

Kurs

C++

Cena szkolenia

Cena spotkania grupowego wynosi 100 zł netto za 60 min. Cena spotkania indywidualnego wynosi 120 zł netto za 60 min. Ilość godzin szkolenia zależy od Twoich postępów w nauce oraz ilości czasu, którą poświęcisz na realizację zadań i projektów zleconych w ramach szkolenia.

Opis kursu

Szkolenie przygotowuje Cię do pracy na stanowisku C++ Developer lub każdym innym, gdzie wymagana jest znajomość języka C++. Po zakończeniu szkolenia będziesz w stanie tworzyć i rozwijać nowoczesne aplikacje w języku C++ oraz frameworkach, które z nimi współpracują.

Naukę zaczniesz od zrozumienia podstaw języka C++ i struktur programistycznych, które w nim występują. Nauczysz się zagadnień związanych z algorytmiką oraz zasad optymalizacji kodu i szybkości działania aplikacji. Wykorzystasz komunikację z bazami danych SQL oraz nauczysz się zarządzać danymi w formacie JSON i nie tylko. W aplikacjach zaimplementujesz mechanizmy wielowątkowości. Programy przetestujesz z wykorzystaniem bibliotek dedykowanych do testów. To oczywiście nie wszystko. Poznasz wiele innych narzędzi wspomagających pracę z opisanymi wyżej technologiami (np. Git). Twoja wiedza będzie systematycznie poszerzana o dodatkowe materiały dydaktyczne, które znajdziesz na moich kanałach YouTube, TikTok lub otrzymasz w formie cotygodniowych newsletterów.

Opanowanie zagadnień pojawiających się w szkoleniu zagwarantuje Ci możliwość dalszego rozwijania się w kierunku tworzenia aplikacji do zarządzania systemami wbudowanymi, aplikacji mobilnych lub gier komputerowych.

W ramach szkolenia rozwiążesz ogromną ilość zadań teoretycznych i praktycznych, które skutecznie przygotują Cię do rozmowy kwalifikacyjnej. Twoja wiedza będzie systematycznie sprawdzana w trakcie naszych spotkań i organizowanych co pewien czas indywidualnych próbnych rozmów kwalifikacyjnych w języku polskim lub angielskim.

Po zakończeniu szkolenia Twoje portfolio powiększy się o kilka lub kilkanaście rozbudowanych projektów, które potwierdzą Twoje praktyczne umiejętności posługiwania się zagadnieniami, pojawiającymi się w kursie. Na każdym etapie szkolenia możesz liczyć na nieograniczoną pomoc mentora. Zagwarantuje to zgodność projektów z przyjętymi założeniami oraz dobrymi praktykami programistycznymi, jak również zmusi Cię do systematycznej pracy nad projektami. Dodatkowo zmierzysz się z zadaniami rozwijania lub modyfikowania istniejących już projektów oraz ich analizy pod kątem wyszukiwania błędów.

Potrzebne aplikacje i narzędzia

Do odbycia szkolenia potrzebujesz komputer lub laptop z zestawem narzędzi do budowania aplikacji pojawiających się w trakcie kursu. Spotkania możemy odbywać również na moim komputerze. Po spotkaniu zawsze otrzymasz wszystkie materiały, które pojawiły się w trakcie spotkania. Potrzebne oprogramowanie szczegółowo opiszę oraz pomogę Ci zainstalować na pierwszym darmowym spotkaniu testowym.

Informacje na temat aplikacji potrzebnych do nawiązania połączenia przekażę w odpowiednim momencie przed rozpoczęciem szkolenia. Dostaniesz wtedy szczegółowy wykaz kolejnych kroków, które należy wykonać w celu przygotowania się do rozpoczęcia kursu.

Umiejętności potrzebne przed rozpoczęciem kursu

Kurs przeznaczony jest dla osób o różnym stopniu zaawansowania. Możesz do niego przystąpić nawet jeżeli do tej pory nie miałeś nic wspólnego z programowaniem. Jeżeli już pracujesz na podobnym stanowisku, szkolenie również jest dla Ciebie. Celem kursu jest przygotowanie do pierwszej pracy programisty, ale również podnoszenie kwalifikacji osób już pracujących w branży IT. Szkolenie możesz rozpocząć od dowolnego punktu w planie szkolenia. Na początek odbędziemy szczegółową rozmowę kwalifikacyjną w języku polskim lub angielskim. Dzięki niej ocenimy Twój poziom z zakresu wiedzy przed wybranym punktem w planie szkolenia.

Plan szkolenia

1. Zagadnienia podstawowe

- Konfiguracja i uruchomienie środowiska programistycznego
- Zmienne
- Zmienne inline
- Podstawowe oraz złożone typy danych
- Inicjalizacja zmiennych
- Słowo kluczowe auto
- Stałe – specyfikator const oraz constexpr
- Literały binarne i separatory dziesiętne
- Literały definiowane przez użytkownika
- Literały UTF-8 oraz heksadecymalne liczby zmiennoprzecinkowe
- Stała nullptr
- Specyfikator volatile
- Typ void
- Zakres ważności oraz czas życia zmiennej
- Przesłanianie nazw zmiennych
- Tworzenie dodatkowej nazwy typu – using oraz typedef
- Typ wyliczeniowy enum oraz enum class
- Określanie typu zadanego wyrażenia – decltype
- Zastosowanie słowa kluczowego alignas
- Omówienie pojęć lwartość oraz rwartość
- Pobieranie oraz prezentacja danych
- Operatory arytmetyczne, logiczne, bitowe
- Operatory przypisania
- Operatory uzyskiwania adresu
- Inne operatory: sizeof, noexcept, static_assert, alignof
- Rzutowanie static_cast, const_cast, dynamic_cast
- Rzutowanie reinterpret_cast
- Instrukcje sterujące oraz pętle

- Pętla for dla zakresów
- Instrukcje if oraz switch z inicjalizacją zmiennych
- Instrukcja constexpr if
- Instrukcje break oraz continue
- Tablice jednowymiarowe i wielowymiarowe
- Zagadnienia uzupełniające

2. Wskaźniki i referencje

- Odwoływanie się do adresu zmiennych
- Definiowanie wskaźników oraz referencji
- Praca ze zmiennymi wskaźnikowymi i referencyjnymi
- Stałe wskaźniki i wskaźniki do stałych
- Definiowanie wskaźnika z użyciem słowa kluczowego auto
- Operator rzutowania reinterpret_cast w pracy ze wskaźnikami
- Wskaźnik void*
- Rezerwacja obszarów pamięci z wykorzystaniem wskaźników
- Zagadnienia uzupełniające

3. Funkcje

- Definicja funkcji i jej wywołanie
- Argumenty funkcji oraz wynik zwracany przez funkcję
- Automatyczne określanie typu zwracanego przez funkcję
- Stos i sarta
- Sposoby przesyłania argumentów do funkcji
- Argumenty domniemane i nienazwane
- Funkcje inline
- Funkcje a podział na pliki
- Rekurencja
- Omówienie wybranych funkcji bibliotecznych
- Funkcje constexpr
- Przeładowanie funkcji

- Wskaźniki do funkcji
- Zagadnienia uzupełniające

4. Napisy

- Napisy w stylu C
- Wprowadzenie do klasy `std::string`
- Omówienie zasady działania napisów `std::string`
- Standardowe operatory w pracy z napisami
- Prezentacja funkcji bibliotecznych klasy `std::string`
- Konwersje pomiędzy dowolnym typem a obiektem `std::string`
- Klasy `std::ostringstream` oraz `std::istringstream`
- Wyrażenia regularne
- Zagadnienia uzupełniające

5. Algorytmika

- Algorytmy badające właściwości geometryczne
- Algorytm badające właściwości matematyczne
- Konwersje pomiędzy systemami liczbowymi
- Badanie ciągów danych pod kątem wybranych właściwości
- Sortowanie ciągów danych
- Zastosowanie metody dziel i zwyciężaj
- Wybrane metody numeryczne
- Programowanie zachłanne
- Algorytmy na tekstach
- Wybrane algorytmy kryptograficzne
- Rekurencja
- Przegląd wybranych struktur danych
- Zagadnienia uzupełniające

6. Programowanie obiektowe

- Definiowanie klasy i obiektu
- Struktury
- Różnica pomiędzy klasą a strukturą
- Omówienie założeń enkapsulacji
- Pola składowe klasy
- Proste funkcje składowe klasy
- Funkcje składowe typu `const`, `volatile` oraz `constexpr`
- Funkcje składowe ze specyfikatorami `default` oraz `delete`
- Słowo kluczowe `this`
- Modyfikatory dostępu
- Klasa a dynamiczny przydział pamięci
- Konstruktory
- Słowo kluczowe `explicit`
- Lista inicjalizacyjna a lista `std::initializer_list`
- Konstruktory `constexpr`
- Wyjaśnienie pojęć `glwartość`, `xwartość` oraz `prwartość`
- Konstruktory kopiujące i przenoszące
- Destruktory
- Składniki statyczne w klasie
- Słowo kluczowe `mutable`
- Klasa a podział na pliki
- Operatory konwersji
- Deklaracja przyjaźni
- Przeładowanie operatorów
- Kompozycja
- Klasy zagnieżdżone
- Tablice obiektów
- Zastosowanie wskaźników do pracy z klasami
- Klasy POD (Plain Old Data)
- Zagadnienia uzupełniające

7. Dziedziczenie

- Omówienie zasad dziedziczenia
- Klasa podstawowa i klasa pochodna
- Dostęp do składników klasy podstawowej
- Konstruktory w dziedziczeniu
- Kolejność wywołania konstruktorów w dziedziczeniu
- Wirtualne funkcje składowe
- Polimorfizm
- Wczesne i późne wiązanie
- Destruktry w dziedziczeniu
- Słowa kluczowe final oraz override
- Klasy final
- Klasy abstrakcyjne
- Dynamiczna identyfikacja typu
- Rzutowanie dynamic_cast a polimorfizm
- Wielodziedziczenie
- Zagadnienia uzupełniające

8. Obsługa wyjątków

- Sposoby przechwytywania i obsługi sytuacji wyjątkowych
- Zagnieżdżanie bloków try
- Specyfikator noexcept oraz operator noexcept
- Wyjątki std::uncaught_exceptions
- Zagadnienia uzupełniające

9. Szablony

- Omówienie założeń programowania uogólnionego
- Definiowanie szablonu klasy
- Definiowanie szablonu funkcji
- Dopuszczalne parametry szablonów
- Szablony a podział na pliki

- Funkcje składowe w szablonach klas
- Składniki statyczne w szablonach klas
- Implementacja obiektu klasy szablonowej za pomocą new
- Przeładowanie operatorów w szablonach klas
- Zagnieżdżanie szablonów
- Deklaracja przyjaźni w szablonach klas
- Specjalizacje szablonów klas
- Szablony zmiennych
- Atrybut auto jako parametr szablonu
- Zagadnienia uzupełniające

10. Wyrażenia lambda

- Funktory wbudowane i definiowane przez programistę
- Wyrażenia lambda i jego różne postaci
- Lista argumentów
- Ciało wyrażenia lambda
- Lista wychwytywania
- Słowo kluczowe mutable w wyrażeniu lambda
- Wyjątki w wyrażeniach lambda
- Wyrażenie lambda zastosowane w funkcji składowej
- Szablon `std::function`
- Wyrażenie lambda jako domniemana wartość argumentu
- Zagadnienia uzupełniające

11. Wzorce projektowe

- Klasyfikacja wzorców projektowych
- Implementacja wzorców kreacyjnych
- Implementacja wzorców strukturalnych
- Implementacja wzorców czynnościowych
- Zagadnienia uzupełniające

12. Biblioteka STL

- Wzorzec projektowy iterator
- Wprowadzenie do kontenerów
- Klasyfikacja kontenerów
- Kontenery sekwencyjne (`std::vector`, `std::list`, `std::deque`)
- Kontenery asocjacyjne (`std::set`, `std::multiset`, `std::map`, `std::multimap`)
- Kontenery asocjacyjne z haszowaniem (`std::unordered_set`, `std::unordered_map`)
- Adaptery kontenerów (`std::stack`, `std::queue`, `std::priority_queue`)
- Typy krotkowe `std::tuple`
- Typ `std::pair`
- Omówienie algorytmów STL
- Wskaźniki inteligentne
- Zastosowanie wskaźników inteligentnych do pracy z kontenerami
- Zagadnienia uzupełniające

13. Zarządzanie danymi w formacie JSON

- Omówienie formatu JSON
- Przegląd bibliotek do pracy z formatem JSON
- Konfiguracja aplikacji do pracy z formatem JSON
- Proste przykłady konwersji danych
- Wykorzystanie formatu JSON do pracy z komponentami biblioteki STL
- Zastosowanie formatu JSON w aplikacjach komercyjnych
- Zagadnienia uzupełniające

14. Komunikacja z relacyjnymi bazami danych

- Wprowadzenie do relacyjnych baz danych
- Język SQL
- Omówienie architektury aplikacji bazodanowej
- Przegląd bibliotek do pracy z relacyjnymi bazami danych
- Konfiguracja aplikacji do pracy z relacyjnymi bazami danych
- Implementacja relacji bazodanowych
- Zarządzanie danymi relacyjnej bazy danych
- Implementacja kompletnej aplikacji bazodanowej
- Zagadnienia uzupełniające

15. Wielowątkowość

- Omówienie założeń wielowątkowości
- Różne sposoby tworzenia wątków
- Dołączanie i odłączanie wątków
- Przekazywanie argumentów do wątków
- Omówienie zagadnienia wyścigów
- Sposoby współdzielenia danych pomiędzy wątkami
- Rozwiązanie problemu wyścigów z wykorzystaniem `std::mutex`
- Wykorzystanie szablonu `std::lock_guard`
- Wprowadzenie do zagadnienia przechwytywania zdarzeń w aplikacjach wielowątkowych
- Zastosowanie `condition variables` do przechwytywania zdarzeń
- Przykłady zastosowań typów `std::future` oraz `std::promise`
- Programowanie asynchroniczne z wykorzystaniem `std::async()`
- Przykład zastosowania szablonu `std::packaged_task`
- Zagadnienia uzupełniające

16. Zagadnienia dodatkowe

- Preprocesor
- Unie i pola bitowe
- Operacje wejścia / wyjścia
- Operacje wejścia / wyjścia na plikach
- Semantyka przenoszenia
- Rozszerzony mechanizm generowania liczb losowych
- Atrybut deprecated
- Przestrzenie nazw i zagnieżdżone przestrzenie nazw
- Nowe typy C++17: `std::string_view`, `std::byte`, `std::optional`, `std::variant`, `std::any`
- Wykorzystanie debuggera do analizy pracy programu

17. Framework Qt

- Architektura Qt
- Wzorzec MVC
- Przegląd oraz implementacja komponentów graficznych Qt
- Obsługa zdarzeń
- Zarządzanie rozkładem
- Aplikacje o wielu oknach / zakładkach
- Implementacja nawigacji pomiędzy oknami / zakładkami
- Parsowanie XML / JSON
- Wyrażenia regularne z wykorzystaniem bibliotek Qt
- Walidacja danych
- Architektura aplikacji bazodanowej
- Język SQL
- Implementacja kompletnej aplikacji bazodanowej z wykorzystaniem frameworka Qt
- Zagadnienia uzupełniające

18. Aplikacje Qt Quick

- Wprowadzenie do aplikacji Qt Quick
- Język QML
- Przegląd oraz implementacja kontrolek Qt Quick
- Mechanizm sygnałów oraz slotów
- Wykorzystanie JavaScript w pracy z Qt Quick
- Pozycjonowanie kontrolek Qt Quick
- Dialogi
- Modele i delegaty
- Animacje
- Transformacje i tranzycje
- Programowanie sieciowe
- Komunikacja z REST API
- Komunikacja z bazą danych
- Zagadnienia uzupełniające

19. Zarządzanie projektami – GIT

- Omówienie architektury GIT
- Przegląd instrukcji GIT
- Rozgałęzianie i scalanie w GIT
- Aplikacje i wtyczki do pracy z GIT
- Praca ze zdalnym repozytorium
- GitHub Pages
- Zagadnienia uzupełniające