Algorytmy na tekstach

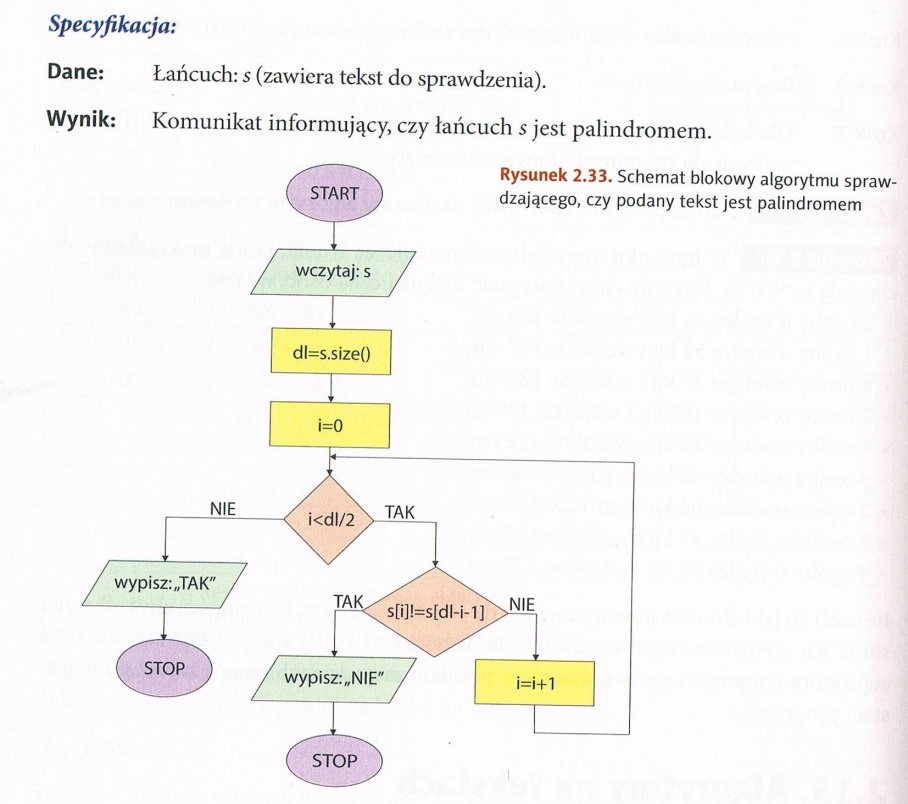
1. **Palindromy**

Palindrom to taki ciąg znaków, który czytany od początku i od końca brzmi tak samo. Np. oko, sos, kajak, itd.

Palindromami mogą być również całe zdania (zwykle pomija się spacje). Np.

* Oko w oko.
* Zakopane na pokaz.
* Nogawka jak wagon.
* Kobyła ma mały bok.
* Może jutro ta dama sama da tortu jeżom?

***Zadanie 1*** Napisz program wg poniższego schematu blokowego, który sprawdza czy podany tekst s jest palindromem.



1. **Sortowanie liter w testach**

Sortowanie tekstu to alfabetyczne uporządkowanie znaków zawartych w tekście. Np. ala -> aal. Będziemy tu zakładać, że teksty składają się z małych liter (bez polskich znaków).

Do sortowania tekstów można stosować algorytmy sortowania poznane na wcześniejszych lekcjach.

***Zadanie 2*** Napisz program, który sortuje znaki w podanym tekście algorytmem bąbelkowym.

1. **Anagramy**

Anagram to tekst, który powstaje przez przestawienie liter danego tekstu. Np.

* algorytm – logarytm
* tyran – narty
* alergia – galeria
* arbuz – burza

***Znajdowanie anagramów przez sortowanie***

Lista kroków:

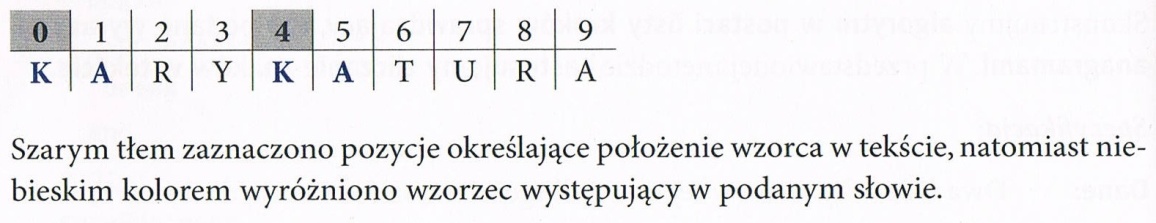
1. Wczytaj dwa łańcuchy znaków s1, s2.
2. Porównaj długości tekstów s1 i s2, jeśli są różne, wypisz „NIE” i zakończ algorytm.
3. Posortuj teksty s1, s2.
4. Sprawdź czy s1==s2, jeśli tak wypisz „TAK” w przeciwnym razie wypisz „NIE”.

***Zadanie 3*** Napisz program, który wczyta dwa teksty i sprawdzi czy są one anagramami.

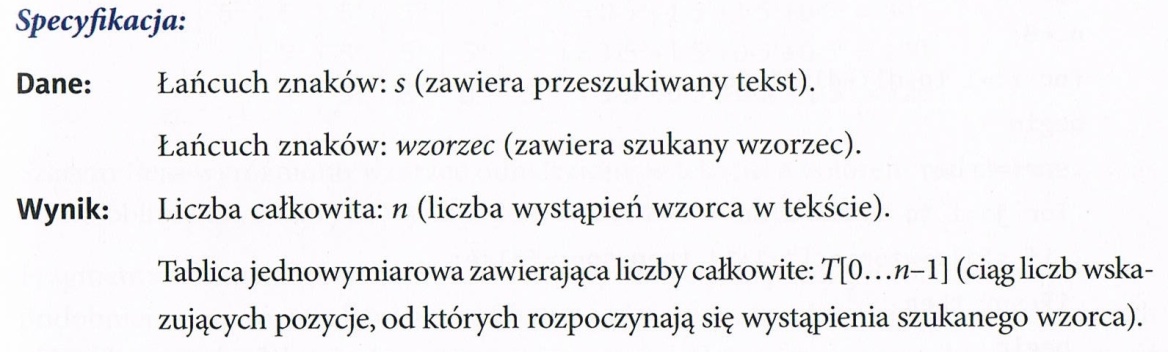
1. **Wyszukiwanie wzorca w tekście**

Wyszukiwanie wzorca w tekście polega na znalezieniu wszystkich wystąpień określonego łańcucha znaków w podanym tekście. W wyniku dostajemy ciąg liczb wskazujący początkowe położenia szukanego wzorca.

**Przykład** Niech tekstem będzie słowo: KARYKATURA, szukanym wzorcem tekst: KA. Wynikiem są liczy 0 i 4, ponieważ od tych pozycji rozpoczynają się wystąpienia wzorca.



***Wyszukiwanie wzorca z wykorzystaniem funkcji bibliotecznych do pracy z tekstem*** (patrz lekcja 14)

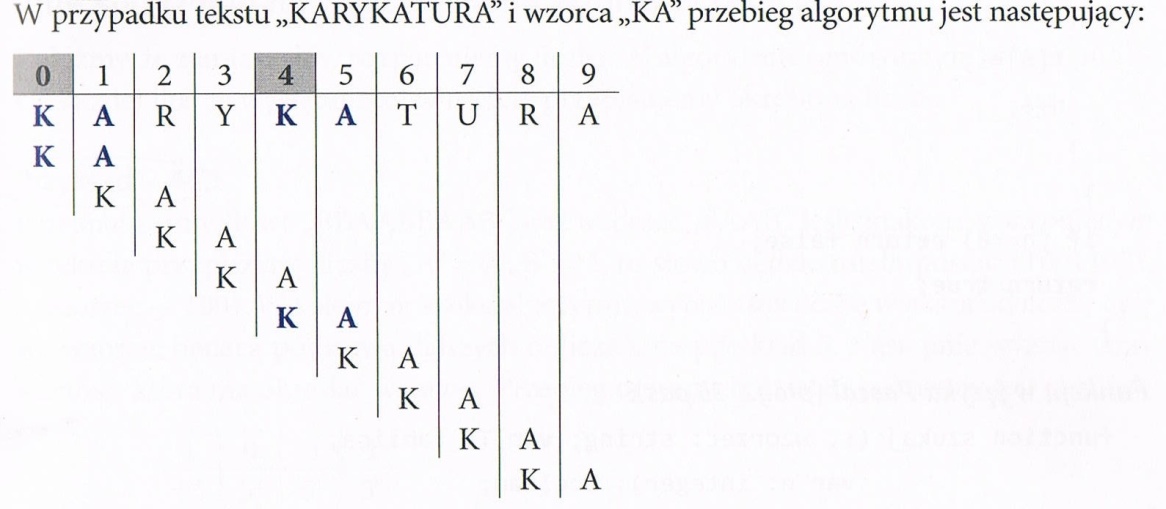


Lista kroków:

1. Wczytaj tekst s i wzorzec.
2. Jeśli długość tekstu jest mniejsza od długości wzorca wypisz NIE i zakończ algorytm.
3. Przypisz ilość wystąpień wzorca n=0.
4. Zastosuj funkcję find() do znalezienia wzorca. Jeśli wzorzec jest wykonaj kroki 5-6 i wróć do kroku 4, w przeciwnym razie wykonaj krok 7.
5. Zapisz w tablicy T[n] pozycję wzorca zwróconą przez funkcję find() i zwiększ n o 1.
6. Usuń wzorzec z tekstu funkcją erase().
7. Jeśli n=0 wypisz NIE, w przeciwnym razie wypisz TAK, wypisz n oraz ciąg n liczb z tablicy T[].

***Zadanie 4*** Napisz program, który wyszuka wszystkie wystąpienia wzorca w tekście lub poda, że wzorzec w tekście nie występuje. Użyj funkcji bibliotecznych. *(folder: wyszukanie wzorca 1)*

***Wyszukiwanie wzorca bez funkcji bibliotecznych do pracy z tekstem***

Algorytm będzie kolejno sprawdzał zgodność wzorca z tekstem znak po znaku, począwszy od pierwszej litery.

Do podanej poprzednio specyfikacji zapisano funkcję szukaj(), która realizuje wyszukiwanie wzorca w tekście.

bool szukaj(string s, string wzorzec, int T[], int &n)

{

int dlT=s.length();

int dlW=wzorzec.length();

if (dlT<dlW) return false;

n=0;

int j;

for(int i=0; i<dlT-dlW+1; i++)

{

for(j=i; j<i+dlW; j++)

if(s[j]!=wzorzec[j-i]) break; //jeśli wzorzec nie zgadza się: break

if(j==i+dlW) //wzorzec znaleziony

{

T[n]=i;

n++;

}

}

if(n==0) return false;

return true;

}

***Zadanie 5*** Napisz program, który wyszuka wszystkie wystąpienia wzorca w tekście lub poda, że wzorzec w tekście nie występuje. Użyj funkcji szukaj(). *(folder: wyszukanie wzorca 2)*