

## Program szkolenia:

# C++ STL - efektywne wykorzystanie i najlepsze praktyki

## Informacje:

<b>Nazwa:</b>	<b>C++ STL - efektywne wykorzystanie i najlepsze praktyki</b>
<b>Kod:</b>	<b>ccpp-C++ STL</b>
<b>Kategoria:</b>	C i C++
<b>Odbiorcy:</b>	developerzy
<b>Czas trwania:</b>	3 dni
<b>Forma:</b>	35% wykłady / 65% warsztaty

---

Szkolenie poświęcone standardowej bibliotece C++ (STL), które pozwala zrozumieć od podszewki moc możliwości dostarczanych przez STL. W nieksiążkowy i nieszablonowy sposób, bazując na użytecznych przykładach kolejno odsłaniane są rozmaite elementy biblioteki standardowej z mocnym akcentem na wydajność.

Szkolenie przeznaczone jest dla programistów znających podstawową składnię C++ chcących w płynny i efektywny sposób wykorzystywać wszystkie mechanizmy dostarczane przez standard tego języka oraz otwartych bibliotek.

## Zalety szkolenia:

- Praktyczne podejście do nauki STL
- Nacisk na wydajność i elastyczność kodu
- Rzeczywiste przykłady

## Szczegółowy program:

### 1. OOP – Object Oriented Programming – programowanie obiektowe

#### 1.1. Paradygmat programowania obiektowego

1.1.1. Analiza paradygmatu programowania obiektowego i jego poprawna interpretacja

1.1.2. GRASP – General Responsibility Assignment Software Patterns (Principles).

1.1.3. SOLID – Single Responsibility Principle (SRP), the Open/Closed Principle (OCP), the Liskov Substitution Principle (LSP), the Dependency Inversion Principle (DIP), and the Interface Segregation Principle (ISP).

### 2. Korzenie języka C++

#### 2.1. Język C

#### 2.2. Niskopoziomowa natura C++

#### 2.3. Typy złożone po raz pierwszy

2.3.1. Klasa a struktura

2.3.2. Klasa/struktura a obiekt

2.3.3. Operatory

#### 2.4. Kopia, wskaźnik i referencja

2.4.1. Wysokopoziomowe spojrzenie na pamięć

2.4.2. Problem zarządzania pamięcią

2.4.3. Wsparcie STL w tej tematyce - inteligentne wskaźniki i obiekty "udające" referencje

### 3. Łańcuchy znaków

#### 3.1. Podstawy - czyli jak to było w C

#### 3.2. `std::string` - własności i możliwości

#### 3.3. boost - gotowe rozwiązania budowania oraz przetwarzania łańcuchów znaków

### 4. Kontenery i algorytmy

#### 4.1. Przegląd kontenerów z uchwyceniem różnic funkcjonalnych i wydajnościowych

4.2. Przetwarzanie kontenerów (i nie tylko) z użyciem standardowych algorytmów

4.3. Efektywne połączenie standardowych kontenerów i algorytmów oraz biblioteki boost

4.4. Problematyka alokatora

## 5. IO - czyli obsługa strumieni

5.1. Czym są strumienie i jakie są ich rodzaje

5.2. Przeciążanie odpowiednich operatorów

5.3. Strumienie w STL

5.3.1. Obsługa standardowego IO

5.3.2. Bufory strumieniowe

5.3.3. Pliki

## 6. C++11

6.1. Zarządzanie pamięcią - dynamiczne alokowanie

6.2. Generator liczb pseudolosowych

6.3. Tuning std::string

6.4. Nowe kontenery i algorytmy oraz wykorzystanie wyrażeń lambda

6.5. Wielowątkowość w C++11