Podstawy programowania: Laboratorium nr 1
Wprowadzenie.

2017-2018

mgr inż. Przemysław Walkowiak
dr inż. Michał Ciesielczyk

Instrukcja

W czasie pisania programu pamiętaj o:

1. dbaniu o czytelność kodu (odpowiednie formatowanie kodu, nazewnictwo zmiennych adekwatne do ich znaczenia, komentarze),
2. dbaniu o czytelność interfejsu z użytkownikiem (w sposób jawny pytaj użytkownika jakie informacje ma podać oraz opisuj informację, które zwracasz),
3. przed fragmentem implementującym poszczególne zadania umieść komentarz: `/*Zadanie X */` oraz wypisz na ekranie analogiczny komunikat (X jest numerem zadania): `std::cout << "Zadanie X"<< std::endl;`,
4. w zadaniach wymagających udzielenia komentarza bądź odpowiedzi, należy umieścić go w kodzie programu (np. w postaci komentarza albo wydrukować na ekranie).

Wprowadzenie

Twój pierwszy program

Każdy program w C++ musi posiadać funkcję główną (`int main()`), w której się rozpoczyna (oraz kończy) działanie programu.

Listing 1: Program "Hello world!" w C++.

```
#include <iostream>
#include <string>
using namespace std;

int main() {
    // wyświetla na ekranie napis "Hello world!"
    cout << "Hello world!" << endl;
}
```

Wewnątrz funkcji głównej (tj. pomiędzy nawiasami klamrowymi { oraz }) umieszczane są instrukcje, które zostaną wykonane po jego uruchomieniu. W przypadku Listingu 1 jest to wypisywanie podanego napisu na ekran (linia 7). Zwróć uwagę, że na końcu każdej instrukcji jest znak `;`.

Znaki `//` oznaczają początek komentarza (linia 6 na Listingu 1) w danej linii – nie są one analizowane przez kompilator. Innymi słowy nie są częścią wykonywanego programu, a jedynie informacją dla programisty.

Pierwsze trzy linie w powyższym kodzie źródłowym to dyrektywy preprocesora dołączające odpowiednie pliki nagłówkowe (tu: `iostream` i `string`) oraz instrukcja wprowadzenia podanej przestrzeni nazw (tu: `std`)– ich dokładniejsze znaczenie poznasz na dalszej części kursu. Są one niezbędne do wykonania zadań z tego laboratorium.

Zwróć uwagę, że celem zwiększenia czytelności niektórych przykładów podanych w materiałach do tego laboratorium często pomijane jest wprowadzenie przestrzeni nazw `std` oraz dołączanie popularnych plików nagłówkowych (takich jak `std::string`).

Obsługa wejścia-wyjścia (std::cin, std::cout)

W programie z interfejsem konsolowym w języku C++ interakcja z użytkownikiem może być zrealizowana z wykorzystaniem obiektów `std::cout` oraz `std::cin`.

Deklaracje `std::cout` oraz `std::cin` znajdują się odpowiednio w plikach nagłówkowych `<ostream>` oraz `<istream>`, ewentualnie `<iostream>`. `std::cout` służy do wyświetlania informacji na standardowym wyjściu (np. na konsoli), `std::cin` pozwala na wczytanie informacji ze standardowego wejścia (np. z konsoli).

Aby wyświetlić komunikat na ekranie należy skorzystać z następującej składni:

```
cout << "Witaj świecie" << endl;
```

Jeśli powyższa linijka z różnego powodu "nie działa" sprawdź czy na pewno umieściłeś ją w odpowiednim miejscu (wewnątrz funkcji) oraz czy pamiętałeś o dodaniu odpowiednich instrukcji na początku pliku (jak na Listingu 1).

Zmienna `endl` odpowiada za reprezentację znaku końca linii (tj. wstawia "enter"). Operację `<<` można łączyć wielokrotnie. Np.:

```
cout << "Mam na imię " << "Jan " << "i mam 15 lat"<< endl;
```

Ewentualnie wplatać zmienne:

```
string imie = "Jan";
int wiek = 15;
cout << "Mam na imie " << imie << " i mam " << wiek << " lat" << endl;
```

Aby wczytać informacje od użytkownika można skorzystać z następującej składni:

```
int liczba;
cin >> liczba;

string napis;
cin >> napis;
```

Własne funkcje

W praktyce, każdy program C++ składa się z wielu funkcji. Funkcja to po prostu odnośnik na fragment kodu wykonującego określone zadanie. W programach na tych laboratoriach możesz zdefiniować oddzielne funkcje dla każdego zadania np. następujący sposób:

```
void zadanie1() {
    // Rozwiązanie zadania 1
}
```

```
5 void zadanie2() {
    // Rozwiązanie zadania 2
}
```

Aby wykonać daną funkcję (w funkcji głównej lub w dowolnej innej funkcji) należy odwołać się do jej nazwy w następujący sposób:

```
zadanie1();
zadanie2();
```

Nazwa funkcji może składać się z liter łacińskich (a-z, A-Z), cyfr (0-9) oraz podkreślnika (_). Nazwa jednak nie może zaczynać się od cyfry, ani być równa jednemu z zarezerwowanych słów w C++. Zwróć uwagę, że wielkość liter ma znaczenie.

Instrukcja warunkowa if-else

Instrukcja **if** pozwala na warunkowe wykonanie innej instrukcji. Składnia tej instrukcji jest następująca:

```
if (condition)
    statement1;
else statement2;
```

gdzie:

- `condition` to warunek, który zostanie sprawdzony,
- `statement1` to instrukcja, która zostanie wykonana, gdy `condition` zostanie spełniony,
- `statement2` to instrukcja, która zostanie wykonana, gdy `condition` nie zostanie spełniony,

Ponieważ `condition` nie może być jednocześnie prawdziwy (**true**) i fałszywy (**false**), `statement1` oraz `statement2` nigdy nie zostaną uruchomione jednocześnie.

Więcej informacji: `http://en.cppreference.com/w/cpp/language/if`.

Materiały

- Programming – Principles and Practice Using C++. Bjarne Stroustrup.
- `http://en.cppreference.com/w/` – aktualna dokumentacja C++
- `https://isocpp.org/faq` – pytania i dobrze sformułowane odpowiedzi z przykładami na większość nurtujących Was pytań na każdym etapie nauki programowania w C++
- `https://msdn.microsoft.com/en-us/library/3bstk3k5.aspx`
- `http://www.cplusplus.com/`

Zadania

Zdefiniuj oddzielne funkcje dla następujących zadań. Przykładowo możesz je nazwać `zadanie1`, `zadanie2`, itd. W każdej funkcji wyświetl informację, że dane zadanie zostało uruchomione, np.:

```
std::cout << "Zadanie X" << std::endl;
```

Wszystkie zadania wywołaj po kolei z funkcji głównej (**int** `main()`).

Zadanie 1

Zaimplementuj ankietę pytając użytkownika o takie informacje jak: imię, nazwisko oraz wiek. Następnie wypisz powyższe informacje na ekranie. Do komunikacji z użytkownikiem wykorzystaj `std::cin` oraz `std::cout`.

Dodatkowe informacje:

- `std::cin`, `std::cout` - `http://www.cplusplus.com/reference/iostream/`.

Zadanie 2

Wczytaj od użytkownika dwie liczby całkowite. Następnie, wykonaj na nich operacje dodawania, mnożenia, odejmowania i dzielenia, za każdym razem wyświetlając wynik działania na ekranie.

Zadanie 3

Wczytaj od użytkownika długości trzech odcinków a, b, c. Sprawdź czy można z nich zbudować trójkąt. Wynik sprawdzania wyświetl na ekranie.

Wskazówka 1 Trójkąt można zbudować z trzech odcinków wtedy i tylko wtedy gdy długość każdego boku jest mniejsza niż suma długości dwóch pozostałych boków.

Na następne zajęcia

- Wypisanie informacji na ekranie – `http://en.cppreference.com/w/cpp/io/cout`.
- Wczytanie informacji z klawiatury – `http://en.cppreference.com/w/cpp/io/cin`.
- Typy proste: **bool**, **char**, **short**, **int**, **long**, **float**, **double** – `http://www.cplusplus.com/doc/tutorial/variables/`, `http://en.cppreference.com/w/cpp/language/types`.
- Instrukcje warunkowe — `http://www.cplusplus.com/doc/tutorial/control/`
  - **if**–**else** — `http://en.cppreference.com/w/cpp/language/if`,

  - **switch** — http://en.cppreference.com/w/cpp/language/switch,

- Pętle:
  - **for** — http://en.cppreference.com/w/cpp/language/for.
  - **while** — http://en.cppreference.com/w/cpp/language/while.
  - **do–while** — http://en.cppreference.com/w/cpp/language/do.
- Instrukcje skoku/opuszczenia:
  - **continue** — http://en.cppreference.com/w/cpp/language/continue.
  - **break** — http://en.cppreference.com/w/cpp/language/break.