Pet Shop

Generated by Doxygen 1.8.16

1 Hierarchical Index 1

1.1 Class Hierarchy . . . 1

2 Class Index 3

2.1 Class List . . . 3

3 Class Documentation 5

3.1 Animal Class Reference . . . 5

3.1.1 Member Function Documentation . . . 6

3.1.1.1 animalToOStream() . . . 6

3.1.1.2 decideIfDiscount() . . . 6

3.1.1.3 operator=() . . . 6

3.1.2 Friends And Related Function Documentation . . . 7

3.1.2.1 operator<< . . . 7

3.1.2.2 operator>> . . . 7

3.2 CashRegister Class Reference . . . 7

3.2.1 Member Function Documentation . . . 8

3.2.1.1 cashRegisterToOStream() . . . 8

3.2.1.2 collectCash() . . . 8

3.2.1.3 operator++() . . . 8

3.2.1.4 operator[]() . . . 8

3.2.1.5 spendCash() . . . 9

3.2.2 Friends And Related Function Documentation . . . 9

3.2.2.1 operator<< . . . 9

3.2.2.2 operator>> . . . 9

3.3 ListMB< T > Class Template Reference . . . 10

3.3.1 Detailed Description . . . 10

3.3.2 Constructor & Destructor Documentation . . . 10

3.3.2.1 ListMB() . . . 11

3.3.3 Member Function Documentation . . . 11

3.3.3.1 operator=() . . . 11

3.3.3.2 popData() . . . 11

3.3.3.3 pushData() . . . 11

3.4 NodeMB< T > Class Template Reference . . . 12

3.4.1 Detailed Description . . . 12

3.5 PetFood Class Reference . . . 12

3.5.1 Member Function Documentation . . . 13

3.5.1.1 decideIfDiscount() . . . 13

3.5.2 Friends And Related Function Documentation . . . 13

3.5.2.1 operator<< . . . 13

3.5.2.2 operator>> . . . 14

3.6 PetShop Class Reference . . . 14

3.6.1 Detailed Description . . . 15

3.6.2 Constructor & Destructor Documentation . . . 15
3.6.2.1 PetShop() . . . 15
3.6.3 Member Function Documentation . . . 15
3.6.3.1 addAnimal() . . . 16
3.6.3.2 addPetFood() . . . 16
3.6.3.3 addPetToy() . . . 16
3.6.3.4 decideIfDiscount() . . . 16
3.6.3.5 operator double() . . . 16
3.6.3.6 operator+() [1/2] . . . 17
3.6.3.7 operator+() [2/2] . . . 17
3.6.3.8 operator+=() [1/3] . . . 17
3.6.3.9 operator+=() [2/3] . . . 17
3.6.3.10 operator+=() [3/3] . . . 18
3.6.3.11 returnAnimal() . . . 18
3.6.3.12 returnPetFood() . . . 18
3.6.3.13 returnPetToy() . . . 19
3.6.3.14 updateAdress() . . . 19
3.6.3.15 updateName() . . . 19
3.6.3.16 updateShopDetails() . . . 19
3.6.3.17 updateTaxNumber() . . . 19
3.6.3.18 userAddAnimal() . . . 20
3.6.3.19 userAddPetFood() . . . 20
3.6.3.20 userAddPetToy() . . . 20
3.6.3.21 userChooseAnimal() . . . 20
3.6.3.22 userChoosePetFood() . . . 20
3.6.3.23 userChoosePetToy() . . . 21
3.6.3.24 userFeedAnimal() . . . 21
3.6.3.25 userUpdateAdress() . . . 21
3.6.3.26 userUpdateName() . . . 21
3.6.3.27 userUpdateTaxNumber() . . . 21
3.6.4 Friends And Related Function Documentation . . . 21
3.6.4.1 operator<< . . . 22
3.6.4.2 operator>> . . . 22
3.7 PetToy Class Reference . . . 22
3.7.1 Member Function Documentation . . . 23
3.7.1.1 changeStock() . . . 23
3.7.1.2 coutReceiptFormat() . . . 23
3.7.1.3 decideIfDiscount() . . . 24
3.7.1.4 userUpdateStockNumber() . . . 24
3.7.2 Friends And Related Function Documentation . . . 24
3.7.2.1 operator<< . . . 24
3.7.2.2 operator>> . . . 24

3.8 Product Class Reference 25
3.8.1 Member Function Documentation 26
3.8.1.1 coutReceiptFormat() 26
3.8.1.2 decideIfDiscount() 27
3.8.1.3 returnProductType() 27
3.8.1.4 returnSpecies() 27
3.8.1.5 userUpdateCost() 28
3.8.1.6 userUpdatePrice() 28
3.8.1.7 userUpdateSpecies() 28
3.8.2 Friends And Related Function Documentation 28
3.8.2.1 operator<< 28
3.9 Receipt Class Reference 29
3.9.1 Member Function Documentation 29
3.9.1.1 addProduct() 29
3.9.2 Friends And Related Function Documentation 29
3.9.2.1 operator<< 29

Index **31**

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

CashRegister 7
ListMB< T > 10
NodeMB< T > 12
PetShop 14
Product 25
  Animal 5
  PetToy 22
    PetFood 12
Receipt 29

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

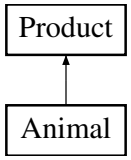
Animal 5
CashRegister 7
ListMB< T >
    LISTA 10
NodeMB< T >
    WEZEŁ 12
PetFood 12
PetShop 14
PetToy 22
Product 25
Receipt 29

Chapter 3

Class Documentation

3.1 Animal Class Reference

Inheritance diagram for Animal:



Public Types

- enum Sex { **male** = 0, **female** = 1 }

  Definicja typu wyliczeniowego płci.

Public Member Functions

- Animal (Species=Product::chomik, Sex=male, int=1, double=10.0, double=10.0)

  Konstruktor zwierzęcia.
- ∼Animal ()

  Dekonstruktor zwierzęcia.
- int returnAge ()

  Metoda zwracająca wiek zwierzęcia.
- void operator= (PetFood &)
- std::string returnSexString ()

  Metoda, która zwraca płeć zwierzęcia (typ enum) w postaci stringa.
- virtual bool decideIfDiscount ()
- virtual std::ostream & coutReceiptFormat (std::ostream &os)

  Metoda wyświetlająca zwierze w formacie paragonowym.
- std::ostream & animalToOStream (std::ostream &)
- void userUpdateSex ()

  Metoda pozwalająca użytkownikowi zaktualizować płeć zwierzęcia.
- void userUpdateAge ()

  Metoda pozwalająca użytkownikowi zaktualizować wiek zwierzęcia.
- virtual std::ostream & productToOstreamEncyclopedia (std::ostream &os)

  Metoda wirtualna przekazująca obiekt w formacie "do wyświetlenia" do obiektu klasy ostream.

Static Public Member Functions

- static int returnHowMany ()

  Statyczna metoda, która zwraca ile obiektów jest aktualnie stworzonych.

Friends

- std::ostream & operator<< (std::ostream &, Animal &)
- std::istream & operator>> (std::istream &is, Animal &pt)

Additional Inherited Members

3.1.1 Member Function Documentation

3.1.1.1 animalToOStream()

```
std::ostream & Animal::animalToOStream (
            std::ostream & os )
```

Metoda przekazująca zwierzę do obiektu klasy ostream

Format zapisu do pliku (data ostatniego karmienia wyrażona w sekundach od ...)

3.1.1.2 decideIfDiscount()

```
bool Animal::decideIfDiscount ( )  [virtual]
```

Metoda wirtualna decydująca o przecenie

Jeśli zwierze jest starsze niż 5 lat, jego marża zostaje zmniejszona o 50%

Implements Product.

3.1.1.3 operator=()

```
void Animal::operator= (
            PetFood & pf )
```

Przeciążony operator =

Karmi (jeśli to dozwolone) zwierze przyrównanym jedzeniem

Przy pomyślnym nakarmieniu, następuje aktualizacja daty ostatniego karmienia

3.1.2 Friends And Related Function Documentation

3.1.2.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            Animal & anim )  [friend]
```

Przeciążony operator <<

Pozwala wyświetlić informacje o zwierzęciu (przekazać go do obiektu klasy ostream)

Format przekazywania: interfejs konsolowy (data wyświetlana w formacie daty)

3.1.2.2 operator>>

```
std::istream& operator>> (
            std::istream & is,
            Animal & pt )  [friend]
```

Przeciążony operator >>

Wczytuje zwierzę zgodnie ze schematem zapisywania w pliku

The documentation for this class was generated from the following files:

- Animal.hpp
- Animal.cpp

3.2 CashRegister Class Reference

Public Member Functions

- CashRegister (double=1000.0)
  Konstruktor kasy fiskalnej.
- ~CashRegister ()
  Dekonstruktor kasy fiskalnej.
- double returnCash ()
  Metoda zwracająca stan gotówki w kasie.
- int returnLength ()
  Metoda zwracająca ilość paragonów w historii (długość wektora)
- bool spendCash (double n)
- void collectCash (double c)
- void operator++ ()
- Receipt & operator[ ] (int)
- void productPushBack (Product *)
  Dodaje produkt na koniec wektora produktów ostatniego paragonu.
- std::ostream & cashRegisterToOStream (std::ostream &)

Friends

- std::ostream & operator<< (std::ostream &, CashRegister &)
- std::istream & operator>> (std::istream &is, CashRegister &cr)

3.2.1 Member Function Documentation

3.2.1.1 cashRegisterToOStream()

```
std::ostream & CashRegister::cashRegisterToOStream (
            std::ostream & os )
```

Metoda przekazująca kasę fiskalną do obiektu klasy ostream w formacie zapisu do pliku

Przekazuje informację o ilości gotówki oraz zapisuje nowoutworzone paragony w historii paragonów

3.2.1.2 collectCash()

```
void CashRegister::collectCash (
            double c )
```

Metoda dodająca do kasy c pieniędzy (wykorzystywana przy dodawaniu do paragonu)

Parameters

c	- ilość pieniędzy, którą należy dodać do kasy fiskalnej

3.2.1.3 operator++()

```
void CashRegister::operator++ ( )
```

Przeciążony operator ++

Dodaje nowy paragon na koniec wektora

3.2.1.4 operator[]()

```
Receipt & CashRegister::operator[] (
            int index )
```

Przeciążony operator indeksowania []

Zwraca referencje na i-ty paragon w historii paragonów

3.2.1.5 spendCash()

```
bool CashRegister::spendCash (
            double n )
```

Metoda pozwalająca "wydać" pieniądze

Odejmuje z kasy n pieniędzy i zwraca 1 jeśli się udało (było wystarczająco dużo) lub 0 jeśli nie starczyło

Parameters

n	- kwota do wydania z kasy

Returns

true - było wystarczająco dużo gotówki; false - nie starczyło gotówki

3.2.2 Friends And Related Function Documentation

3.2.2.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            CashRegister & cr )  [friend]
```

Przeciążony operator <<

Pozwala przekazać do obiektu klasy ostream informacje o obiekcie typu cash register

Format wyświetlania: interfejs konsolowy

3.2.2.2 operator>>

```
std::istream& operator>> (
            std::istream & is,
            CashRegister & cr )  [friend]
```

Przeciążony operator >>

Czyli metoda wczytująca gotówkę w kasie fiskalnej z pliku

The documentation for this class was generated from the following files:

- CashRegister.hpp
- CashRegister.cpp

3.3 ListMB< T > Class Template Reference

LISTA.

`#include <ListMB.h>`

Public Member Functions

- ListMB ()
  konstruktor
- ListMB (ListMB< T > &lArg)
  konstruktor kopiujący
- ~ListMB ()
  dekonstruktor
- int getLength ()
  zwraca długość listy
- std::chrono::milliseconds & getTimeOfInit ()
  zwraca czas inicjalizacji w milisekundach
- void pushBack (T pb)
  pushBack
- void pushFront (T pb)
  pushFront
- void pushData (T pb, int n)
- void popBack ()
  usuwa ostatni element
- void popFront ()
  usuwa pierwszy element
- void popData (int n)
- void coutLast ()
  wyświetla ostatni
- NodeMB< T > & operator[ ] (int n)
  operator indeksowania
- ListMB< T > & operator= (ListMB< T > &list2)
  operator przypisania

3.3.1 Detailed Description

template<typename T>
class ListMB< T >

LISTA.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 ListMB()

```
template<typename T >
ListMB< T >::ListMB (
            ListMB< T > & lArg )
```

konstruktor kopiujący

dodaje węzły z odpowiednimi wartościami

3.3.3 Member Function Documentation

3.3.3.1 operator=()

```
template<typename T >
ListMB< T > & ListMB< T >::operator= (
            ListMB< T > & list2 )
```

operator przypisania

zeruje listę pierwszą

dodaje węzły z odpowiednimi wartościami

3.3.3.2 popData()

```
template<typename T >
void ListMB< T >::popData (
            int n )
```

pozwala usunąć daną z listy na indeksie n (następne przesuwa na indeks o 1 mniejszy) jeśli n jest większy równy długości listy, to usunięty zostaje ostatni element z listy jeśli n <= 0 to usunięty zostaje pierwszy element listy

3.3.3.3 pushData()

```
template<typename T >
void ListMB< T >::pushData (
            T pb,
            int n )
```

pozwala dodać daną do listy na indeks n (następne przesuwa na indeks o 1 większy) jeśli n jest większy równy długości listy, to dana pb zostaje dodana na koniec listy jeśli n jest mniejszy od zera, dana pb zostaje dodana na początek

The documentation for this class was generated from the following file:

- ListMB.h

3.4 NodeMB< T > Class Template Reference

WEZEŁ

`#include <ListMB.h>`

Public Member Functions

- NodeMB (std::chrono::milliseconds refTime, T nodeDataArg)
  konstruktor
- NodeMB< T > *& getNext ()
  zwraca referencje na next
- T & getNodeData ()
  zwraca referencje na dane
- std::chrono::milliseconds & getTimeElapsed ()
  zwraca czas inicjalizacji w milisekundach

3.4.1 Detailed Description

template<typename T>
class NodeMB< T >

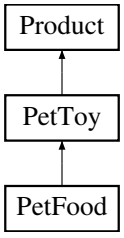
WEZEŁ

The documentation for this class was generated from the following file:

- ListMB.h

3.5 PetFood Class Reference

Inheritance diagram for PetFood:



Public Member Functions

- PetFood (int minAgeArg=0, int maxAgeArg=10, double priceArg=10.0, double costArg=10.0, int stock↩ NumberArg=1, Species speciesArg=Product::chomik, std::string manufacturerArg="")

  Konstruktor pokarmy.
- ∼PetFood ()

  Dekonstruktor pokarmu.
- int returnMaxAge ()

  Zwraca maksymalny wiek zwierzęcia odbiorcy.
- virtual bool decideIfDiscount ()
- void userUpdateMaxAge ()

  Metoda pozwalająca użytkownikowi zaktualizować wiek maksymalny.
- virtual std::ostream & productToOstreamEncyclopedia (std::ostream &os)

  Metoda wirtualna przekazująca obiekt w formacie "do wyświetlenia" do obiektu klasy ostream.

Friends

- std::ostream & operator<< (std::ostream &, PetFood &)
- std::istream & operator>> (std::istream &is, PetFood &pf)

Additional Inherited Members

3.5.1 Member Function Documentation

3.5.1.1 decideIfDiscount()

```
bool PetFood::decideIfDiscount ( )  [virtual]
```

Metoda wirtualna decydująca o promocji produktu

Jeśli pokarmu jest dużo (>=20) i jest przeznaczony dla wąskiego grona zwierząt (maxAge - minAge <= 2) zmniejsza marżę o 50%

Reimplemented from PetToy.

3.5.2 Friends And Related Function Documentation

3.5.2.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            PetFood & pf )  [friend]
```

Przeciążony operator <<

Pozwala wyświetlić informację o pokarmie (przekazuje obiekt do obiektu klasy ostream)

Format wyświetlania zgodny z zapisem do pliku

3.5.2.2 operator>>

```
std::istream& operator>> (
            std::istream & is,
            PetFood & pf )  [friend]
```

Przeciążony operator >>

Pozwala wczytać pokarm zgodnie ze schematem zapisywania w pliku

The documentation for this class was generated from the following files:

- PetFood.hpp
- PetFood.cpp

3.6 PetShop Class Reference

```
#include <PetShop.hpp>
```

Public Member Functions

- PetShop (std::string="Sklep zoologiczny", std::string="Sklepowa 1", std::string="0000000000")
- ~PetShop ()

  Dekonstruktor sklepu.
- CashRegister & returnCashRegister ()

  Zwraca referencje kasy fiskalnej, która jest podobiektem obiektu klasy PetShop.
- void presentInfo ()

  Prezentuje podstawowe informacje o sklepie (cout adres, nazwa, nip)
- void updateShopDetails ()
- std::string userUpdateName ()
- std::string userUpdateAdress ()
- std::string userUpdateTaxNumber ()
- void updateName (std::string newName)
- void updateAdress (std::string newAdress)
- void updateTaxNumber (std::string newTaxNumber)
- void operator+ (PetFood &pf)
- void operator+ (PetToy &pt)
- operator double ()
- void operator+= (Animal *anim)
- void operator+= (PetFood *pf)
- void operator+= (PetToy *pt)
- void addAnimal (Animal::Species spArg, Animal::Sex sexArg, int ageArg, double priceArg, double costArg)
- void addPetFood (int minAgeArg, int maxAgeArg, double priceArg, double costArg, int stockNumberArg, PetFood::Species animalsTypeArg, std::string manufacturerArg)
- void addPetToy (int minAgeArg, double priceArg, double costArg, int stockNumberArg, PetFood::Species animalsTypeArg, std::string manufacturerArg)
- PetFood * returnPetFood (int n)
- Animal * returnAnimal (int n)
- PetToy * returnPetToy (int n)
- std::ostream & petShopToOStream (std::ostream &)

  Metoda przekazująca sklep do obiektu klasy ostream w formacie zapisu do pliku.

- int returnAnimalVectorLength ()

  Metoda zwracająca ilość zwierząt w sklepie.
- int returnPetToyVectorLength ()

  Metoda zwracająca ilość zabawek w sklepie.
- int returnPetFoodVectorLength ()

  Metoda zwracająca ilość pokarmów w sklepie.
- void decideIfDiscount ()
- void userAddAnimal ()
- void userAddPetToy ()
- void userAddPetFood ()
- int userChooseAnimal ()
- int userChoosePetToy ()
- int userChoosePetFood ()
- void userFeedAnimal ()

Static Public Member Functions

- static int returnHowMany ()

  Statyczna metoda, która zwraca liczbę zainicjalizowanych obiektów klasy PetShop.

Friends

- std::ostream & operator<< (std::ostream &, PetShop &)
- std::istream & operator>> (std::istream &is, PetShop &ps)

3.6.1 Detailed Description

PetShop.hpp PetShop

Created by Marcin Badach on 31/10/2019. Copyright © 2019 Marcin Badach. All rights reserved.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 PetShop()

```
PetShop::PetShop (
            std::string nameArg = "Sklep zoologiczny",
            std::string adressArg = "Sklepowa 1",
            std::string taxNumberArg = "0000000000" )
```

Konstruktor sklepu, który ma również podane parametry domyślne

3.6.3 Member Function Documentation

3.6.3.1 addAnimal()

```
void PetShop::addAnimal (
            Animal::Species spArg,
            Animal::Sex sexArg,
            int ageArg,
            double priceArg,
            double costArg )
```

Metoda pozwalająca dodać nowe zwierzę do sklepu (a właściwie kupić, bo trzeba wydać na nie pieniądze z kasy) Nowe zwierze ma cechy takie, jak parametry wejściowe funkcji i jest dodawane do wektora zwierząt sklepu

3.6.3.2 addPetFood()

```
void PetShop::addPetFood (
            int minAgeArg,
            int maxAgeArg,
            double priceArg,
            double costArg,
            int stockNumberArg,
            PetFood::Species animalsTypeArg,
            std::string manufacturerArg )
```

Metoda pozwalająca dodać nowy pokarm do sklepu (a właściwie kupić, bo trzeba wydać na niego pieniądze z kasy) Nowy pokarm ma cechy takie, jak parametry wejściowe funkcji i jest dodawany do wektora pokarmów sklepu

3.6.3.3 addPetToy()

```
void PetShop::addPetToy (
            int minAgeArg,
            double priceArg,
            double costArg,
            int stockNumberArg,
            PetFood::Species animalsTypeArg,
            std::string manufacturerArg )
```

Metoda pozwalająca dodać nową zabawkę do sklepu (a właściwie kupić, bo trzeba wydać na nią pieniądze) Nowa zabawka ma cechy takie, jak parametry wejściowe funkcji i jest dodawana do wektora zabawek sklepu

3.6.3.4 decideIfDiscount()

```
void PetShop::decideIfDiscount ( )
```

Podejmuje dla każdego produktu decyzję czy przecenić Dla każdego produktu w sklepie wywołuje funkcję wirtualną decydującą (i ewentualnie wykonującą) o obniżce

3.6.3.5 operator double()

```
PetShop::operator double ( )
```

Przeciążony operator rzutowania Sklep zrzutowany na double jest teoretyczną wartością sklepu - sumą cen wszystkich produktów i gotówki w kasie

3.6.3.6 operator+() [1/2]

```
void PetShop::operator+ (
            PetFood & pf )
```

Przeciążony operator dodawania Pozwala uzupełnić stan magazynowy konkretnego jedzenia do 25, ale tylko wtedy gdy są na to pieniądze Jeśli nie ma, uzupełnia tylko o tyle o ile się da (ile jest pieniędzy) Jeśli stan magazynowy jedzenie jest większy/równy 25, metoda nie wywołuje zmian

Parameters

pf	- referencja na pokarm, którego stan magazynowy ma zostać uzupełniony

3.6.3.7 operator+() [2/2]

```
void PetShop::operator+ (
            PetToy & pt )
```

Przeciążony operator dodawania Pozwala uzupełnić stan magazynowy konkretnej zabawki do 25, ale tylko wtedy gdy są na to pieniądze Jeśli nie ma, uzupełnia tylko o tyle o ile się da (ile jest pieniędzy) Jeśli stan magazynowy jedzenie jest większy/równy 25, metoda nie wywołuje zmian

Parameters

pt	- referencja na zabawkę, której stan magazynowy ma zostać uzupełniony

3.6.3.8 operator+=() [1/3]

```
void PetShop::operator+= (
            Animal * anim )
```

Przeciążony operator += Operator dodawania zwierzęcia do ostatniego rachunku w kasie Metoda sprawdza, czy jest jakikolwiek paragon w kasie i czy taki wskaźnik (przekazany jako parametr) jest w wektorze zwierząt sklepu, jeśli tak, to wstawia jego kopie do paragonu, a następnie wyrzuca oryginalny z wektora wskaźników zwierząt w sklepie Dodaje pieniądze ze sprzedaży do kasy fiskalnej

Parameters

anim	- wskaźnik na zwierzę, które ma zostać dodane do paragonu

3.6.3.9 operator+=() [2/3]

```
void PetShop::operator+= (
```

```
            PetFood * pf )
```

Przeciążony operator += Operator dodawania pokarmu do rachunku (ostatniego w kasie) Metoda sprawdza, czy jest jakikolwiek paragon w kasie i czy taki wskaźnik jest w wektorze pokarmu sklepu, jeśli tak, to w zależności od stanu magazynu jedzenia, tworzy nowy obiekt i wstawia wskaźnik do wektora w paragonie i zmniejsza stan magazynowy obiektu w sklepie, albo nie pozwala dokonać zakupu (gdy stan == 0) Dodaje pieniądze ze sprzedaży do kasy fiskalnej

Parameters

pf	- wskaźnik na pokarm, który ma zostać sprzedany

3.6.3.10 operator+=() [3/3]

```
void PetShop::operator+= (
            PetToy * pt )
```

Przeciążony operator += Operator dodawania zabawki do rachunku (ostatniego w kasie) Metoda sprawdza, czy jest jakikolwiek paragon w kasie i czy taki wskaźnik jest w wektorze zabawek sklepu, jeśli tak, to w zależności od stanu magazynu zabawki, tworzy nowy obiekt i wstawia wskaźnik do wektora w paragonie i zmniejsza stan magazynowy obiektu w sklepie, albo nie pozwala dokonać zakupu (gdy stan == 0) Dodaje pieniądze ze sprzedaży do kasy fiskalnej

Parameters

pt	- wskaźnik na pokarm, który ma zostać sprzedany

3.6.3.11 returnAnimal()

```
Animal * PetShop::returnAnimal (
            int n )
```

Metoda zwracająca n-ty element wektora zwierząt w sklepie Notacja programistyczna, n=0 to pierwszy element Obsługuje błędy - gdy podany za duży numer - zwraca ostatni, gdy za mały - pierwszy

3.6.3.12 returnPetFood()

```
PetFood * PetShop::returnPetFood (
            int n )
```

Metoda zwracająca n-ty element wektora pokarmu w sklepie Notacja programistyczna, n=0 to pierwszy element Obsługuje błędy - gdy podany za duży numer - zwraca ostatni, gdy za mały - pierwszy

3.6.3.13 returnPetToy()

```
PetToy * PetShop::returnPetToy (
            int n )
```

Metoda zwracająca n-ty element wektora zabawek w sklepie Notacja programistyczna, n=0 to pierwszy element Obsługuje błędy - gdy podany za duży numer - zwraca ostatni, gdy za mały - pierwszy

3.6.3.14 updateAdress()

```
void PetShop::updateAdress (
            std::string newAdress )
```

Uaktualnia adres sklepu na podany jako parametr

Parameters

newAdress	- adres sklepu, który ma zastąpić aktualny

3.6.3.15 updateName()

```
void PetShop::updateName (
            std::string newName )
```

Uaktualnia nazwę sklepu na podaną jako parametr

Parameters

newName	- nazwa sklepu, która ma zastąpić aktualną

3.6.3.16 updateShopDetails()

```
void PetShop::updateShopDetails ( )
```

Pozwala użytkownikowi zaktualizować informacje o sklepie Pyta się czy użytkownik chce zmodyfikować każdy modyfikowalny parametr i jeśli tak, to wywołuje odpowiednie prywatne funkcje modyfikujące Obsługuje błędy

3.6.3.17 updateTaxNumber()

```
void PetShop::updateTaxNumber (
            std::string newTaxNumber )
```

Uaktualnia numer NIP sklepu na podany jako parametr

Parameters

newTaxNumber	- numer NIP, który ma zastąpić aktualny

3.6.3.18 userAddAnimal()

```
void PetShop::userAddAnimal ( )
```

Pozwala dokupić do sklepu zdefiniowane przez użytkownika zwierze Obsługuje błędy

3.6.3.19 userAddPetFood()

```
void PetShop::userAddPetFood ( )
```

pozwala dokupić do sklepu zdefiniowany przez użytkownika pokarm Obsługuje błędy

3.6.3.20 userAddPetToy()

```
void PetShop::userAddPetToy ( )
```

pozwala dokupić do sklepu zdefiniowaną przez użytkownika zabawkę Obsługuje błędy

3.6.3.21 userChooseAnimal()

```
int PetShop::userChooseAnimal ( )
```

Pozwala wybrać użytkownikowi numer zwierzęcia, z dostępnych, a następnie zwraca go w postaci int Obsługuje błędy - sprawdza czy zwierze o danym numerze istnieje itp.

Returns

numer (int) wybranego przez użytkownika zwierzęcia (numeracja od 1)

3.6.3.22 userChoosePetFood()

```
int PetShop::userChoosePetFood ( )
```

Pozwala wybrać użytkownikowi numer jedzenia, z dostępnych, a następnie zwraca go w postaci int Obsługuje błędy - sprawdza czy pokarm o danym numerze istnieje itp.

Returns

numer (int) wybranego przez użytkownika pokarmu (numeracja od 1)

3.6.3.23 userChoosePetToy()

```
int PetShop::userChoosePetToy ( )
```

Pozwala wybrać użytkownikowi numer zabawi, z dostępnych, a następnie zwraca go w postaci int Obsługuje błędy - sprawdza czy zabawka o danym numerze istnieje itp.

Returns

numer (int) wybranej przez użytkownika zabawki (numeracja od 1)

3.6.3.24 userFeedAnimal()

```
void PetShop::userFeedAnimal ( )
```

Pozwala użytkownikowi nakarmić wybrane przez niego zwierze wybranym przez niego produktem Pyta użytkownika o to, które zwierze nakarmić którym produktem.

3.6.3.25 userUpdateAdress()

```
std::string PetShop::userUpdateAdress ( )
```

Pozwala użytkownikowi wpisać nowy adres sklepu i zwraca go Obsługuje błędy

3.6.3.26 userUpdateName()

```
std::string PetShop::userUpdateName ( )
```

Pozwala użytkownikowi wpisać nową nazwę sklepu i zwraca ją Obsługuje błędy

3.6.3.27 userUpdateTaxNumber()

```
std::string PetShop::userUpdateTaxNumber ( )
```

Pozwala użytkownikowi wpisać nowy numer NIP sklepu i zwraca go Obsługuje błędy - NIP musi być ciągiem dokładnie 10 cyfr

3.6.4 Friends And Related Function Documentation

3.6.4.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            PetShop & ps )  [friend]
```

Przeciążony operator << Operator umożliwiający przekazanie obiektu klasy sklep do obiektu klasy ostream w celu wyświetlenia go Wyświetlanie w formacie interfejsu

3.6.4.2 operator>>

```
std::istream& operator>> (
            std::istream & is,
            PetShop & ps )  [friend]
```

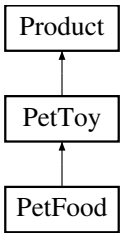
Przeciążony operator >> Operator wczytujący sklep z pliku (wymagany konkretny format)

The documentation for this class was generated from the following files:

- PetShop.hpp
- PetShop.cpp

3.7 PetToy Class Reference

Inheritance diagram for PetToy:



Public Member Functions

- PetToy (int minAgeArg=0, double priceArg=10.0, double costArg=10.0, Species speciesArg=Product↩ ::chomik, int stockNumberArg=1, std::string manufacturerArg="Producent")

  Konstruktor zabawki.
- ∼PetToy ()

  Dekonstruktor zabawki.
- std::string returnManufacturer ()

  Zwraca pole producenta w postaci string.
- int returnMinAge ()

  Zwraca minimalny wiek zwierzęcia odbiorcy.
- virtual bool decideIfDiscount ()
- virtual std::ostream & coutReceiptFormat (std::ostream &os)
- int **returnStockNumber** ()
- void changeStock (int n)
- void userUpdateMinAge ()

  Metoda pozwalająca użytkownikowi zaktualizować minimalny wiek zwierzęcia.
- void userUpdateStockNumber ()
- void userUpdateManufacturer ()

  Metoda pozwalająca użytkownikowi zaktualizować producenta.
- virtual std::ostream & productToOstreamEncyclopedia (std::ostream &os)

  Metoda wirtualna przekazująca obiekt w formacie "do wyświetlenia" do obiektu klasy ostream.

Protected Member Functions

- std::ostream & **minAgeToOStream** (std::ostream &)
- std::ostream & **manufacturerToOStream** (std::ostream &)
- std::ostream & **stockNumberToOStream** (std::ostream &)
- std::istream & **stockNumberFromIStream** (std::istream &)
- std::istream & **minAgeFromIStream** (std::istream &)
- std::istream & **manufacturerFromIStream** (std::istream &)

Protected Attributes

- int **minAge**
- std::string **manufacturer**
- int **stockNumber**

Friends

- std::ostream & operator<< (std::ostream &, PetToy &)
- std::istream & operator>> (std::istream &is, PetToy &pt)

Additional Inherited Members

3.7.1 Member Function Documentation

3.7.1.1 changeStock()

```
void PetToy::changeStock (
            int n )
```

Pozwala na zmane ilości na magazynie

Parameters

n	- nowa wartość stanu magazynowego produktu

3.7.1.2 coutReceiptFormat()

```
std::ostream & PetToy::coutReceiptFormat (
            std::ostream & os )  [virtual]
```

Metoda wirtualna

Przekazuje zabawkę w formacie paragonowym do obiektu klasy ostream

Implements Product.

3.7.1.3 decideIfDiscount()

```
bool PetToy::decideIfDiscount ( )  [virtual]
```

Metoda wirtualna decydująca o obniżeniu ceny

Jeśli zabawki jest więcej niż 20 -> zmniejsza marżę o połowę

Implements Product.

Reimplemented in PetFood.

3.7.1.4 userUpdateStockNumber()

```
void PetToy::userUpdateStockNumber ( )
```

Metoda pozwalająca użytkownikowi zaktualizować stan magazynowy

Uwaga: po zmianie stanu nie pobiera pieniędzy za dodany produkt

3.7.2 Friends And Related Function Documentation

3.7.2.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            PetToy & pf )  [friend]
```

Przeciążony operator <<

Pozwala wyświetlić informację o pokarmie (przekazuje obiekt do obiektu klasy ostream)

Format wyświetlania zgodny z wyświetlaniem w interfejsie oraz zapisywaniem do pliku

3.7.2.2 operator>>

```
std::istream& operator>> (
            std::istream & is,
            PetToy & pt )  [friend]
```

Przeciążony operator >>

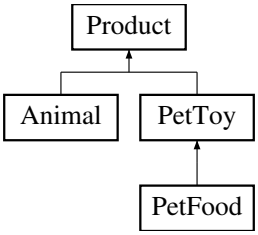
Wczytuje zabawkę zgodnie ze schematem zapisywania w pliku

The documentation for this class was generated from the following files:

- PetToy.hpp
- PetToy.cpp

3.8 Product Class Reference

Inheritance diagram for Product:



Public Types

- enum Species {
  chomik, **kanarek**, **krolik**, **mysz**,
  pajak, **papuga**, **rybka**, **snake** }

  Definicja typu wyliczeniowego dla gatunku/gatunku przeznaczenia.
- enum ProductType { **zwierze**, **zabawka**, **pokarm** }

  Definicja typu wyliczeniowego dla typu produktu.

Public Member Functions

- Product (Species=chomik, double=10.0, double=10.0)

  Konstruktor produktu.
- virtual ~Product ()

  Dekonstruktor produktu.
- double returnPrice ()

  Zwraca cenę produktu.
- double returnCost ()

  Zwraca koszt zakupu produktu.
- int returnSpecies ()
- std::string returnSpeciesString ()

  Zwraca string będący nazwą gatunku produktu.
- int returnProductType ()
- std::string returnProductTypeString ()

  Zwraca string będący typem produktu.
- virtual std::ostream & coutReceiptFormat (std::ostream &)=0
- virtual bool decideIfDiscount ()=0
- void userUpdateSpecies ()
- void userUpdatePrice ()
- void userUpdateCost ()
- virtual std::ostream & productToOstreamEncyclopedia (std::ostream &os)=0

  Metoda wirtualna przekazująca obiekt w formacie "do wyświetlenia" do obiektu klasy ostream.

Static Public Member Functions

- static int returnHowMany ()

  Statyczna metoda, która zwraca ile obiektów klasy Produkt jest aktualnie stworzonych.

Protected Member Functions

- std::ostream & **coutReceiptFormat_TypeSpecies** (std::ostream &os)
- std::ostream & **coutReceiptFormat_Price** (std::ostream &os)
- std::ostream & **speciesToOStream** (std::ostream &)
- std::ostream & **costToOStream** (std::ostream &)
- std::ostream & **priceToOStream** (std::ostream &)
- std::ostream & **typeToOStream** (std::ostream &)
- std::istream & **speciesFromIStream** (std::istream &)
- std::istream & **priceFromIStream** (std::istream &)
- std::istream & **costFromIStream** (std::istream &)

Protected Attributes

- Species **species**
- double **price**
- double **cost**
- ProductType **productType**

Friends

- std::ostream & operator<< (std::ostream &, Product &)

3.8.1 Member Function Documentation

3.8.1.1 coutReceiptFormat()

```
virtual std::ostream& Product::coutReceiptFormat (
            std::ostream &  )  [pure virtual]
```

Metoda wirtualna

Pobiera i zwraca referencje na obiekt klasy ostream i "przekazuje" mu produkt w formacie paragonowym (wykorzystuje metody wyświetlające typ, gatunek i cene w formacie paragonowym)

Format wyświetlania zgodny z formatem na paragonie

Implemented in Animal, and PetToy.

3.8.1.2 decideIfDiscount()

```
virtual bool Product::decideIfDiscount ( )  [pure virtual]
```

Metoda wirtualna

Analizuje produkt i robi jego przecenę, jeśli ma to sens (aby sprzedać zalegające produkty).

Warunkiem koniecznym przeceny jest cena wyższa od kosztu zakupu

Zwierzę zostaje przecenione, gdy jest stare, tzn gdy ma więcej niż 5 lat

Zabawka zostaje przeceniona, gdy jest jej więcej niż 20

Pokarm zostaje przeceniowy, gdy jest go więcej niż 20 i ma wąskie zastosowanie, tzn. wiek maksymalny karmionego zwierzęcia różni się od wieku minimalnego o maksymalnie 2 lata.

Jeśli są spełnione warunki to zmniejsza marżę o 50%

Returns

Zwraca informację, czy podjęta została decyzja o przecenie.

Implemented in Animal, PetFood, and PetToy.

3.8.1.3 returnProductType()

```
int Product::returnProductType ( )
```

Zwraca typ produktu w postaci int

Returns

0 - zwierze; 1 - zabawka; 2 - pokarm

3.8.1.4 returnSpecies()

```
int Product::returnSpecies ( )
```

Zwraca gatunek/gatunek przeznaczenia produktu w postaci int

Returns

int odpowiadający gatunkowi produktu (od 0 - chomik do 7 - snake)

3.8.1.5 userUpdateCost()

```
void Product::userUpdateCost ( )
```

Pozwala użytkownikowi wprowadzić koszt

Po poprawnym wprowadzeniu aktualizuje koszt

3.8.1.6 userUpdatePrice()

```
void Product::userUpdatePrice ( )
```

Pozwala użytkownikowi wprowadzić cenę

Po poprawnym wprowadzeniu aktualizuje cenę produktu

3.8.1.7 userUpdateSpecies()

```
void Product::userUpdateSpecies ( )
```

Pozwala użytkownikowi wprowadzić gatunek w formacie tekstowtym

Po poprawnym wprowadzeniu aktualizuje gatunek produktu

3.8.2 Friends And Related Function Documentation

3.8.2.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            Product & pr )  [friend]
```

Przeciążony operator <<

Operator strumieniowy pozwalający przekazać cenę i koszt produktu w odpowiednim formacie do obiektu klasy ostream

Format odpowiedni do wyświetlania oraz zgodny z formatem zapisu do pliku

The documentation for this class was generated from the following files:

- Product.hpp
- Product.cpp

3.9 Receipt Class Reference

Public Member Functions

- Receipt ()
  Konstruktor paragonu.
- ∼Receipt ()
  Dekonstruktor paragonu.
- void addProduct (Product ∗)

Friends

- std::ostream & operator<< (std::ostream &, Receipt &)

3.9.1 Member Function Documentation

3.9.1.1 addProduct()

```
void Receipt::addProduct (
            Product * ptr )
```

Metoda działająca na wektorze wskaźników polimorficznych

Pozwala na push_back danego produktu do wektora zakupionych produktów

3.9.2 Friends And Related Function Documentation

3.9.2.1 operator<<

```
std::ostream& operator<< (
            std::ostream & os,
            Receipt & rec )  [friend]
```

Przeciążony operator <<

przekazuje paragon do obiektu klasy ostream w określonym formacie

The documentation for this class was generated from the following files:

- Receipt.hpp
- Receipt.cpp

Index

addAnimal
    PetShop, 15
addPetFood
    PetShop, 16
addPetToy
    PetShop, 16
addProduct
    Receipt, 29
Animal, 5
    animalToOStream, 6
    decideIfDiscount, 6
    operator<<, 7
    operator>>, 7
    operator=, 6
animalToOStream
    Animal, 6

CashRegister, 7
    cashRegisterToOStream, 8
    collectCash, 8
    operator<<, 9
    operator>>, 9
    operator++, 8
    operator[], 8
    spendCash, 8
cashRegisterToOStream
    CashRegister, 8
changeStock
    PetToy, 23
collectCash
    CashRegister, 8
coutReceiptFormat
    PetToy, 23
    Product, 26

decideIfDiscount
    Animal, 6
    PetFood, 13
    PetShop, 16
    PetToy, 23
    Product, 26

ListMB
    ListMB< T >, 10
ListMB< T >, 10
    ListMB, 10
    operator=, 11
    popData, 11
    pushData, 11

NodeMB< T >, 12

operator double
    PetShop, 16
operator<<
    Animal, 7
    CashRegister, 9
    PetFood, 13
    PetShop, 21
    PetToy, 24
    Product, 28
    Receipt, 29
operator>>
    Animal, 7
    CashRegister, 9
    PetFood, 13
    PetShop, 22
    PetToy, 24
operator+
    PetShop, 16, 17
operator++
    CashRegister, 8
operator+=
    PetShop, 17, 18
operator=
    Animal, 6
    ListMB< T >, 11
operator[]
    CashRegister, 8

PetFood, 12
    decideIfDiscount, 13
    operator<<, 13
    operator>>, 13
PetShop, 14
    addAnimal, 15
    addPetFood, 16
    addPetToy, 16
    decideIfDiscount, 16
    operator double, 16
    operator<<, 21
    operator>>, 22
    operator+, 16, 17
    operator+=, 17, 18
    PetShop, 15
    returnAnimal, 18
    returnPetFood, 18
    returnPetToy, 18
    updateAdress, 19
    updateName, 19
    updateShopDetails, 19
    updateTaxNumber, 19

userAddAnimal, 20
userAddPetFood, 20
userAddPetToy, 20
userChooseAnimal, 20
userChoosePetFood, 20
userChoosePetToy, 20
userFeedAnimal, 21
userUpdateAdress, 21
userUpdateName, 21
userUpdateTaxNumber, 21
PetToy, 22
changeStock, 23
coutReceiptFormat, 23
decideIfDiscount, 23
operator<<, 24
operator>>, 24
userUpdateStockNumber, 24
popData
ListMB< T >, 11
Product, 25
coutReceiptFormat, 26
decideIfDiscount, 26
operator<<, 28
returnProductType, 27
returnSpecies, 27
userUpdateCost, 27
userUpdatePrice, 28
userUpdateSpecies, 28
pushData
ListMB< T >, 11

Receipt, 29
addProduct, 29
operator<<, 29
returnAnimal
PetShop, 18
returnPetFood
PetShop, 18
returnPetToy
PetShop, 18
returnProductType
Product, 27
returnSpecies
Product, 27

spendCash
CashRegister, 8

updateAdress
PetShop, 19
updateName
PetShop, 19
updateShopDetails
PetShop, 19
updateTaxNumber
PetShop, 19
userAddAnimal
PetShop, 20
userAddPetFood
PetShop, 20
userAddPetToy
PetShop, 20
userChooseAnimal
PetShop, 20
userChoosePetFood
PetShop, 20
userChoosePetToy
PetShop, 20
userFeedAnimal
PetShop, 21
userUpdateAdress
PetShop, 21
userUpdateCost
Product, 27
userUpdateName
PetShop, 21
userUpdatePrice
Product, 28
userUpdateSpecies
Product, 28
userUpdateStockNumber
PetToy, 24
userUpdateTaxNumber
PetShop, 21