C++

- Język C++ jest rozszerzeniem języka C o techniki
  - programowania obiektowego
  - programowania uogólnionego
  - w pewnym stopniu również programowania funkcyjnego

- Jest stosowany między innymi
  - w Microsoft (m.in. Windows XX, Office)
  - w Google (search engine)
  - w Amazon.com (całe oprogramowanie e−commerce)
  - w Sun (HotSpot Java Virtual Machine)
  - we wszystkich najważniejszych produktach Adobe
  - w Maya, wykorzystanym m.in. przy produkcji filmów Star Wars Episode I, Spider−Man, Lord of the Rings, Stuart Little
  - przy realizacji projektów Apple, AT&T, CERN, Ericsson, HP, IBM, Intel, NASA, Nokia, SGI, Siemens,

C++ jest jednym z najczęściej wykorzystywanych języków kompilowanych ogólnego przeznaczenia.

Programowanie orientowane zdarzeniami

- programowanie obiektów $->$ pierwsze graficzne interfejsy użytkownika i graficzne systemy operacyjne
- roproszony charakter danych wejściowych generowanych przez mysz $->$ inne podejście do przetwarzania danych
- użytkownik posługując się myszką i klawiaturą generuje zdarzenia
- programowanie sprowadzało się do dodania funkcji do wybranych klas, opisujących sposób reakcji obiektów tych klas na zaistniałe zdarzenia.
- technika projektowania obiektowego doskonale nadaje się do takiego podejścia do programowania
- zdarzenia są reprezentowane jako obiekty pewnych klas
- **programowanie orientowane zdarzeniami**: technika programowania oparta na obsłudze zdarzeń

Programowanie orientowane zdarzeniami

Programowanie orientowane zdarzeniami zazwyczaj nie jest wspierane bezpośrednio przez język programowania, ale przez bibliotekę implementującą stosowną hierarchię zdarzeń

Programowanie uogólnione (generyczne) 50

Ada, a króko po niej język C++ umożliwiły stosowanie techniki programowania uogólnionego (generycznego).

Programowanie uogólnione to technika programowania, w której typ danych traktowany jest jako dopuszczalny parametr podprogramu.

- W języku C++ programowanie uogólnione realizowane jest poprzez t.zw. **szablony**.
- Programowanie uogólnione pozwala na tworzenie uniwersalnego, bardzo efektywnego kodu na wysokim poziomie abstrakcji

Java

- lat 90te XX wieku, Green Team, SUN
- oparty na wybranych cechach C++
- zorientowany bardziej na produktywność niż efektywność
- język kompilowano−interpretowany.
- łatwa przenaszalność oprogramowania przy zachowaniu relatywnie wysokiego tempa wykonania.

Programowanie wielowątkowe i rozproszone

Java, jako jeden z pierwszych języków wspiera programowanie wielowątkowe, a poprzez swoje biblioteki również programowanie rozproszone.

Programowanie wielowątkowe i rozproszone to technika programowania, która zakłada rozbicie programu na niezależne wątki, które mogą być wykonywane równolegle przez szereg procesorów (programowanie wielowątkowe), a nawet przez szereg niezależnych komputerów połączonych w sieć (programowanie rozproszone).

C#

- 2000, Andres Hejslberg, Scotta Wilamuth, Microsoft
- główna część przedsięwzięcia .NET (dot−net), które stawia sobie ambitny cel stworzenia uniwersalnego środowiska programistycznego obejmującego szereg narzędzi programistycznych, w szczególności szereg języków programowania, przygotowanych do wzajemnej interakcji na płaszczyźnie Internetu.
- język kompilowano−interpretowany

Języki skryptowe 54

- **Języki skryptowe**, to języki zaprojektowane dla szybkiego i relatywnie łatwego pisania programów o stosunkowo małej złożoności obliczeniowej.
- Pozwala to na zastąpienie cyklu kompilacja—wykonanie bezpośrednią interpretacją programu
- Interpreter, w przeciwieństwie do kompilatora jest w stanie wychwycić i komunikatywnie sygnalizować nie tylko błędy syntaktyczne, programu, ale również błędy pojawiające się w trakcie jego wykonywania

Python 57

- latach 90'tych XX wieku, Guido van Rossum, National Research Institute for Mathematics and Computer Science w Amsterdamie
- wieloparadygmatowy, umożliwiający m.in. programowanie obiektowo orientowane, funkcyjne i uogólnione.
- wcięcia zamiast nawiasów do oznaczania bloków kodu

JavaScript 58

- lata 90'te XX wieku, Brendan Eich, Netscape
- przeznaczony do programowania w środowisku przeglądarki internetowej
- interpreter języka stanowi część przeglądarki
- język obiektowy, o składni w pewnym stopniu przypominającej Javę – stąd jego nieco myląca nazwa (wcześniej LiveWire i LiveScript)

C/C++ 2

- C: Dennis Ritchie, początek 1970 r.
- w pełni strukturalny, bardzo zwięzły i wydajny, bliski asemblera
- C ++: Bjarne Stroustrup, 1986
- głębokie rozszerzenie C w kierunku
  - abstrakcji danych
  - programowania obiektowego
  - programowania uogólnionego

Literatura

- **Jerzy Grębosz, *Opus magnum C++ 11. Programowanie w języku C++***
- Jerzy Grębosz, *Opus magnum C++. Misja w nadprzestrzeń C++14/17. Tom 4*
- Jerzy Grębosz, *Pasja C++*
- ***Jerzy Grębosz, https://ifj.edu.pl/private/grebosz/ (m.in. kody źródłowe z książek)***
- Brian W.Kernighan. Dennis M. Ritchie*, C language*
- Bjarne Stroustrup*, C++*
- Bruce Eckel*, Thinking in C++*
- Bruce Eckel, Chuck Allison, *Thinking in C++, Volume 2*
- David Vandevoorde, Nicolai M. Josuttis*, C++ Templates*
- Nicolai M. Josuttis, *C++: Object oriented programming*
- Nicolai M. Josuttis, *C++ Biblioteka standardowa. Programmer's Tutorial*

C++ Biblioteka standardowa 6

- wraz ze standardem C++ opracowana została też biblioteka standardowa
  - funkcje matematyczne
  - operacje wejścia-wyjścia
  - obsługa sytuacji wyjątkowych
  - przetwarzanie tekstu
  - pojemniki standardowe (kolekcje)

W praktyce nawet najprostszy program wymaga korzystania z biblioteki standardowej

Pierwszy program 8

```
#include <iostream>
int main(){
  std::cout << "Welcome to the world of C/C++ !!\n";
}
```

Przykład zapisany jako
przyklad1.cpp

Pierwszy program

- Każdy pełny program w C/C ++ musi zawierać funkcję o nazwie **main**. To miejsce, w którym rozpoczyna się egzekucja.

Pisanie na standardowym wyjściu 10

- W C++ instrukcja pisania na standardowym wyjściu ma postać

```
std::cout << "Trochę tekstu w cudzysłowie\n";
```

- **std**::**cout** - standardowe wyjście (ekran)
- $\backslash n$ - znak specjalny oznaczający nową linię

Zmienne 11

- **Zmienna** - nazwane miejsce w pamięci do przechowywania liczby.
- Zmienne muszą być zadeklarowane przed pierwszym użyciem

```
int number;
int i,j;
float celsjus;
```

- **int** - służy do deklarowania zmiennych przechowujących liczby całkowite
- **float** - służy do deklarowania zmiennych przechowujących liczby zmiennopozycyjne

Czytanie ze standardowego wejścia 12

- W C ++ instrukcja read ma postać

```
std::in >> variable;
```

- **std**::**in** - standardowe wejście (klawiatura)

Czytanie, przechowywanie i zapisywanie liczb

```cpp
#include <iostream>
int main(){
  std::cout << "Enter a number: ";
  int number; // We declare an integer variable
  std::cin >> number;
  std::cout << "Your number is " << number << "\n";
}
```

Przykład zapisany jako przyklad2.cpp
w katalogu Przykłady

Pliki nagłówkowe podprogramów bibliotecznych

- **biblioteczny plik nagłówkowy** : konieczny do korzystania z biblioteki
- dla operacji wejścia/wyjścia w stylu c++:

```
#include <iostream>
```

- dla standardowej biblioteki C :

```
#include <cstdlib>
```

- dla funkcji matematycznych

```
#include <cmath>
```

Przykład ukazujący co potrafią biblioteki
zapisany jako przyklad3.cpp
w katalogu Przykłady

Instrukcja if

```
#include <iostream>
int main(){
  std::cout << "Enter a positive number : ";
  int number;
  std::cin >> number;
  if( number <= 0 ){
    std::cout << ". This is not a positive number. ";
  }
}
```

Przykład zapisany jako przyklad4.cpp
w katalogu Przykłady

Instrukcja if-else

```
#include <iostream>
int main(){
  std::cout << "Enter a positive number : ";
  int number;
  std::cin >> number;
  if( number <= 0 ){
    std::cout << ". This is not a positive number. ";
    std::cout << "Try again\n";
    std::cin >> number;
  }else{
    std::cout << "Thank you\n";
  }
}
```

Przykład zapisany jako
przyklad5.cpp
w katalogu Przykłady

Instrukcja if-else

Wariant instrukcji **else**

```
if(number==0) std::cout << "You entered a zero \n";
else std::cout << "You entered a nonzero number \n";
```

Bloki 18

- **Blok** to ciąg instrukcji zamknięty w nawiasy wąsiaste { i }.

```
int n,m;
std::cout << "Enter nonzero n: ";
cin >> n;
if(n==0){
  std::cout << "You entered zero. Enter n again: ";
  cin >> n;
}else{
  std::cout << "Enter m:\n";
  cin >> m;
}
```

- Blok zwany jest również **instrukcją złożoną**

Nie dajemy średnika po nawiasie } zamykającym blok!

Przykład zapisany jako przyklad6.cpp
w katalogu Przykłady

Błędy w programach komputerowych

- rodzaje błędów
  - **błędy kompilacji**
  - **błędy wykonania**
  - **błędy logiczne**

- błędy kompilacji
  - ostrzeżenia
  - błędy uniemożliwiające dokończenie kompilacji (**Fatal errors**)

Błędy kompilacji

- Zdolności kompilatora do zlokalizowania źródła błędu są ograniczone
- Komunikat wskazuje tylko miejsce zagubienia się kompilatora
- Zwykle jeden błąd generuje wiele komunikatów o błędach

Exercise 0.1. Zlokalizuj i popraw błędy kompilacji w następującym programie:

-  WelcomeWithErrors.cpp

Komentarze 22

```
#include <iostream>
int main(){  // begin of the main function
  /* Below we write
  to the standart output */
  std::cout << "Welcome to the world of C/C++ !!\n";
}            // end of the main function
```

Przykład zapisany jako przyklad7.cpp
w katalogu Przykłady

Komentarze

Program bez dobrze dobranych, czytelnych komentarzy bardzo szybko staje się trudny do zrozumienia, a w konsekwencji jego aktualizacja staje się bardzo pracochłonna.

Identyfikatory 24

- **Identyfikator** albo po prostu **nazwa**: używany do nazywania zmiennych, funkcji, typów ....
- Identyfikator w C++ : ciąg składający się z liter i cyfr, zaczynający się od litery
- litery mogą być wielkie lub małe,
- znak podkreślenia jest traktowany jak litera
- przykłady: **x**, **alpha1**, **red_car**, **numberOfCars**
- niektóre identyfikatory są zdefiniowane w bibliotece standardowej. Na przykład **std** or **cout**

Przykład dobrego zapisywania programu
zapisany jako przyklad8.cpp
w katalogu Przykłady

Formatowanie programów

- Formatowanie służy do zwiększenia czytelności programu przez człowieka
- **białe znaki** – spacja, nowa linia, tabulator etc.
- Kompilator potrzebuje białych znaków tylko do oddzielenia identyfikatorów.

```
#include <iostream.h>
using namespace std;
int main () {cout << "Nasz pierwszy program"; }
```

Daje to nam swobodę używania białych znaków do zwiększenia czytelności programu na wiele różnych sposobów.

Słowa kluczowe w C++

- **using**, **namespace**, **int** to przykłady słów kluczowych.
- **Słowo kluczowe**: identyfikator o specjalnym zastrzeżonym znaczeniu
- Nie można go używać w żadnym innym znaczeniu. W szczególności nie można go użyć do nazwania zmiennej
- To samo słowo kluczowe może mieć różne znaczenie w zależności od kontekstu.

Słowa kluczowe w C++

and	and_eq	asm	auto	bitand
bitor	bool	break	case	catch
char	class	compl	const	const_cast
continue	default	delete	for	double
dynamic_cast	else	enum	explicit	export
extern	false	float	for	friend
goto	if	inline	int	long
mutable	namespace	new	not	not_eq
operator	or	or_eq	private	protected
public	register	reinterpret_cast	return	short
signed	sizeof	static	static_cast	struct
switch	template	this	throw	true
try	typedef	typeid	typename	union
unsigned	using	virtual	void	volatile
wchar_t	while	xor	xor_eq	