Politechnika Śląska

Wydział Automatyki, Elektroniki i Informatyki

Programowanie Komputerów 2

Palindrom

autor	Borys Kowalik
prowadzący	Katarzyna Nurzyńska
rok akademicki	2019/20
kierunek	Informatyka
rodzaj studiów	SSI
semestr	2
sekcja	12
Termin oddania sprawozdania	2020-07-01

1. Treść zadania

Napisać program wyszukujący w pliku palindromy. Wyszukane palindromy zapisywane są do pliku wyjściowego w kolejności alfabetycznej. Nazwa pliku wejściowego jest podawana w linii poleceń po przełączniku -i, wyjściowego po przełączniku -o.

2. Analiza zadaniowa

Zagadnienie przedstawia problem pobrania z listy słów, tych które są palindromami i uszeregowanie ich alfabetycznie.

2.1 Struktury danych

W programie wykorzystano listę jednokierunkową do przechowywania słów. Każde słowo zawiera wskaźnik na początek listy dwukieronkowej liter z których się składa.

2.2 Algorytmy

Program sprawdza czy dane słowo jest palindromem przez porównywanie ze sobą odpowiednich liter przesuwając się od końca i początku. Liczba porównań jest proporcjonalna do liczby liter w słowie.

3. Specyfikacja zewnętrzna

Program jest uruchamiany z linii poleceń. Należy przekazać do programu nazwy plików: wejściowego i wyjściowego po odpowiednich przełącznikach (odpowiednio -i dla pliku wejściowego i -o dla pliku wyjściowego). Po uruchomieniu z przełącznikiem -h nastąpi wyświetlenie krótkiej pomocy, np.:

program.exe -o wyjscie.txt -i wejscie.txt

Pliki są plikami tekstowymi ale mogą mieć dowolne rozszerzenie (lub go nie mieć). Przełączniki mogą być podane w dowolnej kolejności. Jeśli będzie brakowało jednego z przełączników program nie zostanie wykonany i wyświetlona zostanie krótka pomoc. Jeśli plik wejściowy nie będzie istnieć wyświetlony zostanie odpowiedni komunikat błędu.

4. Specyfikacja zewnętrzna

Program został zrealizowany zgodnie z paradygmatem strukturalnym. W programie rozdzielono interfejs (komunikację z użytkownikiem) od logiki aplikacji (tworzenie listy słów).

4.1 Ogólna struktura programu

W funkcji głównej na początku pobierane są wartości z parametrów przekazanych z linii poleceń i zapisywane do odpowiednich zmiennych za pomocą funkcji *zamiana.* Następnie strumień wejściowy otwiera plik wejściowy. Jeśli plik nie istnieje zwracany jest komunikat błędu i program zostaje przerwany. Jeśli plik istnieje program rozpoczyna pętlę. Funkcja *pushback* dodaje litery ze słowa do tymczasowej listy liter do momentu napotkania na spację lub koniec pliku. Następnie funkcje *palindrom* sprawdza czy dane słowo jest palindromem – jeśli jest funkcja *dodaj* dodaje wskaźnik na dane słowo do listy słów, w innym wypadku zaś lista jest usuwana. Jest ono dodawane do listy słów alfabetycznie poprzez porównywanie go z każdym kolejnym słowem z listy z użyciem funkcji *porownanie* i dodanie go przed pierwszym słowem które jest po nim alfabetycznie. Oryginalne wskaźniki na początek i koniec listy są zerowane. Po pobraniu wszystkich słów funkcja *drukarka* wypisuje je do pliku wyjściowego po czym funkcja *zwolnij1* zwalnia pamięć.

4.2 Szczegółowy opis typów i funkcji

Szczegółowy opis typów i funkcji zawarty jest w załączniku.

5. Testowanie

Program został przetestowany na różnego rodzaju plikach. Program został sprawdzony pod kątem wycieków pamięci.

6. Wnioski

Program do wypisywania listy palindromów alfabetycznie jest programem prostym, chociaż wymaga samodzielnego zarządzania pamięcia. Najbardziej wymagające okazało stworzenie funkcji któa dodawała palindromy do listy słów w sposób alfabetyczny. Należało rozważyć przypadki gdy lista była pusta lub dane słowo należało dodać na początek.

Palindrom

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 slowa

`#include <funkcje.h>`

Pola danych

- slowo_type * **wyraz**
- struct slowa * pNext

  wskaznik na listę liter

3.1.1 Opis szczegółowy

Wezel listy słów

Dokumentacja dla tej struktury została wygenerowana z pliku:

- ConsoleApplication5/funkcje.h

3.2 Dokumentacja struktury slowo

`#include <funkcje.h>`

Pola danych

- char **litera**
- struct slowo * pPrev

  litera
- struct slowo * pNext

  wskaznik na poprzednią literę

3.2.1 Opis szczegółowy

Wezel listy liter

Dokumentacja dla tej struktury została wygenerowana z pliku:

- ConsoleApplication5/funkcje.h

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku ConsoleApplication5/funkcje.h

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

Struktury danych

- struct slowo
- struct slowa

Definicje typów

- typedef struct slowo slowo_type
- typedef struct slowa slowa_type

Funkcje

- void pushback (slowo_type **head, slowo_type **tail, char literka)
- int porownanie (slowo_type *head1, slowo_type *head2)
- void dodaj (slowa_type **glowa, slowo_type *wyrazik)
- void drukarka (slowa_type *glowa, char *output)
- int palindrom (slowo_type *glowa, slowo_type *ogon)
- void zamiana (char *destination, const char *source)
- int moje_strcmp (char string1[ ], char string2[ ])
- void zwolnij1 (slowa_type *glowa)
- void zwolnij2 (slowo_type *glowa)

4.1.1 Dokumentacja definicji typów

4.1.1.1 slowa_type

```
typedef struct slowa slowa_type
```

Wezel listy słów

4.1.1.2 slowo_type

```
typedef struct slowo slowo_type
```

Wezel listy liter

4.1.2 Dokumentacja funkcji

4.1.2.1 dodaj()

```
void dodaj (
            slowa_type ** glowa,
            slowo_type * wyrazik )
```

Funkcja dodaje slowo do listy slow @oparam[in, out] glowa wskaznik na pierwszy wezel listy @oparam[in, out] wyrazik wskaznik na liste liter

4.1.2.2 drukarka()

```
void drukarka (
            slowa_type * glowa,
            char * output )
```

Funkcja wypisuje listę slow do pliku wyjsciowego @oparam[in, out] glowa wskaznik na pierwszy wezel listy

Parametry

output	nazwa pliku wyjsciowego

4.1.2.3 moje_strcmp()

```
int moje_strcmp (
            char string1[],
            char string2[] )
```

Funkcja sprawdza czy dwa slowa sa identyczne

Parametry

string1	pierwsze slowo
string2	drugie slowo

Zwraca

funkcja zwraca 1 jeśli slowa sa identyczne

4.1.2.4 palindrom()

```
int palindrom (
            slowo_type * glowa,
            slowo_type * ogon )
```

Funkcja sprawdza czy slowo jest palindromem @oparam[in, out] glowa wskaznik na pierwszy wezel listy @oparam[in, out] ogon wskaznik na ostatni wezel listy

Zwraca

funkcja zwraca 1 jeśli slowo jest palindromem

4.1.2.5 porownanie()

```
int porownanie (
            slowo_type * head1,
            slowo_type * head2 )
```

Funkcja porownuje dwie listy liter @oparam[in, out] head1 wskaznik na pierwszy wezel pierwszej listy @oparam[in, out] head2 wskaznik na pierwszy wezel drugiej listy

Zwraca

funkcja zwraca 1 jeśli drugie slowo jest po pierwszym alfabetycznie

4.1.2.6 pushback()

```
void pushback (
            slowo_type ** head,
            slowo_type ** tail,
            char literka )
```

Funkcja dodaje literę do listy liter @oparam[in, out] head wskaznik na pierwszy wezel listy @oparam[in, out] tail wskaznik na ostatni wezel listy

Parametry

literka	litera ktora chcemy dodać do listy

4.1.2.7 zamiana()

```
void zamiana (
            char * destination,
            const char * source )
```

Funkcja kopiuje slowo ze zrodla do celu

Parametry

destination	docelowe słowo
source	slowo zrodlowe

4.1.2.8 zwolnij1()

```
void zwolnij1 (
            slowa_type * glowa )
```

Funkcja usuwa liste liter @oparam[in, out] glowa wskaznik na liste liter

4.1.2.9 zwolnij2()

```
void zwolnij2 (
            slowo_type * glowa )
```

Funkcja usuwa listę slow @oparam[in, out] glowa wskaznik na liste slow

Indeks