```

Funkcja odczytuje dane z pliku i na ich podstawie tworzy jednokierunkową listę faktur.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę jednokierunkowej listy faktur
PATH	ścieżka do folderu z plikiem do odczytu, gdy "" to odczyt z folderu w którym jest program
fileName	nazwa pliku do odczytu (wymagane rozszerzenie - ".txt")

4.4.1.2 readDate()

```
char* readDate ( )
```

Funkcja odczytuje datę z konsoli w odpowiednim formacie (dd.mm.rrrr).w przypadku błędnych danych prosi o ponowne podanie daty.

Zwraca

odczytana data jako string

4.4.1.3 readInteger()

```
int readInteger ( )
```

Funkcja odczytuje z konsoli liczbę całkowitą.

Zwraca

odczytana liczba całkowita

4.4.1.4 readLine()

```
void readLine (
            char * target,
            int length )
```

Funkcja odczytuje dane wyprowadzone na konsolę i zapisuje je do przekazanego stringa.

Parametry

target	string do którego będą zapisane dane odczytane przez funkcję
length	maksymalna długość zapisywanych w stringu danych

4.4.1.5 readNumber()

```
float readNumber ( )
```

Funkcja odczytuje liczbę rzeczywistą z konsoli, w przypadku błędnych danych prosi o ponowne podanie liczby.

Zwraca

odczytana liczba rzeczywista

4.4.1.6 readPercentage()

```
float readPercentage ( )
```

Funkcja odczytuje liczbę rzeczywistą z konsoli, w przypadku błędnych danych prosi o ponowne podanie liczby.

Zwraca

odczytana liczba rzeczywista podana w procentach

4.4.1.7 readSelectOption()

```
int readSelectOption (
            int start,
            int end )
```

Funkcja odczytuje liczbę całkowitą, w przypadku błędnych danych prosi o ponowne podanie liczby.

Parametry

start	początek przedziału z jakiego ma być liczba
end	koniec przedziału z jakiego ma być liczba

Zwraca

odczytana liczba całkowita

4.4.1.8 readYesOrNoOption()

```
int readYesOrNoOption (
            char text[] )
```

Funkcja odczytuje wyniki wyboru [Y/n] gdzie domyślną wartością jest "Y".

Parametry

text	tekst jaki ma być wyświetlony przed poproszeniem o wybór

Zwraca

1 dla "Y" (zgody), 0 dla "n" (niezgody)

4.5 Dokumentacja pliku Invoice.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "Address.h"
#include "Person.h"
#include "Ware.h"
#include "Invoice.h"
#include "InvoiceBox.h"
#include "../functions/utilities.h"
#include "../functions/input.h"
#include "../functions/output.h"
```

Funkcje

- struct Invoice * createInvoice ()
- void fillInvoice (struct Invoice *invoice, struct Person *solder, struct Person *buyer, char documentNumber[ ], char date[ ])
- void showInvoice (struct Invoice *invoice)
- void addWare (struct Invoice *invoice, struct Ware *ware)
- int readDataInvoice (struct Invoice *invoice)
- void issuingInvoice (struct Invoice **invoiceList)
- void putWareList (struct Invoice *invoice)
- void calculateSumWares (struct Invoice *invoice)
- void deleteInvoice (struct Invoice *invoice)
- void editInvoice (struct Invoice *invoiceList, struct Invoice *invoice)
- void wareOptions (struct Invoice *invoice, struct Ware *ware)
- void showWareList (struct Invoice *invoice)
- struct Ware * selectWare (struct Invoice *invoice)
- int lengthWareList (struct Ware *ware)
- void deleteWareFromList (struct Invoice *invoice, struct Ware *ware)
- char * generateUniqueID (struct Invoice *invoiceList)
- void showInvoiceToPaid (struct Invoice *invoice)

4.5.1 Dokumentacja funkcji

4.5.1.1 addWare()

```
void addWare (
            struct Invoice * invoice,
            struct Ware * ware )
```

Funkcja dodaje do faktury nowy towar/usługę.

Parametry

invoice	wskaźnik na fakturę
ware	wskaźnik na towar/usługę która ma być dodana

4.5.1.2 calculateSumWares()

```
void calculateSumWares (
            struct Invoice * invoice )
```

Funkcja oblicza sumy wartości netto brutto i VAT na podstawie listy towarów/usług, i przypisuję je do faktury.

Parametry

invoice	wskaźnik na fakturę

4.5.1.3 createInvoice()

```
struct Invoice* createInvoice ( )
```

Konstruktor wypełnia strukturę faktury domyślnymi danymi i alokuje pamięć na nią.

Zwraca

wskaźnik na utworzoną strukturę faktury

4.5.1.4 deleteInvoice()

```
void deleteInvoice (
            struct Invoice * invoice )
```

Funkcja usuwa fakturę i zwalnia pamięć.

Parametry

invoice	wskaźnik na fakturę do usunięcia

4.5.1.5 deleteWareFromList()

```
void deleteWareFromList (
            struct Invoice * invoice,
            struct Ware * ware )
```

Funkcja usuwa towar/usługę z listy towarów/usług.

Parametry

invoice	wskaźnik do faktury w której jest lista towarów/usług
ware	wskaźnik na towar/usługę do usunięcia

4.5.1.6 editInvoice()

```
void editInvoice (
            struct Invoice * invoiceList,
            struct Invoice * invoice )
```

Funkcja prosi o i odczytuje dane do edycji faktury.

Parametry

invoiceList	wskaźnik na głowę listy faktur
invoice	wskaźnik na fakturę do edycji

4.5.1.7 fillInvoice()

```
void fillInvoice (
            struct Invoice * invoice,
            struct Person * solder,
            struct Person * buyer,
            char documentNumber[],
            char date[] )
```

Funkcja wypełnia strukturę faktury danymi.

Parametry

invoice	wskaźnik na fakturę do wypełnienia
solder	wskaźnik na sprzedawcę
buyer	wskaźnik na nabywcę
documentNumber	tablica znaków zawierająca numer faktury
date	tablica znaków zawierająca datę

4.5.1.8 generateUniqueID()

```
char* generateUniqueID (
            struct Invoice * invoiceList )
```

Funkcja generuje unikalny numer dokumentu na podstawie aktualnej daty i listy faktur.

Parametry

invoiceList	wskaźnik na głowę listy faktur

4.5.1.9 issuingInvoice()

```
void issuingInvoice (
            struct Invoice ** invoiceList )
```

Funkcja prosi o i odczytuje dane do wystawienia nowej faktury.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur

4.5.1.10 lengthWareList()

```
int lengthWareList (
            struct Ware * ware )
```

Funkcja oblicza długość listy towarów/usług.

Parametry

ware	wskaźnik na głowę listy towarów/usług

Zwraca

długość listy

4.5.1.11 putWareList()

```
void putWareList (
            struct Invoice * invoice )
```

Funkcja tworzy listę towarów/usług podczas wystawiania faktury.

Parametry

invoice	wskaźnik na fakture

4.5.1.12 readDataInvoice()

```
int readDataInvoice (
            struct Invoice * invoice )
```

Funkcja prosi o i odczytuje dane faktury z konsoli oraz przypisuje je do faktury.

Parametry

invoice	wskaźnik na fakturę do której będą przekazane dane

Zwraca

1 gdy faktura jest opłacona 0 w przeciwnym wypadku

4.5.1.13 selectWare()

```
struct Ware* selectWare (
            struct Invoice * invoice )
```

Funkcja prosi o numer towaru/usługi na podstawie wcześniej wyświetlonej listy.

Parametry

invoice	wskaźnik do faktury w której jest lista towarów/usług

Zwraca

wskaźnik na wybraną towar/usługę

4.5.1.14 showInvoice()

```
void showInvoice (
            struct Invoice * invoice )
```

Funkcja wyświetla fakturę w konsoli.

Parametry

wskaźnik	na fakturę do wyświetlenia

4.5.1.15 showInvoiceToPaid()

```
void showInvoiceToPaid (
            struct Invoice * invoice )
```

Funkcja wyświetla dane do przelewu i prosi o potwierdzenie.

Parametry

invoice	wskaźnik na fakturę, która jest do opłacenia

4.5.1.16 showWareList()

```
void showWareList (
            struct Invoice * invoice )
```

Funkcja wyświetla listę towarów/usług.

Parametry

invoice	wskaźnik do faktury w której jest lista towarów/usług do wyświetlenia

4.5.1.17 wareOptions()

```
void wareOptions (
            struct Invoice * invoice,
            struct Ware * ware )
```

Funkcja wyświetla i prosi o wybór opcji akcji dla towaru/usługi.

Parametry

invoice	wskaźnik na fakturę
ware	wskaźnik na wybrany towar/usługę

4.6 Dokumentacja pliku Invoice.h

Struktury danych

- struct Invoice

Funkcje

- struct Invoice ∗ createInvoice ()
- void fillInvoice (struct Invoice ∗invoice, struct Person ∗solder, struct Person ∗buyer, char documentNumber[ ], char date[ ])
- void showInvoice (struct Invoice ∗invoice)
- void addWare (struct Invoice ∗invoice, struct Ware ∗ware)
- int readDataInvoice (struct Invoice ∗invoice)
- void issuingInvoice (struct Invoice ∗∗invoiceList)
- void putWareList (struct Invoice ∗invoice)
- void calculateSumWares (struct Invoice ∗invoice)
- void deleteInvoice (struct Invoice ∗invoice)
- void editInvoice (struct Invoice ∗invoiceList, struct Invoice ∗invoice)
- void wareOptions (struct Invoice ∗invoice, struct Ware ∗ware)
- void showWareList (struct Invoice ∗invoice)
- struct Ware ∗ selectWare (struct Invoice ∗invoice)
- int lengthWareList (struct Ware ∗ware)
- void deleteWareFromList (struct Invoice ∗invoice, struct Ware ∗ware)
- char ∗ generateUniqueID (struct Invoice ∗invoiceList)
- void showInvoiceToPaid (struct Invoice ∗invoice)

4.6.1 Dokumentacja funkcji

4.6.1.1 addWare()

```
void addWare (
            struct Invoice * invoice,
            struct Ware * ware )
```

Funkcja dodaje do faktury nowy towar/usługę.

Parametry

invoice	wskaźnik na fakturę
ware	wskaźnik na towar/usługę która ma być dodana

4.6.1.2 calculateSumWares()

```
void calculateSumWares (
            struct Invoice * invoice )
```

Funkcja oblicza sumy wartości netto brutto i VAT na podstawie listy towarów/usług, i przypisuję je do faktury.

Parametry

invoice	wskaźnik na fakturę

4.6.1.3 createInvoice()

```
struct Invoice* createInvoice ( )
```

Konstruktor wypełnia strukturę faktury domyślnymi danymi i alokuje pamięć na nią.

Zwraca

wskaźnik na utworzoną strukturę faktury

4.6.1.4 deleteInvoice()

```
void deleteInvoice (
            struct Invoice * invoice )
```

Funkcja usuwa fakturę i zwalnia pamięć.

Parametry

invoice	wskaźnik na fakturę do usunięcia

4.6.1.5 deleteWareFromList()

```
void deleteWareFromList (
            struct Invoice * invoice,
            struct Ware * ware )
```

Funkcja usuwa towar/usługę z listy towarów/usług.

Parametry

invoice	wskaźnik do faktury w której jest lista towarów/usług
ware	wskaźnik na towar/usługę do usunięcia

4.6.1.6 editInvoice()

```
void editInvoice (
            struct Invoice * invoiceList,
            struct Invoice * invoice )
```

Funkcja prosi o i odczytuje dane do edycji faktury.

Parametry

invoiceList	wskaźnik na głowę listy faktur
invoice	wskaźnik na fakturę do edycji

4.6.1.7 fillInvoice()

```
void fillInvoice (
            struct Invoice * invoice,
            struct Person * solder,
            struct Person * buyer,
            char documentNumber[],
            char date[] )
```

Funkcja wypełnia strukturę faktury danymi.

Parametry

invoice	wskaźnik na fakturę do wypełnienia
solder	wskaźnik na sprzedawcę
buyer	wskaźnik na nabywcę
documentNumber	tablica znaków zawierająca numer faktury
date	tablica znaków zawierająca datę

4.6.1.8 generateUniqueID()

```
char* generateUniqueID (
            struct Invoice * invoiceList )
```

Funkcja generuje unikalny numer dokumentu na podstawie aktualnej daty i listy faktur.

Parametry

invoiceList	wskaźnik na głowę listy faktur

4.6.1.9 issuingInvoice()

```
void issuingInvoice (
            struct Invoice ** invoiceList )
```

Funkcja prosi o i odczytuje dane do wystawienia nowej faktury.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur

4.6.1.10 lengthWareList()

```
int lengthWareList (
            struct Ware * ware )
```

Funkcja oblicza długość listy towarów/usług.

Parametry

ware	wskaźnik na głowę listy towarów/usług

Zwraca

długość listy

4.6.1.11 putWareList()

```
void putWareList (
            struct Invoice * invoice )
```

Funkcja tworzy listę towarów/usług podczas wystawiania faktury.

Parametry

invoice	wskaźnik na fakture

4.6.1.12 readDataInvoice()

```
int readDataInvoice (
            struct Invoice * invoice )
```

Funkcja prosi o i odczytuje dane faktury z konsoli oraz przypisuje je do faktury.

Parametry

invoice	wskaźnik na fakturę do której będą przekazane dane

Zwraca

1 gdy faktura jest opłacona 0 w przeciwnym wypadku

4.6.1.13 selectWare()

```
struct Ware* selectWare (
            struct Invoice * invoice )
```

Funkcja prosi o numer towaru/usługi na podstawie wcześniej wyświetlonej listy.

Parametry

invoice	wskaźnik do faktury w której jest lista towarów/usług

Zwraca

wskaźnik na wybraną towar/usługę

4.6.1.14 showInvoice()

```
void showInvoice (
            struct Invoice * invoice )
```

Funkcja wyświetla fakturę w konsoli.

Parametry

wskaźnik	na fakturę do wyświetlenia

4.6.1.15 showInvoiceToPaid()

```
void showInvoiceToPaid (
            struct Invoice * invoice )
```

Funkcja wyświetla dane do przelewu i prosi o potwierdzenie.

Parametry

invoice	wskaźnik na fakturę, która jest do opłacenia

4.6.1.16 showWareList()

```
void showWareList (
            struct Invoice * invoice )
```

Funkcja wyświetla listę towarów/usług.

Parametry

invoice	wskaźnik do faktury w której jest lista towarów/usług do wyświetlenia

4.6.1.17 wareOptions()

```
void wareOptions (
            struct Invoice * invoice,
            struct Ware * ware )
```

Funkcja wyświetla i prosi o wybór opcji akcji dla towaru/usługi.

Parametry

invoice	wskaźnik na fakturę
ware	wskaźnik na wybrany towar/usługę

4.7 Dokumentacja pliku InvoiceBox.c

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

```
#include "Address.h"
#include "Person.h"
#include "Ware.h"
#include "Invoice.h"
#include "InvoiceBox.h"
#include "../functions/input.h"
#include "../functions/output.h"
```

Funkcje

- void addInvoice (struct Invoice **invoiceList, struct Invoice *invoice)
- void showInvoiceList (struct Invoice *invoiceList)
- struct Invoice * selectInvoice (struct Invoice *invoiceList)
- void deleteInvoiceFromList (struct Invoice **invoiceList, struct Invoice *invoice)
- int lengthInvoiceList (struct Invoice *invoiceList)
- void invoiceOptions (struct Invoice **invoiceList, struct Invoice *invoice)
- void invoiceEditOptions (struct Invoice *invoiceList, struct Invoice *invoice)
- void searchInvoicesByDate (struct Invoice **invoiceList, char date[])
- int searchInvoicesByPaid (struct Invoice **invoiceList)
- void deleteInvoiceList (struct Invoice **invoiceList)

4.7.1 Dokumentacja funkcji

4.7.1.1 addInvoice()

```
void addInvoice (
            struct Invoice ** invoiceList,
            struct Invoice * invoice )
```

Funkcja dodająca funkcję do jednokierunkowej listy funkcji.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
invoice	wskaźnik na fakturę która ma być dodana

Zwraca

4.7.1.2 deleteInvoiceFromList()

```
void deleteInvoiceFromList (
            struct Invoice ** invoiceList,
            struct Invoice * invoice )
```

Funkcja usuwa fakturę z listy faktur.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
invoice	wskaźnik na fakturę do usunięcia

4.7.1.3 deleteInvoiceList()

```
void deleteInvoiceList (
            struct Invoice ** invoiceList )
```

Funkcja usuwa listę faktur i zwalnia pamięć.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur

4.7.1.4 invoiceEditOptions()

```
void invoiceEditOptions (
            struct Invoice * invoiceList,
            struct Invoice * invoice )
```

Funkcja wyświetla i prosi o wybór opcji edycji dla faktury.

Parametry

invoiceList	wskaźnik na głowę listy faktur
invoice	wskaźnik na wybraną fakturę

4.7.1.5 invoiceOptions()

```
void invoiceOptions (
            struct Invoice ** invoiceList,
            struct Invoice * invoice )
```

Funkcja wyświetla i prosi o wybór opcji akcji dla faktury.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
invoice	wskaźnik na wybraną fakturę

4.7.1.6 lengthInvoiceList()

```
int lengthInvoiceList (
            struct Invoice * invoiceList )
```

Funkcja oblicza długość listy faktur.

Parametry

invoiceList	wskaźnik na głowę listy faktur

Zwraca

długość listy

4.7.1.7 searchInvoicesByDate()

```
void searchInvoicesByDate (
            struct Invoice ** invoiceList,
            char date[] )
```

Funkcja wyszukuje i wyświetla faktury na podstawie daty.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
date	data przekazana jako tablica znaków

4.7.1.8 searchInvoicesByPaid()

```
int searchInvoicesByPaid (
            struct Invoice ** invoiceList )
```

Funkcja wyszukuje i wyświetla faktury które nie są opłacone.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur

4.7.1.9 selectInvoice()

```
struct Invoice* selectInvoice (
            struct Invoice * invoiceList )
```

Funkcja prosi o numer faktury na podstawie wcześniej wyświetlonej listy.

Parametry

invoiceList	wskaźnik na głowę listy faktur

Zwraca

wskaźnik na wybraną fakturę

4.7.1.10 showInvoiceList()

```
void showInvoiceList (
            struct Invoice * invoiceList )
```

Funkcja wyświetla listę faktur.

Parametry

invoiceList	wskaźnik na głowę listy faktur

4.8 Dokumentacja pliku InvoiceBox.h

Funkcje

- void addInvoice (struct Invoice **invoiceList, struct Invoice *invoice)
- void showInvoiceList (struct Invoice *invoiceList)
- struct Invoice * selectInvoice (struct Invoice *invoiceList)
- void deleteInvoiceFromList (struct Invoice **invoiceList, struct Invoice *invoice)
- int lengthInvoiceList (struct Invoice *invoiceList)
- void invoiceOptions (struct Invoice **invoiceList, struct Invoice *invoice)
- void invoiceEditOptions (struct Invoice *invoiceList, struct Invoice *invoice)
- void searchInvoicesByDate (struct Invoice **invoiceList, char date[ ])
- int searchInvoicesByPaid (struct Invoice **invoiceList)
- void deleteInvoiceList (struct Invoice **invoiceList)

4.8.1 Dokumentacja funkcji

4.8.1.1 addInvoice()

```
void addInvoice (
            struct Invoice ** invoiceList,
            struct Invoice * invoice )
```

Funkcja dodająca funkcję do jednokierunkowej listy funkcji.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
invoice	wskaźnik na fakturę która ma być dodana

Zwraca

4.8.1.2 deleteInvoiceFromList()

```
void deleteInvoiceFromList (
            struct Invoice ** invoiceList,
            struct Invoice * invoice )
```

Funkcja usuwa fakturę z listy faktur.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
invoice	wskaźnik na fakturę do usunięcia

4.8.1.3 deleteInvoiceList()

```
void deleteInvoiceList (
            struct Invoice ** invoiceList )
```

Funkcja usuwa listę faktur i zwalnia pamięć.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur

4.8.1.4 invoiceEditOptions()

```
void invoiceEditOptions (
            struct Invoice * invoiceList,
            struct Invoice * invoice )
```

Funkcja wyświetla i prosi o wybór opcji edycji dla faktury.

Parametry

invoiceList	wskaźnik na głowę listy faktur
invoice	wskaźnik na wybraną fakturę

4.8.1.5 invoiceOptions()

```
void invoiceOptions (
            struct Invoice ** invoiceList,
            struct Invoice * invoice )
```

Funkcja wyświetla i prosi o wybór opcji akcji dla faktury.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
invoice	wskaźnik na wybraną fakturę

4.8.1.6 lengthInvoiceList()

```
int lengthInvoiceList (
            struct Invoice * invoiceList )
```

Funkcja oblicza długość listy faktur.

Parametry

invoiceList	wskaźnik na głowę listy faktur

Zwraca

długość listy

4.8.1.7 searchInvoicesByDate()

```
void searchInvoicesByDate (
            struct Invoice ** invoiceList,
            char date[] )
```

Funkcja wyszukuje i wyświetla faktury na podstawie daty.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur
date	data przekazana jako tablica znaków

4.8.1.8 searchInvoicesByPaid()

```
int searchInvoicesByPaid (
            struct Invoice ** invoiceList )
```

Funkcja wyszukuje i wyświetla faktury które nie są opłacone.

Parametry

invoiceList	wskaźnik na wskaźnik na głowę listy faktur

4.8.1.9 selectInvoice()

```
struct Invoice* selectInvoice (
            struct Invoice * invoiceList )
```

Funkcja prosi o numer faktury na podstawie wcześniej wyświetlonej listy.

Parametry

invoiceList	wskaźnik na głowę listy faktur

Zwraca

wskaźnik na wybraną fakturę

4.8.1.10 showInvoiceList()

```
void showInvoiceList (
            struct Invoice * invoiceList )
```

Funkcja wyświetla listę faktur.

Parametry

invoiceList	wskaźnik na głowę listy faktur

4.9 Dokumentacja pliku output.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "../structures/Address.h"
#include "../structures/Person.h"
#include "../structures/Ware.h"
#include "../structures/Invoice.h"
#include "output.h"
```

Funkcje

- void saveDataToFile (struct Invoice ∗invoiceList, const char ∗PATH, char ∗fileName)
- void filePrintPersonData (struct Person ∗person, FILE ∗fptr)
- void filePrintAddress (struct Address ∗address, FILE ∗fptr)
- void filePrintWare (struct Ware ∗ware, FILE ∗fptr)
- void filePrintSeparator (int n, char separator, FILE ∗fptr)
- void printSeparator (int n, char separator)
- void printMenu ()
- void printOptions ()

4.9.1 Dokumentacja funkcji

4.9.1.1 filePrintAddress()

```
void filePrintAddress (
            struct Address * address,
            FILE * fptr )
```

Funkcja zapisuje do pliku dane adresowe.

Parametry

address	wskaźnik na adres
fptr	wskaźnik na plik

4.9.1.2 filePrintPersonData()

```
void filePrintPersonData (
            struct Person * person,
            FILE * fptr )
```

Funkcja zapisuje do pliku dane osoby.

Parametry

person	wskaźnik na osobę
fptr	wskaźnik na plik

4.9.1.3 filePrintSeparator()

```
void filePrintSeparator (
            int n,
            char separator,
            FILE * fptr )
```

Funkcja zapisuje do pliku separator.

Parametry

n	długość separatora
separator	znak z jakiego ma być zbudowany separator
fptr	wskaźnik na plik

4.9.1.4 filePrintWare()

```
void filePrintWare (
            struct Ware * ware,
            FILE * fptr )
```

Funkcja zapisuje do pliku dane towaru/usługi.

Parametry

ware	wskaźnik na towar/usługę
fptr	wskaźnik na plik

4.9.1.5 printMenu()

```
void printMenu ( )
```

Funkcja wypisuje menu do konsoli.

4.9.1.6 printOptions()

```
void printOptions ( )
```

Funkcja wypisuje opcje do konsoli.

4.9.1.7 printSeparator()

```
void printSeparator (
            int n,
            char separator )
```

Funkcja wypisuje separator do konsoli.

Parametry

n	długość separatora
separator	znak z jakiego ma być zbudowany separator

4.9.1.8 saveDataToFile()

```
void saveDataToFile (
            struct Invoice * invoiceList,
            const char * PATH,
            char * fileName )
```

Funkcja zapisuje dane do pliku.

Parametry

invoiceList	wskaźnik na jednokierunkową listę faktur
PATH	ścieżka do folderu z plikiem do zapisu, gdy "" to zapis do folderu w którym jest program
fileName	nazwa pliku do zapisu (wymagane rozszerzenie - ".txt")

4.10 Dokumentacja pliku output.h

Funkcje

- void saveDataToFile (struct Invoice ∗invoiceList, const char ∗PATH, char ∗fileName)

- void filePrintPersonData (struct Person ∗person, FILE ∗fptr)
- void filePrintAddress (struct Address ∗address, FILE ∗fptr)
- void filePrintWare (struct Ware ∗ware, FILE ∗fptr)
- void filePrintSeparator (int n, char separator, FILE ∗fptr)
- void printSeparator (int n, char separator)
- void printMenu ()
- void printOptions ()

4.10.1 Dokumentacja funkcji

4.10.1.1 filePrintAddress()

```
void filePrintAddress (
            struct Address * address,
            FILE * fptr )
```

Funkcja zapisuje do pliku dane adresowe.

Parametry

address	wskaźnik na adres
fptr	wskaźnik na plik

4.10.1.2 filePrintPersonData()

```
void filePrintPersonData (
            struct Person * person,
            FILE * fptr )
```

Funkcja zapisuje do pliku dane osoby.

Parametry

person	wskaźnik na osobę
fptr	wskaźnik na plik

4.10.1.3 filePrintSeparator()

```
void filePrintSeparator (
            int n,
```

```
            char separator,
            FILE * fptr )
```

Funkcja zapisuje do pliku separator.

Parametry

n	długość separatora
separator	znak z jakiego ma być zbudowany separator
fptr	wskaźnik na plik

4.10.1.4 filePrintWare()

```
void filePrintWare (
            struct Ware * ware,
            FILE * fptr )
```

Funkcja zapisuje do pliku dane towaru/usługi.

Parametry

ware	wskaźnik na towar/usługę
fptr	wskaźnik na plik

4.10.1.5 printMenu()

```
void printMenu ( )
```

Funkcja wypisuje menu do konsoli.

4.10.1.6 printOptions()

```
void printOptions ( )
```

Funkcja wypisuje opcje do konsoli.

4.10.1.7 printSeparator()

```
void printSeparator (
            int n,
            char separator )
```

Funkcja wypisuje separator do konsoli.

Parametry

n	długość separatora
separator	znak z jakiego ma być zbudowany separator

4.10.1.8 saveDataToFile()

```
void saveDataToFile (
            struct Invoice * invoiceList,
            const char * PATH,
            char * fileName )
```

Funkcja zapisuje dane do pliku.

Parametry

invoiceList	wskaźnik na jednokierunkową listę faktur
PATH	ścieżka do folderu z plikiem do zapisu, gdy "" to zapis do folderu w którym jest program
fileName	nazwa pliku do zapisu (wymagane rozszerzenie - ".txt")

4.11 Dokumentacja pliku Person.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "Address.h"
#include "Person.h"
#include "../functions/utilities.h"
#include "../functions/input.h"
```

Funkcje

- struct Person ∗ createPerson ()
- void fillPerson (struct Person ∗person, struct Address ∗address, char companyName[ ], char name[ ], char surname[ ], char nip[ ], char accountNumber[ ])
- void deletePerson (struct Person ∗person)
- void readDataPerson (struct Person ∗person, int isSolder)
- void editPerson (struct Person ∗person, int isSolder)
- void showPersonsTogether (struct Person ∗solder, struct Person ∗buyer)
- void showPersonsInLine (struct Person ∗solder, struct Person ∗buyer)
- char ∗ getLinePerson (struct Person ∗person)
- int isCompany (struct Person ∗person)

4.11.1 Dokumentacja funkcji

4.11.1.1 createPerson()

```
struct Person* createPerson ( )
```

Konstruktor wypełnia strukturę osoby domyślnymi danymi i alokuje pamięć na nią.

Zwraca

wskaźnik na utworzoną strukturę osoby

4.11.1.2 deletePerson()

```
void deletePerson (
            struct Person * person )
```

Funkcja usuwa osobę wraz z jej adresem i zwalnia pamięć .

Parametry

person	wskaźnik na osobę

4.11.1.3 editPerson()

```
void editPerson (
            struct Person * person,
            int isSolder )
```

Funkcja prosi o i odczytuje nowe dane osobowe do edycji z konsoli oraz przypisuje je do osoby.

Parametry

person	wskaźnik na osobę do która będzie edytowana
isSolder	sprawdzenie czy osoba będzie sprzedawcą czy nabywcą

4.11.1.4 fillPerson()

```
void fillPerson (
```

```
            struct Person * person,
            struct Address * address,
            char companyName[],
            char name[],
            char surname[],
            char nip[],
            char accountNumber[] )
```

Funkcja wypełnia strukturę osoby danymi.

Parametry

person	wskaźnik na osobę
address	wskaźnik na adres
companyName	tablica znaków zawierająca nazwę firmy
name	tablica znaków zawierająca imię
surname	tablica znaków zawierająca nazwisko
nip	tablica znaków zawierająca nip
accountNumber	tablica znaków zawierająca numer konta

4.11.1.5 getLinePerson()

```
char* getLinePerson (
            struct Person * person )
```

Funkcja łączy nazwę firmy, imię i nazwisko osoby w jedną linie.

Parametry

person	wskaźnik na osobę

Zwraca

string zawierający połączone wszystkie dane z " " pomiędzy każdą

4.11.1.6 isCompany()

```
int isCompany (
            struct Person * person )
```

Funkcja sprawdza czy przekazana osoba posiada firmę czy jest osobą prywatną.

Parametry

person	wskaźnik na osobę

Zwraca

1 jeśli posiada firmę, 0 w przeciwnym wypadku

4.11.1.7 readDataPerson()

```
void readDataPerson (
            struct Person * person,
            int isSolder )
```

Funkcja prosi o i odczytuje dane osobowe z konsoli oraz przypisuje je do osoby.

Parametry

person	wskaźnik na osobę do której będą przekazane dane
isSolder	sprawdzenie czy osoba będzie sprzedawcą czy nabywcą

4.11.1.8 showPersonsInLine()

```
void showPersonsInLine (
            struct Person * solder,
            struct Person * buyer )
```

Funkcja wyświetla dane sprzedawcy i nabywcy obok siebie.

Parametry

solder	wskaźnik na osobę sprzedawcy
buyer	wskaźnik na osobę nabywcy

4.11.1.9 showPersonsTogether()

```
void showPersonsTogether (
            struct Person * solder,
            struct Person * buyer )
```

Funkcja wyświetla podpisane dane sprzedawcy i nabywcy obok siebie.

Parametry

solder	wskaźnik na osobę sprzedawcy
buyer	wskaźnik na osobę nabywcy

4.12 Dokumentacja pliku Person.h

Struktury danych

- struct Person

Funkcje

- struct Person * createPerson ()
- void fillPerson (struct Person *person, struct Address *address, char companyName[ ], char name[ ], char surname[ ], char nip[ ], char accountNumber[ ])
- void deletePerson (struct Person *person)
- void readDataPerson (struct Person *person, int isSolder)
- void editPerson (struct Person *person, int isSolder)
- void showPersonsTogether (struct Person *solder, struct Person *buyer)
- void showPersonsInLine (struct Person *solder, struct Person *buyer)
- char * getLinePerson (struct Person *person)
- int isCompany (struct Person *person)

4.12.1 Dokumentacja funkcji

4.12.1.1 createPerson()

```
struct Person* createPerson ( )
```

Konstruktor wypełnia strukturę osoby domyślnymi danymi i alokuje pamięć na nią.

Zwraca

wskaźnik na utworzoną strukturę osoby

4.12.1.2 deletePerson()

```
void deletePerson (
            struct Person * person )
```

Funkcja usuwa osobę wraz z jej adresem i zwalnia pamięć .

Parametry

person	wskaźnik na osobę

4.12.1.3 editPerson()

```
void editPerson (
            struct Person * person,
            int isSolder )
```

Funkcja prosi o i odczytuje nowe dane osobowe do edycji z konsoli oraz przypisuje je do osoby.

Parametry

person	wskaźnik na osobę do która będzie edytowana
isSolder	sprawdzenie czy osoba będzie sprzedawcą czy nabywcą

4.12.1.4 fillPerson()

```
void fillPerson (
            struct Person * person,
            struct Address * address,
            char companyName[],
            char name[],
            char surname[],
            char nip[],
            char accountNumber[] )
```

Funkcja wypełnia strukturę osoby danymi.

Parametry

person	wskaźnik na osobę
address	wskaźnik na adres
companyName	tablica znaków zawierająca nazwę firmy
name	tablica znaków zawierająca imię
surname	tablica znaków zawierająca nazwisko
nip	tablica znaków zawierająca nip
accountNumber	tablica znaków zawierająca numer konta

4.12.1.5 getLinePerson()

```
char* getLinePerson (
            struct Person * person )
```

Funkcja łączy nazwę firmy, imię i nazwisko osoby w jedną linie.

Parametry

person	wskaźnik na osobę

Zwraca

string zawierający połączone wszystkie dane z " " pomiędzy każdą

4.12.1.6 isCompany()

```
int isCompany (
            struct Person * person )
```

Funkcja sprawdza czy przekazana osoba posiada firmę czy jest osobą prywatną.

Parametry

person	wskaźnik na osobę

Zwraca

1 jeśli posiada firmę, 0 w przeciwnym wypadku

4.12.1.7 readDataPerson()

```
void readDataPerson (
            struct Person * person,
            int isSolder )
```

Funkcja prosi o i odczytuje dane osobowe z konsoli oraz przypisuje je do osoby.

Parametry

person	wskaźnik na osobę do której będą przekazane dane
isSolder	sprawdzenie czy osoba będzie sprzedawcą czy nabywcą

4.12.1.8 showPersonsInLine()

```
void showPersonsInLine (
            struct Person * solder,
            struct Person * buyer )
```

Funkcja wyświetla dane sprzedawcy i nabywcy obok siebie.

Parametry

solder	wskaźnik na osobę sprzedawcy
buyer	wskaźnik na osobę nabywcy

4.12.1.9 showPersonsTogether()

```
void showPersonsTogether (
            struct Person * solder,
            struct Person * buyer )
```

Funkcja wyświetla podpisane dane sprzedawcy i nabywcy obok siebie.

Parametry

solder	wskaźnik na osobę sprzedawcy
buyer	wskaźnik na osobę nabywcy

4.13 Dokumentacja pliku utilities.c

```
#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <string.h>
#include "utilities.h"
```

Funkcje

- int checkString (const char ∗string, char ∗value)
- void cutString (char ∗string, int length)
- char ∗ getCurrentDate (char dateFormat[ ])
- char ∗ getFutureDate (char dateFormat[ ], int weeks)
- int isNegative (char string[ ])
- char ∗ concatenationStrings (char a[ ], char b[ ])
- int isValidDateFormat (const char ∗date)
- int isValidDatePart (char ∗date, int start, int length, int min, int max)
- char ∗ formatAccountNumber (const char ∗accountNumber)

4.13.1 Dokumentacja funkcji

4.13.1.1 checkString()

```
int checkString (
            const char * string,
            char * value )
```

Funkcja sprawdza czy początek stringa zawiera przekazaną wartość.

Parametry

string	string w którego początek będzie sprawdzany
value	wartość zapisana w stringu jako podstawa do sprawdzenia

4.13.1.2 concatenationStrings()

```
char* concatenationStrings (
            char a[],
            char b[] )
```

Funkcja łączy przekazane stringi i oddziela je " " między sobą.

Parametry

a	string do połączenia
b	string do połączenia

Zwraca

string połączone wyrazy oddzielone " "

4.13.1.3 cutString()

```
void cutString (
            char * string,
            int length )
```

Funkcja przycina przekazany string na podaną długość i daje na końcu znak końca stringa ("\0").

Parametry

string	string do przycięcia
length	długość do jakiej ma być przycięty string

4.13.1.4 formatAccountNumber()

```
char* formatAccountNumber (
            const char * accountNumber )
```

Funkcja formatuje przekazany numer konta(" " po grupie 4 cyfr).

Parametry

accountNumber	numer konta do formatowania zapisany jako string

Zwraca

sformatowany numer konta jako string

4.13.1.5 getCurrentDate()

```
char* getCurrentDate (
            char dateFormat[] )
```

Funkcja pobiera aktualną datę z systemu.

Parametry

dateFormat	format w jakim ma być zwrócona data

Zwraca

aktualna data jako łańcuch znaków

4.13.1.6 getFutureDate()

```
char* getFutureDate (
            char dateFormat[],
            int weeks )
```

Funkcja do aktualnej daty dodaje przekazaną ilość tygodni.

Parametry

dateFormat	format w jakim ma być zwrócona data
weeks	ilość tygodni jaka ma być dodana do daty

Zwraca

obliczona data jako łańcuch znaków

4.13.1.7 isNegative()

```
int isNegative (
            char string[] )
```

Funkcja sprawdza czy przekazany string jest liczbą ujemną.

Parametry

string	tablica znaków do sprawdzenia

Zwraca

1 jeśli liczba ujemna, 0 w przeciwnym razie

4.13.1.8 isValidDateFormat()

```
int isValidDateFormat (
            const char * date )
```

Funkcja sprawdza czy przekazana data jest w odpowiednim formacie.

Parametry

date	data do sprawdzenia zapisana jako string

Zwraca

1 jeśli format daty jest poprawny, 0 w przeciwnym razie

4.13.1.9 isValidDatePart()

```
int isValidDatePart (
            char * date,
            int start,
            int length,
            int min,
            int max )
```

Funkcja sprawdza czy część przekazanej daty jest poprawna.

Parametry

date	data do sprawdzenia zapisana jako string
start	początek przedziału do sprawdzenia
length	długość przedziału do sprawdzenia
min	wartość minimalna jaka może być w danym przedziale
max	wartość maksymalna jaka może być w danym przedziale

Zwraca

1 jeśli wartość w danej części daty jest poprawna, 0 w przeciwnym razie

4.14 Dokumentacja pliku utilities.h

Funkcje

- int checkString (const char ∗string, char ∗value)
- void cutString (char ∗string, int length)
- char ∗ getCurrentDate (char dateFormat[ ])
- char ∗ getFutureDate (char dateFormat[ ], int weeks)
- int isNegative (char string[ ])
- char ∗ concatenationStrings (char a[ ], char b[ ])
- int isValidDateFormat (const char ∗date)
- int isValidDatePart (char ∗date, int start, int length, int min, int max)
- char ∗ formatAccountNumber (const char ∗accountNumber)

4.14.1 Dokumentacja funkcji

4.14.1.1 checkString()

```
int checkString (
            const char * string,
            char * value )
```

Funkcja sprawdza czy początek stringa zawiera przekazaną wartość.

Parametry

string	string w którego początek będzie sprawdzany
value	wartość zapisana w stringu jako podstawa do sprawdzenia

4.14.1.2 concatenationStrings()

```
char* concatenationStrings (
            char a[],
            char b[] )
```

Funkcja łączy przekazane stringi i oddziela je " " między sobą.

Parametry

a	string do połączenia
b	string do połączenia

Zwraca

string połączone wyrazy oddzielone " "

4.14.1.3 cutString()

```
void cutString (
            char * string,
            int length )
```

Funkcja przycina przekazany string na podaną długość i daje na końcu znak końca stringa ("\0").

Parametry

string	string do przycięcia
length	długość do jakiej ma być przycięty string

4.14.1.4 formatAccountNumber()

```
char* formatAccountNumber (
            const char * accountNumber )
```

Funkcja formatuje przekazany numer konta(" " po grupie 4 cyfr).

Parametry

accountNumber	numer konta do formatowania zapisany jako string

Zwraca

sformatowany numer konta jako string

4.14.1.5 getCurrentDate()

```
char* getCurrentDate (
            char dateFormat[] )
```

Funkcja pobiera aktualną datę z systemu.

Parametry

dateFormat	format w jakim ma być zwrócona data

Zwraca

aktualna data jako łańcuch znaków

4.14.1.6 getFutureDate()

```
char* getFutureDate (
            char dateFormat[],
            int weeks )
```

Funkcja do aktualnej daty dodaje przekazaną ilość tygodni.

Parametry

dateFormat	format w jakim ma być zwrócona data
weeks	ilość tygodni jaka ma być dodana do daty

Zwraca

obliczona data jako łańcuch znaków

4.14.1.7 isNegative()

```
int isNegative (
            char string[] )
```

Funkcja sprawdza czy przekazany string jest liczbą ujemną.

Parametry

string	tablica znaków do sprawdzenia

Zwraca

1 jeśli liczba ujemna, 0 w przeciwnym razie

4.14.1.8 isValidDateFormat()

```
int isValidDateFormat (
            const char * date )
```

Funkcja sprawdza czy przekazana data jest w odpowiednim formacie.

Parametry

date	data do sprawdzenia zapisana jako string

Zwraca

1 jeśli format daty jest poprawny, 0 w przeciwnym razie

4.14.1.9 isValidDatePart()

```
int isValidDatePart (
            char * date,
            int start,
            int length,
            int min,
            int max )
```

Funkcja sprawdza czy część przekazanej daty jest poprawna.

Parametry

date	data do sprawdzenia zapisana jako string
start	początek przedziału do sprawdzenia
length	długość przedziału do sprawdzenia
min	wartość minimalna jaka może być w danym przedziale
max	wartość maksymalna jaka może być w danym przedziale

Zwraca

1 jeśli wartość w danej części daty jest poprawna, 0 w przeciwnym razie

4.15 Dokumentacja pliku Ware.c

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "Ware.h"
#include "../functions/utilities.h"
#include "../functions/input.h"
```

Funkcje

- struct Ware * createWare ()
- void fillWare (struct Ware *ware, char name[ ], char amount[ ], char netPrice[ ], char tax[ ])
- void showWare (struct Ware *ware, int i)
- void readDataWare (struct Ware *ware)
- void editWare (struct Ware *ware)
- void calculateValuesWare (struct Ware *ware)

4.15.1 Dokumentacja funkcji

4.15.1.1 calculateValuesWare()

```
void calculateValuesWare (
            struct Ware * ware )
```

Funkcja wyliczająca i przypisująca do towaru/usługi wartości netto brutto i VAT.

Parametry

ware	wskaźnik na towar/usługę

4.15.1.2 createWare()

```
struct Ware* createWare ( )
```

Konstruktor alokuje pamięć na strukturę.

Zwraca

wskaźnik na utworzoną strukturę towaru/usługi

4.15.1.3 editWare()

```
void editWare (
            struct Ware * ware )
```

Funkcja prosi o i odczytuje nowe dane towaru/usługi do edycji z konsoli oraz przypisuje je do towaru/usługi.

Parametry

ware	wskaźnik na towar/usługę do która będzie edytowana

4.15.1.4 fillWare()

```
void fillWare (
            struct Ware * ware,
```

```
            char name[],
            char amount[],
            char netPrice[],
            char tax[] )
```

Funkcja wypełnia strukturę towaru/usługi danymi.

Parametry

ware	wskaźnik na towar/usługę
name	tablica znaków zawierająca nazwę
amount	tablica znaków zawierająca ilość
netPrice	tablica znaków zawierająca cenę netto
tax	tablica znaków zawierająca podatek

4.15.1.5 readDataWare()

```
void readDataWare (
            struct Ware * ware )
```

Funkcja prosi o i odczytuje dane towaru/usługi z konsoli oraz przypisuje je do towaru/usługi.

Parametry

ware	wskaźnik na towar/usługę do której będą przekazane dane

4.15.1.6 showWare()

```
void showWare (
            struct Ware * ware,
            int i )
```

Funkcja wyświetla towar/usługę.

Parametry

ware	wskaźnik na towar/usługę
i	numer towaru usługi na liście

4.16 Dokumentacja pliku Ware.h

Struktury danych

- struct Ware

Funkcje

- struct Ware ∗ createWare ()
- void fillWare (struct Ware ∗ware, char name[ ], char amount[ ], char netPrice[ ], char tax[ ])
- void showWare (struct Ware ∗ware, int i)
- void readDataWare (struct Ware ∗ware)
- void editWare (struct Ware ∗ware)
- void calculateValuesWare (struct Ware ∗ware)

4.16.1 Dokumentacja funkcji

4.16.1.1 calculateValuesWare()

```
void calculateValuesWare (
            struct Ware * ware )
```

Funkcja wyliczająca i przypisująca do towaru/usługi wartości netto brutto i VAT.

Parametry

ware	wskaźnik na towar/usługę

4.16.1.2 createWare()

```
struct Ware* createWare ( )
```

Konstruktor alokuje pamięć na strukturę.

Zwraca

wskaźnik na utworzoną strukturę towaru/usługi

4.16.1.3 editWare()

```
void editWare (
            struct Ware * ware )
```

Funkcja prosi o i odczytuje nowe dane towaru/usługi do edycji z konsoli oraz przypisuje je do towaru/usługi.

Parametry

ware	wskaźnik na towar/usługę do która będzie edytowana

4.16.1.4 fillWare()

```
void fillWare (
            struct Ware * ware,
            char name[],
            char amount[],
            char netPrice[],
            char tax[] )
```

Funkcja wypełnia strukturę towaru/usługi danymi.

Parametry

ware	wskaźnik na towar/usługę
name	tablica znaków zawierająca nazwę
amount	tablica znaków zawierająca ilość
netPrice	tablica znaków zawierająca cenę netto
tax	tablica znaków zawierająca podatek

4.16.1.5 readDataWare()

```
void readDataWare (
            struct Ware * ware )
```

Funkcja prosi o i odczytuje dane towaru/usługi z konsoli oraz przypisuje je do towaru/usługi.

Parametry

ware	wskaźnik na towar/usługę do której będą przekazane dane

4.16.1.6 showWare()

```
void showWare (
            struct Ware * ware,
            int i )
```

Funkcja wyświetla towar/usługę.

Parametry

ware	wskaźnik na towar/usługę
i	numer towaru usługi na liście