

Przetwarzanie łańcuchów znaków

1. Przykład

Poniższy przykład pokazuje sposób deklaracji łańcuchów znaków i tablic znakowych.

```
#include <cstdlib>
#include <iostream>
#include <string>

using namespace std;

int main()
{
    // deklaracja i definicja obiektów klasy string (C++)
    string str1("ala ma kota "); // konstruktor
    string str2 = "kot ma ale "; // operator

    // deklaracja i definicja tablic znakowych (null character string, C)
    const char *nap1 = "tim ma psa ";
    const char *nap2 = "pies ma tima ";

    // operator << obsługuje oba typy
    cout << "str1 = " << str1 << endl;
    cout << "str2 = " << str2 << endl << endl;

    cout << "nap1 = " << nap1 << endl;
    cout << "nap2 = " << nap2 << endl;

    return EXIT_SUCCESS;
}
```

- Przepisz, uruchom i przetestuj program.

2. Przykład

Poniższy przykład pokazuje sposób zmiany typów pozwalających przechowywać tekst.

```
#include <cstdlib>
#include <iostream>
#include <string>

using namespace std;

int main()
{
    string str1("ala ma kota ");
    string str2 = "kot ma ale ";

    const char *nap1 = "tim ma psa ";
    const char *nap2 = "pies ma tima ";

    // string z tablicy
    string str3(nap1); // konstruktor
    string str4 = nap2; // operator
}
```

```
cout << "str3 = " << str3 << endl;
cout << "str4 = " << str4 << endl << endl;

// tablica ze stringa
const char *nap3 = str1.c_str();
const char *nap4 = str2.c_str();

cout << "nap3 = " << nap3 << endl;
cout << "nap4 = " << nap4 << endl;

return EXIT_SUCCESS;
}
```

- Przepisz, uruchom i przetestuj program.

3. Przykład

Poniższy przykład pokazuje konkatencję i badanie długości tekstu.

```
#include <cstdlib>
#include <iostream>
#include <string>

using namespace std;

int main()
{
    string str1("ala ma kota ");
    string str2 = "kot ma ale ";

    const char *nap1 = "tim ma psa ";
    const char *nap2 = "pies ma tima ";

    // konkatencja + (operator przeciazony)
    string str5, str6;

    str5 = str1 + str2;
    str6 = str2 + nap2;

    cout << "str5 = " << str5 << endl;
    cout << "str6 = " << str6 << endl << endl;

    // dlugosc tekstu
    cout << "str1.size() = " << str1.size() << endl;
    cout << "strlen(nap1) = " << strlen(nap1) << endl;

    return EXIT_SUCCESS;
}
```

- Przepisz, uruchom i przetestuj program.

4. Przykład

Poniższy przykład pokazuje wybrane metody klasy **string**.

```
#include <cstdlib>
#include <iostream>
#include <string>
```

```
using namespace std;

int main()
{
    string str1("ala ma kota ");

    // dodawanie i usuwanie znaku
    str1.push_back('+');
    cout << "str1 = " << str1 << endl;

    str1.pop_back();
    cout << "str1 = " << str1 << endl << endl;

    // porównywanie łańcuchów
    int wynik;

    string str7 = "dzień dobry";
    string str8 = "dobranoc";

    cout << "str7 = " << str7 << endl;
    cout << "str8 = " << str8 << endl << endl;

    wynik = str7.compare(str7);
    cout << "str7.compare(str7) = " << wynik << endl;

    wynik = str7.compare(str8);
    cout << "str7.compare(str8) = " << wynik << endl << endl;

    // dodawanie
    str7.append(" jest osma");
    cout << "str7 = " << str7 << endl << endl;

    // wstawianie
    str7.insert(12, "kotku ");
    cout << "str7 = " << str7 << endl << endl;

    // usuwanie
    str7.erase(12, 6);
    cout << "str7 = " << str7 << endl << endl;

    // wybieranie
    string kolory = "czerwony zielony niebieski";
    string kolor = kolory.substr(9, 7);

    cout << "kolory = " << kolory << endl;
    cout << "kolor = " << kolor << endl << endl;

    // wstawianie
    kolory.replace(9, 7, "czarny");
    cout << "kolory = " << kolory << endl << endl;

    // szukanie
    string str9 = "łańcuchy są wyszukiwane bardzo szybko";

    cout << "str9.find(\"sz\") = " << str9.find("sz") << endl;
    cout << "str9.rfind(\"sz\") = " << str9.rfind("sz") << endl;

    return EXIT_SUCCESS;
}
```

- Przepisz, uruchom i przetestuj program.

5. Przykład

Poniższy przykład pokazuje sposób wczytywania tekstu z białymi znakami.

```
#include <cstdlib>
#include <iostream>
#include <string>

using namespace std;

int main()
{
    string imieNazwisko;

    cout << "Podaj imie i nazwisko oddzielone spacja: ";

    // konstrukcja cin >> imieNazwisko nie zadziała poprawnie
    getline(cin, imieNazwisko);
    cout << imieNazwisko << endl;

    return EXIT_SUCCESS;
}
```

- Przepisz, uruchom i przetestuj program.

Formatowanie liczb zmiennoprzecinkowych

6. Przykład

Poniższy przykład pokazuje zmianę domyślnego formatowania liczb zmiennoprzecinkowych strumienia `cout`.

```
#include <cstdlib>
#include <iostream>
#include <sstream>
#include <string>
#include <iomanip>

using namespace std;

int main()
{
    int i = 10;
    double p = 3.14159265359;
    double s = 5.67040040e-8;

    // domyslne formatowanie
    cout << "i = " << i << endl;
    cout << "p = " << p << endl;
    cout << "s = " << s << endl << endl;

    // trzy miejsca po przecinku
    cout << setprecision(3);
    cout << "i = " << i << endl;
    cout << "p = " << p << endl;
    cout << "s = " << s << endl << endl;

    // szesnascie miejsc po przecinku
    cout << setprecision(16);
```

```
cout << "i = " << i << endl;
cout << "p = " << p << endl;
cout << "s = " << s << endl << endl;

// uzupełnianie do szesnastu miejsc po przecinku
// granica dokładności typu double
cout << fixed << setprecision(16);
cout << "i = " << i << endl;
cout << "p = " << p << endl;
cout << "s = " << s << endl << endl;

return EXIT_SUCCESS;
}
```

- Przepisz, uruchom i przetestuj program.

7. Przykład

Poniższy przykład pokazuje sposób zmiany formatowania liczb zmiennoprzecinkowych bez zmiany domyślnych ustawień strumienia **cout** (z wykorzystaniem strumienia napisowego **stringstream**).

```
#include <cstdlib>
#include <iostream>
#include <sstream>
#include <string>
#include <iomanip>

using namespace std;

int main()
{
    int i = 10;
    double p = 3.14159265359;
    double s = 5.67040040e-8;

    stringstream streamLiczba;
    string stringLiczba;

    streamLiczba << fixed << setprecision(16);

    streamLiczba << i;
    stringLiczba = streamLiczba.str();
    streamLiczba.str(string());

    cout << "i = " << stringLiczba << endl;

    streamLiczba << p;
    stringLiczba = streamLiczba.str();
    streamLiczba.str(string());

    cout << "p = " << stringLiczba << endl;

    streamLiczba << s;
    stringLiczba = streamLiczba.str();
    streamLiczba.str(string());

    cout << "s = " << stringLiczba << endl;

    return EXIT_SUCCESS;
}
```

- Przepisz, uruchom i przetestuj program.

Zadania dodatkowe

1. Napisz program, który prosi o podanie, a potem wyświetla informacje o użytkowniku:

```
Podaj nazwę ulicy na ktorej mieszkasz? Stefana Batorego
Podaj nr domu? 43/15
Na ile oceniasz swoje umiejetności programowania w skali [2-6]? 3
Twój staz programistyczny? 5
Adres: Stefana Batorego 43/15
Ocena: 2
Staż: 5 dni
```

Program powinien przyjmować adres składający się więcej niż z jednego wyrazu. Dodatkowo ocena podana przez użytkownika powinna być przy wyświetleniu mniejsza o 1. Pogrubione informacje to te, które wpisuje użytkownik.

2. Napisz program, który prosi o podanie osobno imienia i nazwiska, a wyświetlający to w taki o to sposób:

```
| Nazwisko, Imię - życzę miłego dnia.
```

3. Napisz program, w którym użytkownik podaje dane techniczne trzech samochodów: marka, model, pojemność silnika, prędkość maksymalna. Dodatkowo oblicz ich średnią prędkości i pojemność silnika. Wszystkie dane mają być wprowadzone, oraz wyświetlone w sposób czytelny dla użytkownika podczas działania programu. Zaprojektuj czytelny interfejs. Wykorzystaj funkcję `setw()` (biblioteka `iomanip.h`) i metodę `assign()` klasy `string` w celu wielokrotnego dodawania tego samego znaku.

```
+-----+-----+-----+
| Marka      | Pojemnosc | Predkosc |
+-----+-----+-----+
| Fiat 126p  | 0.65      | 140      |
| Audi S6    | 4.2       | 250      |
| Syrena 105 | 0.84      | 120      |
+-----+-----+-----+
| Srednia predkosc: 170
| Srednia pojemnosc: 1.8967
+-----+-----+-----+
```

Do przechowywania danych o samochodach wygodnie będzie użyć struktury:

```
struct TSamochod
{
    string nazwa;
    float pojemnosc;
    int predkosc;
};
```