Laboratorium Informatyki II
Ćwiczenie 12

1 Ciąg tekstowy

Przetwarzanie ciągów tekstowych jest jednym z podstawowych zadań realizowanych przez programy. W języku C są one przechowywane w tablicy typu `char`, natomiast w C++ w sposób zdecydowanie wygodniejszy za pomocą obiektów klasy `string`.

Korzystając z tablic znakowych konieczne jest pilnowanie limitów znaków jaka może być zapisana oraz zakończenie ciągu tekstowego znakiem końca `\0`. Znacznym usprawnieniem przetwarzania ciągów tekstowych w tablicach znakowych jest stosowanie dynamicznej alokacji pamięci, ale nadal konieczne jest "ręczne" pilnowanie pojemności tablicy i zapewnienie zapisania znaku końca ciągu. Bez niego szereg funkcji dedykowanych do przetwarzania tego typu danych nie działały by prawidłowo. Funkcje te dostępne są w bibliotece `<string.h>` (dla języka C) i `<cstring>` (dla języka C++).

Obiekt klasy string i dedykowane dla niego funkcje zapewniają pełną automatyzację rozmiaru oraz wygodne narzędzia przetwarzania danych tekstowych.

2 Klasa *string*

Funkcje dedykowane dla klasy `string` dostępne są w bibliotece `<string>`. Klasa i biblioteka `string` dostępne są w ramach standardowej przestrzeni nazw, oznacza to że albo należy ustawić domyślną przestrzeń nazwa (`using namespace std;`) albo odwoływać się do przestrzeni standardowej przed deklaracją obiektu lub odwołaniem się do metod lub funkcji klasy i biblioteki `string`. Obiekt może przechować bardzo dużą liczbę znaków, zazwyczaj limitowaną maksymalną wartością typu `unsigned int`.

Klasa `string` jest częścią biblioteki standardowej języka C++ i formalnie nie ma konieczności podpinania biblioteki `<string>` do kodu, jest to realizowane automatycznie jeżeli wykryta zostanie w kodzie deklaracja obiektu klasy `string`. Jednakże ze względu na czytelność kodu zaleca się umieszczenie w kodzie dyrektywy z podpięciem biblioteki: `#include <string>`.

Obiekt klasy `string` może być tworzony z wykorzystaniem kilku konstruktorów:

- `std::string nazwa;` - podstawowy konstruktor tworzący obiekt pusty
- `std::string nazwa("Ciąg tekstowy");` - inicjuje obiekt ciągiem tekstowym podanym w cudzysłowach
- `std::string nazwa(20,'$');` - inicjuje obiekt określoną liczbą znaków podanych w apostrofach
- `std::string nazwa(obiekt);` - inicjowany innym obiektem klasy string, będzie kopią obiektu źródłowego

- `std::string nazwa(tablica, 20);` - inicjacja określoną liczbą znaków z wskazanej tablicy typu `char`
- `std::string nazwa(tablica+5, tablica+10);` - inicjuje obiekt określonym zakresem elementów tablicy
- `std::string nazwa(&obiekt[5], &obiekt[10]);` - inicjuje obiekt określonym zakresem elementów innego obiektu klasy `string`
- `std::string nazwa(obiekt, 5, 20);` - inicjuje obiekt określoną liczbą elementów innego obiektu klasy `string` począwszy od wskazanego elementu
- `std::string nazwa = {'T','E','K','S','T'};` - inicjacja listą znaków

Dane do obiektu `string` przypisuje się za pomocą:

- konstruktora
- przekierowanie danych z strumienia wejściowego operatorem '`>>`'
- funkcji lub metod odczytu danych z strumienia wejściowego: `getchar()`, `getline()` lub `cin.getline()`
- funkcji języka `C` z biblioteki `cstdio`: `gerc()` czy `scanf()`
- przeciążonych w klasie `string` operatorów '`=`', '`+`' czy '`+=`'
- metod klasy `string` jak `.append(std::string)` lub `.push_pack(char)`
- metod klasy `string` lub funkcji biblioteki `<string>` wykonujących operacji na obiekcie klasy `string`

2.1 Metody klasy *string*

Klasa `string` dostarcza szeregu metod wspierających przetwarzanie i analizę ciągów tekstowych przechowywanych w obiekcie `string`. Aby Do tych podstawowych i najczęściej stosowanych można zaliczyć:

- `.size()` - zwraca liczbę znaków
- `.length()` - zwraca liczbę znaków
- `.capacity()` - zwraca rozmiar pamięci zarezerwowany dla ciągu tekstowego
- `.reserve()` - umożliwia ustawienie rozmiaru pamięci dedykowanego dla ciągu tekstowego
- `.resize()` - zmienia rozmiar ciągu tekstowego
- `.find()` - zwraca indeks pierwszego wystąpienia szukanego znaku (ciągu znaków) począwszy od lewej strony ciągu tekstowego
- `.rfind()` - zwraca indeks pierwszego wystąpienia szukanego znaku (ciągu znaków) począwszy od prawej strony ciągu tekstowego
- `.find_first_of()` - zwraca indeks pierwszego wystąpienia dowolnego znaku z podanego ciągu znaków począwszy od lewej strony ciągu tekstowego

- `.find_first_not_of()` - zwraca indeks pierwszego wystąpienia dowolnego znaku który nie występuje w podanym ciągu znaków począwszy od lewej strony
- `.find_last_of()` -zwraca indeks ostatniego wystąpienia dowolnego znaku z podanego ciągu znaków począwszy od lewej strony ciągu tekstowego
- `.find_last_not_of()` - zwraca indeks ostatniego wystąpienia dowolnego znaku który nie występuje w podanym ciągu znaków począwszy od lewej strony
- `.clear()` – czyści zmienną
- `.assign()` – podmienia ciąg znaków
- `.insert()` – wstawia ciąg znaków
- `.replece()` – podmienia ciąg znaków
- `.pop_back()` – kasuje znak z końca
- `.swap()` – zamienia dwa ciągi tekstowe
- `.erase()` – kasuje fragment ciągu
- `.c_str()` - zwraca wskaźnik na tablicę typu `char` zapierającą ciąg tekstowy przechowywany w obiekcie klasy `string`

2.2 Funkcje przetwarzające dane obiektu klasy *string*

W języku `C++` jest dostępnych szereg metod dostarczanych przez biblioteki (`<iostream>`, `<cstring>` czy `sstream`) które bezpośrednio lub pośrednio mogą być stosowane do przetwarzania danych w obiektach klasy `string`. Poniżej zestawiono kilka najczęściej stosowanych funkcji:

- `to_string()` - konwertuje dane liczbowe na ciąg tekstowy pozwalając na łączenie danych liczbowych i ciągów tekstowych
- konwertujące ciągi znaków na liczby całkowite: `stoi()`, `stol()`, `stoll()` lub liczby zmiennoprzecinkowe: `stof()`, `stod()`, `stold()`
- zamieniające znaki w ciągu na wielkie `toupper()` lub małe `tolower()` litery
- funkcje z biblioteki `<cstring>` z języka C, mogą wymagać zastosowania metody `.c_str()` w celu zapewnienia kompatybilności

Zadania

W oparciu o informacje wykładowe i treść niniejszej instrukcji napisać program obiektowy. Program oparty jest na klasie `textAnaliza` który pozwala:

- zainicjowanie obiektu podanym ciągiem tekstowym, obiektem klasy string, innym obiektem klasy `textAnaliza` i tworząca pusty obiekt
- klasa dostarcza metod:
  - zwracającą informację o liczbie słów w przypisanym tekście

  - zwracającą informacje o liczbie i pozycjach wystąpienia podanego ciągu znaków w tekście
  - dodania ciągu tekstowego
  - zastąpienia bieżącego ciągu nowym

Napisać klasę rozszerzającą (dziedziczącą) klasę `textAnaliza` o nowe funkcjonalności:

- pamiętającą poprzednią wersję ciągu przechowywanego w obiekcie
- podmienianie znaku lub ciągu tekstowego innym znakiem lub ciągiem tekstowym
- wyświetlanie aktualnego i poprzedniego ciągu tekstowego