Politechnika Śląska

Wydział Informatyki, Elektroniki i Informatyki

Programowanie komputerów 2

Temat projektu: Archiwizator

autor	**Hubert Olszewski**
prowadzący	**mgr.inż. Maciej Długosz**
rok akademicki	**2019/2020**
kierunek	**informatyka**
rodzaj studiów	**SSI**
semestr	**2**
termin laboratorium	**piątek, 10:15 - 11:45**
sekcja	**61**
termin oddania sprawozdania	**19.06.2020**

1 Treść zadania

Napisać program umożliwiający przechowywanie struktury katalogów oraz zawartych w nich plików w jednym binarnym pliku archiwum. Program powinien mieć możliwość tworzenia i usuwania katalogów z archiwum, a także dodawania do niego plików z dysku oraz ich usuwania. Ponadto powinna istnieć możliwość przeglądania zawartości archiwum oraz wyodrębniania zarchiwizowanych plików na dysk. Poziom zagłębienia katalogów może być dowolny.

Uwagi techniczne Katalog ma podwieszone dwie listy: plików i katalogów potomnych (z kolejnymi listami katalogów i plików).

2 Analiza zadania

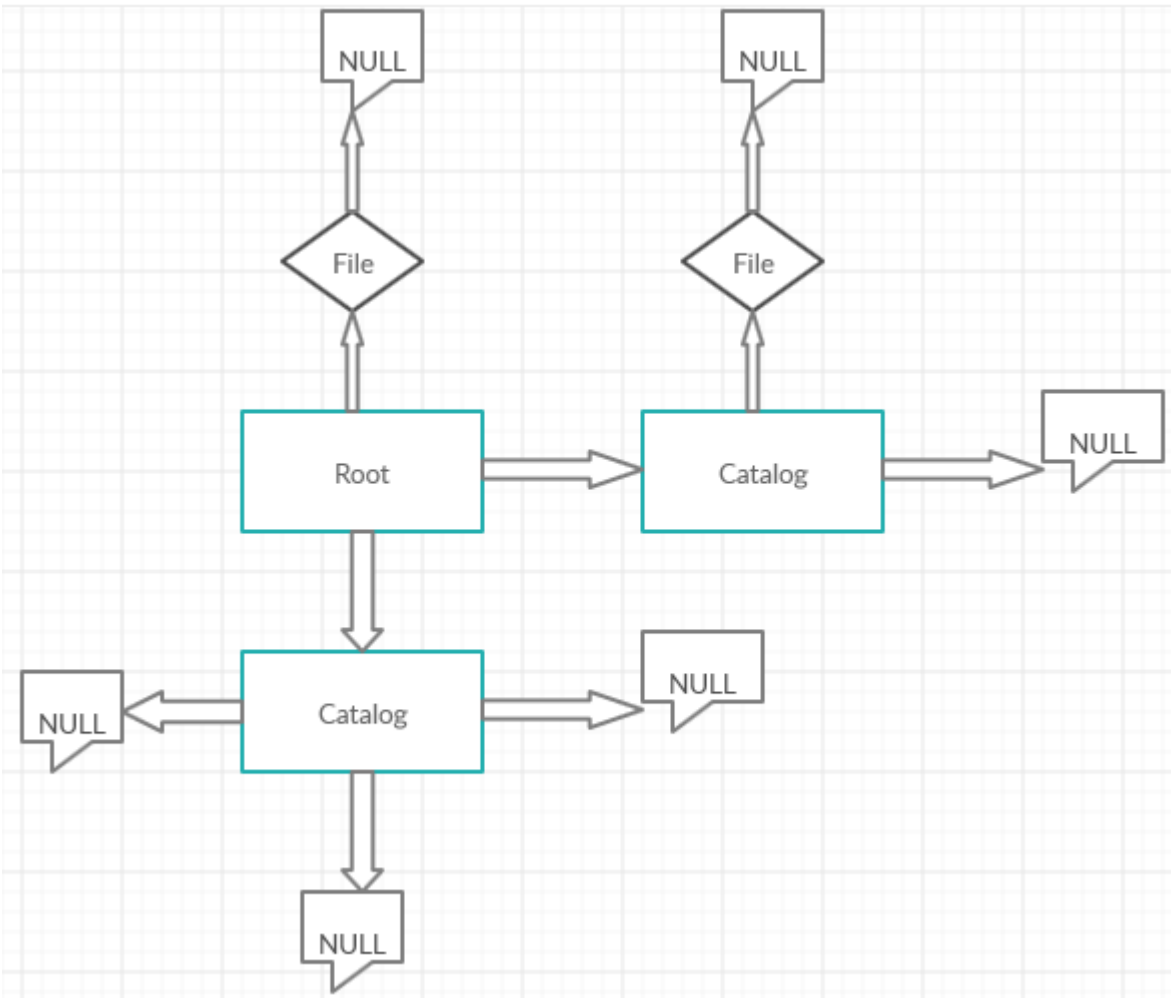
Zagadnienie przedstawia problem dodawania odczytywanych danych z plików do odpowiadającym im struktur, wykonywania operacji na tych danych oraz zapisywania wyniku operacji w jednym pliku binarnym, który przechowuje wprowadzone zmiany.

2.1 Struktury danych

W programie wykorzystano strukturę wielopoziomową. W pierwszej kolejności jest drzewo binarne przeznaczone do przechowywania informacji o katalogach. Drzewo binarne przechowuje dane w węzłach. Węzeł może mieć od 0 do 2 potomków, przy czym po lewej stronie węzła znajdują się potomki przechowujące katalogi potomne, natomiast po prawej stronie węzła znajdują się katalogi będące na równym poziomie zagnieżdżenia. Dodatkowo każdy węzeł posiada również listę jednokierunkową, w której przechowywane są dane z odczytanych plików. **Rysunek 1** przedstawia przykład takiej struktury wielopoziomowej. Taka struktura danych pozwala na łatwe nawigowanie pomiędzy katalogami i plikami.

2.2 Algorytmy

Program dodaje do drzewa binarnego nazwy katalogów oraz ścieżki, które zawierają drogę do niego. Sposób dodawania takiego węzła do drzewa jest określany na podstawie ścieżki, która jest podawana przez użytkownika. Ścieżka ta jest nawigatorem, dzięki któremu pliki są umieszczane do odpowiednich katalogów. Do listy jednokierunkowej przechowującej dane plików, elementy dodawane są w sposób posortowany według nazwy pliku. Wypisanie wielopoziomowej struktury jest realizowane poprzez rekurencyjne przejście przez węzły drzewa oraz iteracyjnie dla każdej listy jednokierunkowej. Zwolnienie pamięci wykonuje się w sposób przejścia do ostatnich elementów struktury.



Rysunek 1. Przykład struktury wielopoziomowej przechowującej katalogi oraz pliki.

3 Specyfikacja zewnętrzna

Program jest uruchamiany z linii poleceń. Są trzy możliwe otwarcia programu.

- Uruchomienie programu bez żadnych argumentów lub z błędnymi argumentami spowoduje zakończenie działania i wyświetlenie krótkiej pomocy np.

  Archiwizator.exe x1 x2

- Po przekazaniu do programu pierwszego argumentu po przełączniku **–o** spowoduje pominięcie odczytywania pliku i udzielenie dostępu do menu programu np.

  Archiwizator.exe –o

- W momencie przekazania argumentów w następującej kolejności, nastąpi archiwizacja podanych plików do podanego istniejącego katalogu w **archiwum**, a następnie uzyskamy dostęp do menu programu np.

  Archiwizator.exe katalog1/katalog2 plik1.txt plik2.bin

4 Specyfikacja wewnętrzna

Program został zrealizowany zgodnie z paradygmatem strukturalnym. W programie rozdzielono interfejs od logiki aplikacji.

4.1 Ogólna struktura programu

W funkcji głównej wywoływana jest funkcja **PobierzArchiwum.** Funkcja ta sczytuje do struktury zawartość pliku binarnego **Archiwum**, w przypadku, jeśli nie istnieje to wywoływana jest funkcja **UtwórzArchiwum** , która je tworzy i kończy program, w przeciwnym wypadku następuje wywołanie funkcji **CzytajParametry**, która sprawdza podane argumenty i w zależności od nich wykonuje się jeden z przypadków opisanych w specyfikacji zewnętrznej.

4.2 Opis typów i funkcji

Szczegółowy opis typów i funkcji został opisany w załączniku.

5 Testowanie

Program został przetestowany na różnego rodzaju plikach oraz na różnego rodzaju argumentach przy uruchamianiu. Argumenty nie poprawne (nie zawierające oczekiwanego formatu, niezgodne ze specyfikacją) powodują zgłoszenie błędu oraz wypisanie stosownego komunikatu.

Program został sprawdzony pod kątem wycieków pamięci przy użyciu funkcji z biblioteki zewnętrznej **visual leaks detector**.

6 Wnioski

Zrealizowałem zadanie, które polegało na napisaniu programu do archiwizacji danych. Program nie jest prosty, lecz nie jest również bardzo skomplikowany. Napisanie go wymagało umiejętności samodzielnego zarządzania pamięcią oraz zaznajomienia się z funkcjami biblioteki string.h, czyli obsługi łańcuchów znakowych. Dość wymagające okazało się usunięcie wycieków pamięci oraz odpowiednie przekazywanie typów zmiennych między funkcjami. Wiele razy w zły sposób przekazywałem argumenty do różnych funkcji, co sprawiało błędy w odczytywaniu danych. Wiele czasu musiałem zainwestować w znalezienie i naprawienie tych problemów, dzięki temu zdobyłem potrzebną wiedzę, którą wykorzystam aby nie powielać takich błędów w przyszłości.

My Project

Generated by Doxygen 1.8.17

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 katalog Struct Reference

`#include <Struktury.h>`

Public Attributes

- char **nazwa_katalogu** [MAX_SIZE_FILENAME]
- char * **sciezka**
- struct plik * **pFile**
- struct katalog * **pCatalog**
- struct katalog * **pNextCatalog**

3.1.1 Detailed Description

Struktura dynamiczna opisująca katalog. Jest w postaci drzewa-BST z podwieszonymi listami jednokierunkowymi plików.

See also

typedef struct plik

Parameters

nazwa_katalogu	tablica znaków przechowująca nazwę katalogu, ograniczona do 261 znaków
sciezka	wskaźnik na tablicę dynamiczną znaków, która przechowuje lokalizację katalogu

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

- Struktury.h

3.2 plik Struct Reference

`#include <Struktury.h>`

Public Attributes

- char **nazwa_pliku** [MAX_SIZE_FILENAME]
- char **data_modyfikacji** [DATA_FORMAT]
- unsigned int **rozmiar_pliku**
- char * **tresc_pliku**
- struct plik * **pNextPlik**

3.2.1 Detailed Description

Struktura dynamiczna opisująca plik. Jest w postaci listy jednokierunkowej.

Parameters

nazwa_pliku	tablica znaków przechowująca nazwę pliku, ograniczona do 261 znaków
data_modyfikacji	tablica znaków przechowująca datę zarchiwizowania pliku, ograniczona do 20 znaków
rozmiar_pliku	przechowuje liczbę znaków odczytanych z pliku
tresc_pliku	wskaźnik na tablicę dynamiczną znaków, która przechowuje zawartość odczytanego pliku

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

- Struktury.h

Chapter 4

File Documentation

4.1 Struktury.h File Reference

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

Classes

- struct plik
- struct katalog

Macros

- #define **_CRT_SECURE_NO_WARNINGS**
- #define MAX_SIZE_FILENAME 261
- #define DATA_FORMAT 20
- #define LiczbaNawigacji 3
- #define ARCHIWUM "Archiwum"
- #define KONIEC "KONIEC"
- #define catalog 2
- #define NowaSciezka 0
- #define wyciety 1
- #define iloscZnakow 1

Typedefs

- typedef FILE * **file**
- typedef struct plik plik
- typedef struct katalog katalog

Functions

- void czas (char *godzina)
- size_t PoliczSlowa (char *s1)
- void Wytnij_poczatek (char *sciezka, char **wyrazWydzielony, char **nowaSciezka)
- void Wytnij_Koniec (char *Sciezka, char **wyrazWyciety)
- void Init2DTab (char ***Dynamic, int size)
- void Free2DTab (char ***Dynamic, int size)
- int PoliczPliki (katalog *root)
- void printERROR ()
- plik * WypelnijPlik (plik *pNowy, const char *filename, char *data_modyfikacji, unsigned int rozmiar, char *tresc)
- bool DodajPlik (katalog **root, const char *filename, char *data_modyfikacji, unsigned int rozmiar, char *tresc)
- int OdczytajPlik (katalog **root, const char *filename, char *Sciezka)
- int UsunPlik (plik **head, const char *filename)
- void WyswietI_Plik (plik *head, int wciecie)
- plik * ZnajdzPlik (katalog *root, const char *filename)
- bool WyodrebnijPlikNaDysk (katalog **root, const char *filename)
- void Wypelnij_Katalog (katalog **root, const char *catalogName, char *Sciezka)
- void DodajKatalog (katalog **root, char *Sciezka)
- void WyswietlKatalogi (katalog *root, int wciecie)
- void ZwolnijStukture (katalog **root)
- int UsunKatalog (katalog **root, char *Sciezka)
- katalog * ZnajdzKatalog (katalog *root, char *Sciezka)
- bool PlikDoKatalogu (katalog **root, const char *filename, char *data_modyfikacji, unsigned int rozmiar, char *tresc, char *Sciezka)
- void WpiszPlikDoArchiwum (plik *head, file pfile, int liczbaPlikow)
- void WypelnijArchiwum (katalog *root, file pfile)
- bool ZapiszStruktureWArchiwum (katalog *root)
- void OdczytajPlikiKataloguZArchiwum (katalog **root, file prfile)
- bool OdczytajStrukture (katalog **root, file pFile)
- bool PobierzArchiwum (katalog **NowyRoot)
- void Menu (katalog **root)
- bool CzytajParametry (int argc, char **argv, char **Path, char ***FileNames)
- bool UtworzArchiwum ()
- void MemoryRemove (katalog **NowyRoot, char *Path, char **FileNames, int argc)

4.1.1 Macro Definition Documentation

4.1.1.1 ARCHIWUM

```
#define ARCHIWUM "Archiwum"
```

Łańcuch znakowy określający nazwę archiwum, plik binarny przechowujący zarchiwizowane dane.

4.1.1.2 catalog

```
#define catalog 2
```

Określa indeks w tablicy słów pod którym znajduje się ostatni wyraz ścieżki.

4.1.1.3 DATA_FORMAT

```
#define DATA_FORMAT 20
```

Wielkość tablicy dynamicznej przechowującej format daty.

4.1.1.4 iloscZnakow

```
#define iloscZnakow 1
```

Określa liczbę znaków jaka ma zostać wpisana lub odczytana do lub z pliku.

4.1.1.5 KONIEC

```
#define KONIEC "KONIEC"
```

Łańcuch znakowy określający wyraz kończący wydzielanie ścieżek katalogów.

4.1.1.6 LiczbaNawigacji

```
#define LiczbaNawigacji 3
```

Określa liczbę słów do zainicjalizowania w tablicy słów, potrzebnej do operowania na wydzielaniu ścieżek katalogu.

4.1.1.7 MAX_SIZE_FILENAME

```
#define MAX_SIZE_FILENAME 261
```

Określa maksymalną liczbę znaków nazw plików i katalogów.

4.1.1.8 NowaSciezka

```
#define NowaSciezka 0
```

Określa indeks w tablicy słów pod którym znajduje się reszta ścieżki po wydzieleniu.

4.1.1.9 wyciety

```
#define wyciety 1
```

Określa indeks w tablicy słów pod którym znajduje się pierwszy wyraz ścieżki.

4.1.2 Typedef Documentation

4.1.2.1 katalog

```
typedef struct katalog katalog
```

Struktura dynamiczna opisująca katalog. Jest w postaci drzewa-BST z podwieszonymi listami jednokierunkowymi plików.

See also

typedef struct plik

Parameters

nazwa_katalogu	tablica znaków przechowująca nazwę katalogu, ograniczona do 261 znaków
sciezka	wskaźnik na tablicę dynamiczną znaków, która przechowuje lokalizację katalogu

4.1.2.2 plik

```
typedef struct plik plik
```

Struktura dynamiczna opisująca plik. Jest w postaci listy jednokierunkowej.

Parameters

nazwa_pliku	tablica znaków przechowująca nazwę pliku, ograniczona do 261 znaków
data_modyfikacji	tablica znaków przechowująca datę zarchiwizowania pliku, ograniczona do 20 znaków
rozmiar_pliku	przechowuje liczbę znaków odczytanych z pliku
tresc_pliku	wskaźnik na tablicę dynamiczną znaków, która przechowuje zawartość odczytanego pliku

4.1.3 Function Documentation

4.1.3.1 czas()

```
void czas (
            char * godzina )
```

Funkcja odczytuje czas, który upłynął od 1970r. i na tej podstawie określa datę, w której został dodany plik do archiwum.

Parameters

godzina	C-string, w którym jest zapisywany format pobranego czasu

4.1.3.2 CzytajParametry()

```
bool CzytajParametry (
            int argc,
            char ** argv,
            char ** Path,
            char *** FileNames )
```

Funkcja dostaje parametry jakimi są przełączniki oraz nazwy plików z lini poleceń.

Parameters

argc	ilość parametrów z lini poleceń
argv	tablica ciągu znaków z lini poleceń
Path	zwraca odzielony fragment z argv reprezentujący lokalizację katalogu
FileNames	zwraca nazwy plików wydzielone z argv

Returns

zwraca prawdę jeśli funkcja została wykonana prawidłowo lub zwraca fałsz, gdy funkcja natrafiła na błędne dane

4.1.3.3 DodajKatalog()

```
void DodajKatalog (
            katalog ** root,
            char * Sciezka )
```

Funkcja dodaje element do drzewa katalogów względem lokalizacji określonej w ścieżce.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
Sciezka	C-string zawierający lokalizację katalogu

4.1.3.4 DodajPlik()

```
bool DodajPlik (
            katalog ** root,
            const char * filename,
            char * data_modyfikacji,
            unsigned int rozmiar,
            char * tresc )
```

Funkcja dodaje nowy element do listy plików w sposób posortowany względem nazwy pliku

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
filename	C-string zawierający nazwe pliku
data_modyfikacji	C-string zawierający datę odczytania pliku
rozmiar	przechowuje liczbę elementów sczytanych z pliku
tresc	C-string przechowujący zawartość odczytanego pliku

Returns

zwraca prawdę, gdy element został dodany poprawnie w przeciwnym wypadku zwraca fałsz

4.1.3.5 Free2DTab()

```
void Free2DTab (
            char *** Dynamic,
            int size )
```

Funkcja zwalnia pamięć tablicy słów.

Parameters

Dynamic	wskaźnik na dwuwymiarową tablicę
size	liczba zawierająca jak wiele pamięci zostanie zwolnione

4.1.3.6 Init2DTab()

```
void Init2DTab (
            char *** Dynamic,
            int size )
```

Funkcja inicjalizuje tablice słów.

Parameters

Dynamic	wskaźnik na dwuwymiarową tablicę
size	liczba zawierająca jak wiele miejsca na słowa ma zostać zainicjalizowane

4.1.3.7 MemoryRemove()

```
void MemoryRemove (
            katalog ** NowyRoot,
            char * Path,
            char ** FileNames,
            int argc )
```

Funkcja zbiorczo zwalnia pamięć struktury, tablicy słów oraz C-string w funkcji głównej main.

Parameters

NowyRoot	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
FileNames	przyjmuje tablicę słów, zawierającą nazwy plików
argc	ilość parametrów z lini poleceń

4.1.3.8 Menu()

```
void Menu (
            katalog ** root )
```

Funkcja główna programu. Wyświetla na ekran zawartość struktury i podaje użytkownikowi opcje wyboru do wykonania na danych zawartych w strukturze oraz je wykonuje przy pomocy funkcji pomocniczych.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST

4.1.3.9 OdczytajPlik()

```
int OdczytajPlik (
            katalog ** root,
            const char * filename,
            char * Sciezka )
```

Funkcja odczytuje plik do zarchiwizowania.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
filename	C-string zawierający nazwę pliku
Sciezka	lokalizacja, w której ma zostać umieszczony odczytany plik

Returns

zwraca 0 w przypadku, gdy plik do odczytu nie istnieje, -1 gdy nie został dodany poprawnie, 1 gdy plik został odczytany i dodany poprawnie

4.1.3.10 OdczytajPlikiKataloguZArchiwum()

```
void OdczytajPlikiKataloguZArchiwum (
            katalog ** root,
            file prfile )
```

Funkcja odczytuje informacje o zarchiwizowanych plikach w pliku binarnym(archiwum) i dodaje element do struktury

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
prfile	wskaźnik na plik binarny(archiwum)

4.1.3.11 OdczytajStrukture()

```
bool OdczytajStrukture (
            katalog ** root,
            file pFile )
```

Funkcja odczytuje informacje o zarchiwizowanych katalogach i plikach w nich zawartych. Na tej podstawie wypełnia strukture elementami.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
pFile	wskaźnik na plik binarny(archiwum)

Returns

zwraca prawdę, gdy element zostanie odczytany w przeciwnym wypadku fałsz

4.1.3.12 PlikDoKatalogu()

```
bool PlikDoKatalogu (
            katalog ** root,
            const char * filename,
            char * data_modyfikacji,
            unsigned int rozmiar,
            char * tresc,
            char * Sciezka )
```

Funkcja dodaje plik do katalogu na podstawie ścieżki.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
filename	C-string zawierający nazwę pliku
data_modyfikacji	C-string zawierający datę odczytania pliku
rozmiar	przechowuje liczbę elementów sczytanych z pliku
tresc	C-string przechowujący zawartość odczytanego pliku
Sciezka	C-string zawierający lokalizację katalogu

Returns

zwraca - prawdę, gdy plik został dodany poprawnie do katalogu. Fałsz - katalog nie został znaleziony

4.1.3.13 PobierzArchiwum()

```
bool PobierzArchiwum (
            katalog ** NowyRoot )
```

Funkcja otwiera archiwum i wykonuje OdczytajStrukture.

See also

OdczytajStrukture(katalog **root, file pFile)

Parameters

NowyRoot	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST

Returns

zwraca prawdę, gdy wykonana się poprawnie, w przypadku braku archiwum lub pustego archiuwm zwraca fałsz

4.1.3.14 PoliczPliki()

```
int PoliczPliki (
            katalog * root )
```

Funkcja zliczająca liczbe plików w danym węźle drzewa-BST.

Parameters

root	wskaźnik na korzeń drzewa katalogów

Returns

zwraca liczbę znalezionych plików w węźle

4.1.3.15 PoliczSlowa()

```
size_t PoliczSlowa (
            char * s1 )
```

Funkcja zlicza ilość słów w ścieżce.

Parameters

s1	C-string, docelowo ma przyjmować ścieżke do katalogu

Returns

zwraca liczbe słów

4.1.3.16 printERROR()

```
void printERROR ( )
```

Funkcja wypisuje na ekran informacje o błędzie.

4.1.3.17 UsunKatalog()

```
int UsunKatalog (
            katalog ** root,
            char * Sciezka )
```

Funkcja usuwa z pamięci węzeł drzewa(katalog) oraz wszystkie lewe poddrzewa(katalogi w nim umieszczone) i doczepione listy plików. Katalog docelowy jest usuwany na podstawie podanej ścieżki.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
Sciezka	C-string zawierający lokalizację katalogu

Returns

zwraca wartosc 1, gdy znaleziono katalog do usunięcia, wartosc 0 gdy element nie został znaleziony

4.1.3.18 UsunPlik()

```
int UsunPlik (
            plik ** head,
            const char * filename )
```

Funkcja usuwa plik ze listy plików na podstawie nazwy pliku.

Parameters

head	wskaźnik na oryginalny wskaźnik listy plików
filename	C-string zawierający nazwę pliku

Returns

zwraca 1 - w przypadku poprawnego wykonania funkcji, 0 - gdy usunięcie pliku nie było możliwe

4.1.3.19 UtworzArchiwum()

```
bool UtworzArchiwum ( )
```

Funkcja tworzy plik binarny archiwum.

Returns

zwraca prawdę kiedy archiwum zostało utowrzone lub fałsz gdy archiwum już istniało

4.1.3.20 WpiszPlikDoArchiwum()

```
void WpiszPlikDoArchiwum (
            plik * head,
            file pfile,
            int liczbaPlikow )
```

Funkcja wpisuje do pliku binarnego(archiwum) zawartość listy plików.

Parameters

head	wskaźnik na listę plików
pfile	wskaźnik na plik binarny
liczbaPlikow	przechowuje liczbę plików do wpisania

4.1.3.21 WyodrebnijPlikNaDysk()

```
bool WyodrebnijPlikNaDysk (
            katalog ** root,
            const char * filename )
```

Funkcja na podstawie nazwy wyodrębnia plik na dysk z archiwum.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
filename	C-string zawierający nazwę pliku

Returns

zwraca prawdę - gdy funkcja zostanie poprawnie wykonana, w momencie gdy nie ma pliku do wyodrębnienia - fałsz

4.1.3.22 Wypelnij_Katalog()

```
void Wypelnij_Katalog (
            katalog ** root,
            const char * catalogName,
            char * Sciezka )
```

Funkcja wypełnia element struktury katalogów.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST
catalogName	C-string zawierający nazwę katalogu
Sciezka	C-string zawierający lokalizację katalogu

4.1.3.23 WypelnijArchiwum()

```
void WypelnijArchiwum (
            katalog * root,
            file pfile )
```

Funkcja wpisuje do pliku binarnego(archiwum) zawartość całej struktury, wszystkie drzewa katalogów i listy plików.

Parameters

root	wskaźnik na drzewo katalogów
pfile	wskaźnik na plik binarny

4.1.3.24 WypelnijPlik()

```
plik* WypelnijPlik (
            plik * pNowy,
            const char * filename,
            char * data_modyfikacji,
            unsigned int rozmiar,
            char * tresc )
```

Funkcja wypełnia strukture pliku danymi.

Parameters

pNowy	wskaźnik na liste plików w danym węźle drzewa-BST
filename	C-string zawierający nazwe pliku

Parameters

data_modyfikacji	C-string zawierający datę odczytania pliku
rozmiar	przechowuje liczbę elementów sczytanych z pliku
tresc	C-string przechowujący zawartość odczytanego pliku

Returns

zwraca wskaźnik na utworzony element pliku

4.1.3.25 Wyswietl_Plik()

```
void Wyswietl_Plik (
            plik * head,
            int wciecie )
```

Funkcja wyświetla informacje zawarte liście plików.

Parameters

head	oryginalny wskaźnik na listę plików
wciecie	liczba określająca z jak wielkim wcięciem mają zostać wypisane informacje

4.1.3.26 WyswietlKatalogi()

```
void WyswietlKatalogi (
            katalog * root,
            int wciecie )
```

Funkcja wypisuje na ekran wszystkie informacje zawarte w całej strukturze.

Parameters

root	wskaźnik na drzewo katalogów
wciecie	liczba określająca z jak wielkim wcięciem mają zostać wypisane informacje

4.1.3.27 Wytnij_Koniec()

```
void Wytnij_Koniec (
            char * Sciezka,
            char ** wyrazWyciety )
```

Funkcja wydziela z podawanej ścieżki wyraz ostatni, nie modyfikując oryginału.

Parameters

Sciezka	C-string, z którego wydzielany jest ostatni wyraz
wyrazWyciety	wskaźnik na C-string, do którego jest wpisywany wydzielony wyraz

4.1.3.28 Wytnij_poczatek()

```
void Wytnij_poczatek (
            char * sciezka,
            char ** wyrazWydzielony,
            char ** nowaSciezka )
```

Funkcja dzieli podawaną ścieżkę na pierwszy wyraz oraz resztę ścieżki nie modyfikując oryginału.

Parameters

sciezka	C-string, z którego wycinany jest wyraz
wyrazWydzielony	wskaźnik na C-string, do którego wpisywany jest wyraz wycięty
nowaSciezka	wskaźnik na C-string, do którego jest wpisywana reszta ścieżki

4.1.3.29 ZapiszStruktureWArchiwum()

```
bool ZapiszStruktureWArchiwum (
            katalog * root )
```

Funkcja otwiera plik binarny i wykonuje funkcje WypelnijArchiwum.

See also

WypelnijArchiwum(katalog *root, file pfile)

Parameters

root	wskaźnik na drzewo katalogów

Returns

zwraca prawdę jeśli funkcja wykonała się poprawnie, fałsz - nie istnieje struktura do wpisania

4.1.3.30 ZnajdzKatalog()

```
katalog* ZnajdzKatalog (
            katalog * root,
            char * Sciezka )
```

Funkcja wyszukuje katalog na podstawie podanej ścieżki do katalogu.

Parameters

root	wskaźnik na drzewo katalogów
Sciezka	C-string zawierający lokalizację katalogu

Returns

zwraca wskaźnik na znaleziony katalog

4.1.3.31 ZnajdzPlik()

```
plik* ZnajdzPlik (
            katalog * root,
            const char * filename )
```

Funkcja znajduje plik w liście plików na podstawie nazwy.

Parameters

root	wskaźnik na drzewo katalogów
filename	C-string zawierający nazwę pliku

Returns

zwraca wskaźnik na znaleziony element

4.1.3.32 ZwolnijStukture()

```
void ZwolnijStukture (
            katalog ** root )
```

Funkcja usuwa z pamięci całą strukturę.

Parameters

root	wskaźnik na oryginalny wskaźnik struktury katalogów drzewa-BST

Index