Podstawy programowania (w języku C++)

Ćwiczenia

Marek Marecki

19 grudnia 2020

Spis treści

Spis listingów

1 Podstawy

```
#include <iostream>

auto main() -> int
{
    std::cout << "Hello, World!\n";
    return 0;
}
```

Listing 1: Hello, World!

1.0.0.1 Hello, World! Zmodyfikuj program z listingu 1 tak żeby wyświetlał twoje imię i nazwisko, lub jakiś inny wybrany tekst.

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

auto ask_user_for_integer(std::string const prompt) -> int
{
    if (not prompt.empty()) {
        std::cout << prompt;
    }
    auto value = std::string{};
    std::getline(std::cin, value);
    return std::stoi(value);
}
```

Listing 2: Hello, World!

1.0.0.2 Dodawanie Wykorzystując funkcję z listingu 2 napisz program, który pobierze od użytkownika dwie liczby i doda je do siebie. Wynik wydrukuj na `std::cout`.

1.0.0.3 Mnożenie Wykorzystując funkcję z listingu 2 napisz program, który pobierze od użytkownika dwie liczby i pomnoży je przez siebie. Wynik wydrukuj na `std::cout`.

1.0.0.4 Większa liczba Wykorzystując funkcję z listingu 2 napisz program, który pobierze od użytkownika dwie liczby i wydrukuje większą z nich. Wynik wydrukuj na `std::cout`.

1.0.0.5 Wartość absolutna Napisz program, który pobierze od użytkownika liczbę i poda jej wartość absolutną. Wynik wydrukuj na `std::cout`.

```
./program 2 2
2 == 2
./program 0 3
0 < 3
./program 1 -1
1 > -1
```

Listing 3: relacja między liczbami

1.0.0.6 Relacja między liczbami Wykorzystując funkcję z listingu 2 napisz program, który pobierze od użytkownika dwie liczby i wydrukuje relację między nimi tak jak na listingu 3. Wynik wydrukuj na `std::cout`.

1.0.0.7 Dodatnia-nieujemna-ujemna Napisz program, który pobierze od użytkownika liczbę i poda następujący wynik:

1. `1` jeśli liczba jest dodatnia
2. `0` jeśli liczba jest zerem
3. `-1` jeśli liczba jest ujemna

Wynik wydrukuj na `std::cout`.

1.0.0.8 Największa Napisz program, który pobierze od użytkownika trzy liczby i wydrukuje największą. Wynik wydrukuj na `std::cout`.

2 Pętle

2.0.0.1 Lista liczb Napisz program, który pobierze od użytkownika dwie liczby (`a` i `b`), a następnie wydrukuje listę liczb większych lub równych `a` i mniejszych od `b`. Wynik wydrukuj na `std::cout`.

2.0.0.2 Lista liczb (2) Rozwiń program z poprzedniego zadania tak żeby pobierał trzecią liczbę (`c`) i drukował jedynie liczby podzielne przez `c`. Upewnij się, że program odrzuci `c` równe 0. Wynik wydrukuj na `std::cout`.

2.0.0.3 Lista liczb (3) Rozwiń program z zadania 2.0.0.1 tak żeby pobierał liczbę `s` i użył jej jako kroku pętli. Upewnij się, że program działa też dla ujemnej liczby `s`. Upewnij się, że program odrzuci krok o wartości 0. Wynik wydrukuj na `std::cout`.

2.0.0.4 Liczba pierwsza Napisz program, który pobierze od użytkownika liczbę i sprawdzi czy jest ona liczbą pierwszą. Wynik wydrukuj na `std::cout`.

2.0.0.5 Suma liczb pierwszych Napisz program, który pobierze od użytkownika liczbę i sprawdzi czy jest ona liczbą pierwszą. Jeśli tak, to niech poda sumę liczb pierwszych mniejszych lub równych podanej liczbie. Wynik wydrukuj na `std::cout`.

```
./program 2 2 0 3 8 -1
2 == 2
2 > 3
2 < 3
2 < 8
2 > -1
```

Listing 4: relacja między liczbami (2)

2.0.0.6 Relacja między liczbami (2) Rozwiń program z zadania 1.0.0.6 tak, żeby porównywał więcej liczb naraz, tak jak na listingu 4. Wynik wydrukuj na `std::cout`.

2.0.0.7 Suma podzielnych Napisz program, który pobierze od użytkownika dwie liczby: limit i dzielnik. Niech program obliczy sumę wszystkich liczb większych od zera, ale mniejszych lub równych *limitowi*, które są podzielne przez *dzielnik*. Wynik wydrukuj na `std::cout`.

```
for (auto i = 0; i < 42; ++i) {
    // do something
}
```

Listing 5: pętla for

```
auto i = 0;
while (i < 42) {
    // do something
    ++i;
}
```

Listing 6: pętla while

```
auto i = 0;
do {
    // do something
    ++i;
} while (i < 42);
```

Listing 7: pętla do-while

2.0.0.8 Silnia (for) Wykorzystując pętlę `for` (patrz listing 5) napisz program, który pobierze od użytkownika liczbę i obliczy jej silnię. Wynik wydrukuj na `std::cout`.

2.0.0.9 Silnia (while) Wykorzystując pętlę `while` (patrz listing 6) napisz program, który pobierze od użytkownika liczbę i obliczy jej silnię. Wynik wydrukuj na `std::cout`.

2.0.0.10 Silnia (do-while) Wykorzystując pętlę `do-while` (patrz listing 7) napisz program, który pobierze od użytkownika liczbę i obliczy jej silnię. Wynik wydrukuj na `std::cout`.

```
./program-prostokat 2 4
****
****
```

Listing 8: prostokąt z gwiazdek

```
./program-odwrocony-trojkat 4
*
**
***
****
```

Listing 9: trójkąt gwiazdek

```
./program-odwrocony-trojkat 4
****
***
**
*
```

Listing 10: odwrócony trójkąt gwiazdek

2.0.0.11 Rysowanie figury (prostokąt) Wykorzystując dowolną pętlę napisz program, który pobierze z wiersza poleceń wymiary prostokąta i narysuje go. Wynik wydrukuj na `std::cout`. Przykładowe uruchomienie na listingu 8.

2.0.0.12 Rysowanie figury (trójkąt) Wykorzystując dowolną pętlę napisz program, który pobierze z wiersza poleceń wymiary trójkąta i narysuje go. Wynik wydrukuj na `std::cout`. Przykładowe uruchomienie na listingu 9.

2.0.0.13 Rysowanie figury (odwrócony trójkąt) Wykorzystując dowolną pętlę napisz program, który pobierze z wiersza poleceń wymiary „odwróconego trójkąta” (tj. niech wierzchołek będzie na dole, patrz listing 10) i narysuje go. Wynik wydrukuj na `std::cout`.

```
./program-pusty-kwadrat 4
****
*  *
*  *
****
```

Listing 11: pusty kwardat

2.0.0.14 Rysowanie figury (pusty kwadrat) Napisz program, który pobierze z wiersza poleceń wymiary figury, a potem narysuje „pusty kwardat" (patrz listing 11). Wymiar nie może być mniejszy niż 3. Wynik wydrukuj na `std::cout`.

3 Wskaźniki

```
auto x  = int{42};  // an integer
auto xp = &x;       // a pointer to an integer (new style)
int* xo = &x;       // a pointer to an integer (old style)
```

Listing 12: pobranie wskaźnika

```
auto x = int{42};
auto xp = &x;   // xp contains the address of the x variable
auto xd = *xp;  // xd contains a copy of the variable to which
                // xp is pointing

*xp = 64;       // x contains 64 after this assignment
```

Listing 13: dereferencja wskaźnika

3.0.0.1 Pobranie wskaźnika Napisz program, w którym w funkcji `main()` utworzysz zmienną typu `std::string`, której wartością będzie `Hello, World!`. Pobierz wskaźnik i wydrukuj adres tej zmiennej w pamięci.
Wynik wydrukuj na `std::cout`.

3.0.0.2 Dereferencja Napisz funkcję `print()`, która będzie jako parametr przyjmować wskaźnik na `std::string`. W funkcji `print()` wydrukuj adres, na który wskazuje wskaźnik oraz napis stojący za tym wskaźnikiem, np. „`1781f89a980 = Hello, World!`”. W funkcji `main()` napisz kod, który wywołuje funkcję `print()`. Wynik wydrukuj na `std::cout`.

```
./program-zamiana
42 64
64 42
```

Listing 14: zamiana

3.0.0.3 Zamiana Napisz funkcję `swap()`, która będzie jako parametr przyjmować dwa wskaźniki na `int`.

W funkcji `main()` napisz kod, który wywołuje funkcję `swap()`. Wydrukuj wartość dwóch testowych liczb przed i po zamianie (patrz listing 14.

Wynik wydrukuj na `std::cout`.