PhoneBook Documentation

PhoneBook Class:

PhoneBook
- totalContacts: int
+ getTotalContacts(): int | + printContactWithFilters(option: int): void

The PhoneBook class represents a phone book and provides methods to manage contacts and retrieve information from the phone book.

Member Variables:

totalContacts: An int variable that stores the total number of contacts in the phone book.

Member Functions:

getTotalContacts(): Returns the total number of contacts in the phone book as an int.

printContactWithFilters(int option): Prints contact information based on the specified filter option.

option: An integer representing the filter option.

1: Prints the first name, last name, and age of the contact.

2: Prints the phone numbers of the contact.

3: Prints the address of the contact.

Person Class:

Person
- firstName: string - lastName: string - age: int - phone: Phone* - address: Address*
+ getFirstName(): string + getLastName(): string + getAge(): int + getPhone(): Phone* + getAddress(): Address* + printPerson(): void + setFirstName(firstName: string): void + setLastName(lastName: string): void

The Person class represents an individual person and provides access to their personal information, such as their first name, last name, age, phone number, and address.

Member Functions:

getFirstName(): Returns the first name of the person as a std::string.

getLastName(): Returns the last name of the person as a std::string.

getAge(): Returns the age of the person as an integer.

getPhone(): Returns a pointer to the Phone object associated with the person.

getAddress(): Returns a pointer to the Address object associated with the person.

printPerson(): Prints the information of the person, including their first name, last name, and age.

Phone Class:

Phone
- mobileNumber: string - landlineNumber: string
+ getMobileNumber(): string + getLandlineNumber(): string + setMobileNumber(mobile: string): void + setLandLineNumber(landline: string): void + printPhone(): void

The Phone class represents a phone and provides methods to retrieve and modify phone number information, including mobile and landline numbers.

Member Functions:

getMobileNumber(): Returns the mobile phone number as a std::string.

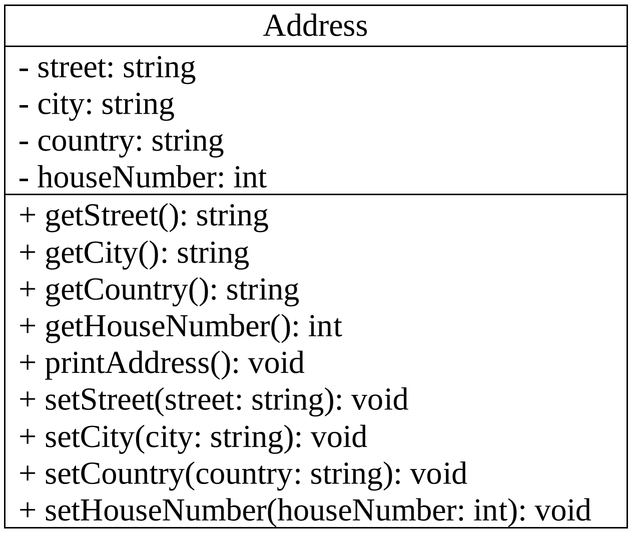
getLandlineNumber(): Returns the landline phone number as a std::string.

setMobileNumber(std::string mobile): Sets the mobile phone number.

setLandLineNumber(std::string landline): Sets the landline phone number.

printPhone(): Prints the mobile and landline phone numbers.

Adress Class:



```
Address
- street: string
- city: string
- country: string
- houseNumber: int
+ getStreet(): string
+ getCity(): string
+ getCountry(): string
+ getHouseNumber(): int
+ printAddress(): void
+ setStreet(street: string): void
+ setCity(city: string): void
+ setCountry(country: string): void
+ setHouseNumber(houseNumber: int): void
```

The Address class represents a physical address with street, city, country, and house number.

The Address class encapsulates the details of a physical address. It provides methods to access and manipulate the address information, such as retrieving the street, city, country, and house number. The class also includes a function to print the complete address details to the console.

Class Members:

- street (string): Stores the name of the street.

- city (string): Stores the name of the city.

- country (string): Stores the name of the country.

- houseNumber (int): Stores the house number.

Constructor:

Address(const std::string& street, const std::string& city, const std::string& country, int houseNumber):

Constructs an Address object with the specified street, city, country, and house number.

Member Functions:

std::string getStreet() const: Retrieves the street name; Returns: The street name as a string.

std::string getCity() const: Retrieves the city name; Returns: The city name as a string.

std::string getCountry() const: Retrieves the country name; Returns: The country name as a string.

int getHouseNumber() const: Retrieves the house number; Returns: The house number as an integer.

void printAddress(): Prints the complete address details to the console.

Handler:

Global Variables:

phoneBook: An instance of the linkedList class representing the phone book.

Function: loadData(const std::string& filename)

This function is responsible for loading data from a file into the phone book.

It takes a filename parameter specifying the name of the file to load.

Inside the function, the file is opened and read line by line.

Each line is then parsed using a comma-separated values (CSV) format.

The parsed values are used to create instances of the Person, Address, Phone, and PhoneBook classes, and a new node is added to the phoneBook linked list.

If the data format of a line is invalid, an error message is displayed.

Finally, the file is closed, and a message indicating the successful data load is printed.

Function: handle(int opt, int *load)

This function handles different options based on the input opt.

The load parameter is used to keep track of whether the data has already been loaded from a file or not.

If load is 0, the function loads data from a file before executing any other options.

The function uses a switch statement to handle different options, such as adding a contact, updating a contact, or deleting a contact.

Each option prompts the user for input, performs input validation, and performs the corresponding operations on the phone book.

After modifying the phone book, the data is saved to the file "data.csv".

Overall, the code provides functionalities to load, add, update, and delete contacts in the phone book. It also includes input validation to ensure the correctness of user input.

Utils:

The utils.h file provides utility functions for various checks and operations.

Functions:

checkName(const std::string& name): Checks if the given name is valid.

name: The name to be checked as a std::string.

Returns true if the name is valid (contains only alphabetic characters, spaces, hyphens, and apostrophes), otherwise returns false.

checkMenuOption(const int opt, int startValue, int endValue): Checks if the given menu option is within a valid range.

opt: The menu option to be checked as an int.

startValue: The starting value of the valid range as an int.

endValue: The ending value of the valid range as an int.

Returns true if the menu option is within the valid range, otherwise returns false.

checkMobilePhone(const std::string& mobileNumber): Checks if the given mobile phone number is valid.

mobileNumber: The mobile phone number to be checked as a std::string.

Returns true if the mobile phone number matches the specified pattern (valid Romanian mobile phone number format), otherwise returns false.

checkLandlineNumber(const std::string& landlineNumber): Checks if the given landline number is valid.

landlineNumber: The landline number to be checked as a std::string.

Returns true if the landline number matches the specified pattern (valid Romanian landline number format), otherwise returns false.

checkNumber(int n): Checks if the given number is positive.

n: The number to be checked as an int.

Returns true if the number is positive, otherwise returns false.

checkCountryName(const std::string& country): Checks if the given country name is valid.

country: The country name to be checked as a std::string.

Returns true if the country name matches the specified pattern (contains only alphabetic characters, spaces, hyphens, and apostrophes), otherwise returns false.

clearConsole(): Clears the console screen.

Note:

The regular expressions used in the checkMobilePhone and checkLandlineNumber functions are specific to the Romanian phone number formats and may need modification for different country formats.

The clearConsole function is implemented using the system function, which may not be platform-independent.