Palindrómy

Čo je to palindróm?

Palindróm je pojem odvodený od gréckeho slova, ktoré v preklade znamená bežiaci pospiatky. Ide o vetu, slovo, číslo či akúkoľvek inú sekvenciu, pre ktoré je typickou vlastnosťou to, že ich môžete čítať zľava doprava i sprava doľava a vždy vám vyjde to isté. Klasickým palindrómom je napríklad veta "Kobyla má malý bok".

Algoritmy na zisťovanie palindrómov

V tejto sekcii popíšem rôzne metódy zisťovania palindrómov, na ktoré som narazila pri vypracovávaní druhého zadania.

Metóda 01

#include <stdio.h>

int main() {

int n, obrateneCislo = 0, pom, povodneCislo;

printf ("Zadaj sekvenciu čísiel: ");

scanf ("%d", &n);

povodneCislo = n;

// otocenie povodneho cisla pomocou pomocnej premennej pom

while (n != 0) {

pom = n % 10;

obrateneCislo = obrateneCislo \* 10 + pom;

n /= 10;

}

// je palindrom aj povodneCislo a obrateneCislo su totozne

if (povodneCislo == obrateneCislo)

printf ("%d je palindrom.", povodneCislo);

else

printf ("%d nie je palindrom.", povodneCislo);

return 0;

}

Táto metóda je založená na tom, že palindróm sa číta rovnako z oboch strán. Používateľ má zadať číslo (integer), ktorý sa priradí do premennej n, a do povodneCislo. Potom je nájdené prevrátené číslo pomocou zvyšku po delení 10 a postupným delením 10, a priradené do obrateneCislo. V prípade, že povodneCislo a obrateneCislo sú totožné, ide o palindróm. Táto metóda však nefunguje na slová.

Metóda 02

#include <stdio.h>

#include <string.h>

int main() {

char a[100], b[100];

printf ("Zadaj sekvenciu písmen: ");

gets (a);

strcpy (b, a); // kopirovanie vstupu do pola b

strrev (b); // vytvorenie reverzneho stringu

// porovnanie imptu a reverzneho stringu

if (strcmp (a, b) == 0)

printf ("String je palindrom.");

else

printf ("String nie je palindrom.");

return 0;

}

Ide o obdobnú metódu tej prvej, s tým rozdielom, že táto funkcia je zameraná na stringy. V tomto prípade sú využívané funkcie z knižnice string.h. Do pola a sa načíta používateľský vstup. Ten sa potom skopíruje do pola b, a vytvorí sa z neho reverzný string. Tie sa následne porovnajú (rovnaký princíp ako prvá metóda) funkciou strcmp, ktorá vracia hodnotu 0 alebo 1, podľa toho, či sa polia (stringy) zhodujú.

Metóda 03

#include <stdio.h>

int main() {

char vstup[100];

int zaciatok, stred, koniec, dlzka=0;

printf ("Zadaj sekvenciu písmen: ");

gets (vstup);

while (vstup[dlzka] != '\0') {

dlzka++;

}

koniec = dlzka – 1;

stred = dlzka / 2;

for (zaciatok = 0; zaciatok < stred; zaciatok++) {

// ked sa najde nezhoda

if (vstup[zaciatok] != vstup[koniec]) {

printf ("Vstup nie je palindróm");

break;

}

koniec--;

}

// ak prebehne cely cyklus palindrom sa porovna az po stred

if (zaciatok == koniec)

printf ("Vstup je palindróm");

return 0;

}

Táto metóda nevyužíva funkcie s knižnice string.h, namiesto toho pristupuje do poľa vstupu cez indexy, a postupne porovnáva začiatočný znak s koncovým, kým sa nedostane do pólky, pričom kým nenájde nezhodu inkrementuje začiatok a dekrementuje koniec. Pokiaľ sa funkcia bez prerušenia do pólky dostane, ide o palindróm. V opačnom prípade o palindróm nejde a porovnávací cyklus sa pri nezhode zastaví.

Palindromové sub-stringy

Ide o nájdenie (najdlhšieho) sub-stringu v zadanom stringu, ktorý bude palindróm. Napíklad v skole „ajojky“ by to bol sub-string „joj“.

Na túto problematiku existuje tiež viacero metód. Napríklad:

Metóda 01 – hrubá sila

Najjednoduchší prístup je skontrolovať všetky sub-stringy, či sa jedná o palindróm alebo nie. Môžu prebehnúť tri cykly, vonkajšie dva vyberú všetky sub-stringy tým, že upravujú začiatky a konce, a vnútorný cyklus skúma každý sub-string, či je palindróm. Takýto prístup je ale náročný, pre veľký počet úkonov ktoré musia prebehnúť.

Metóda 02 – dynamické programovanie

Túto metódu nebudem hlbšie rozoberať, nakoľko nemáme písať o riešení zadania. Existujú však oveľa efektívnejšie spôsoby na to, ako takéto sub-stringy hľadať.

Užitočné odkazy

<https://www.tutorialgateway.org/c-program-to-check-the-given-string-is-palindrome/>

<https://www.programiz.com/c-programming/examples/palindrome-number>

<https://www.programmingsimplified.com/c-program-find-palindrome>

<https://www.programmingsimplified.com/c/source-code/c-program-palindrome-number>

<https://www.geeksforgeeks.org/longest-palindromic-substring-set-2/>

<https://www.geeksforgeeks.org/longest-palindrome-substring-set-1/>