Z czego uczyć się C++'a w 2018 roku?

Materiałów do nauki języka C++ jest mnóstwo, zarówno polskich, jak i anglojęzycznych. Nie jest trudno znaleźć rekomendację, czy od kolegi, czy biorąc pod lupę najpopularniejsze kursy/poradniki/książki, ale znaczna większość osób które zaczynają dopiero z programowaniem nie bierze pod uwagi jednej, bardzo ważnej rzeczy - jakości.

Jakość powinna być głównym czynnikiem (tuż obok grupy docelowej dla której dany kurs jest przeznaczony) na podstawie którego chcemy wybrać kurs. I z przykrością stwierdzam że najbardziej polecane w Polsce kursy (przez losowe osoby) mają niewiele wspólnego z jakością.

W tym artykule przedstawię moją, subiektywną charakterystykę dobrego kursu C++ w roku 2018, materiały na podstawie których warto się aktualnie uczyć języka oraz takie których należy unikać.

Charakterystyka dobrego kursu

Tutaj należy rozdzielić temat na dwie części - materiały dla początkującego powinny być inaczej napisane, niż te przeznaczone dla osób kontynuujących naukę. Początkująca osoba musi wyrobić sobie pewien sposób myślenia, który pozwoli jej kontynuować na własną rękę. Jeśli taki efekt nie został osiągnięty po ukończeniu kursu, wtedy zawiodły materiały lub dana osoba odpowiednio się nie przyłożyła do nauki (opcjonalnie, programowanie lub dany język nie jest dla niej).

Kurs dla początkującego powinien:

- **Być napisany prostym i zrozumiałym językiem.** Osoba która nie miała wcześniej styczności z programowaniem nie zrozumie typowo programistycznych terminów
- **Przedstawiać uproszczone koncepcje, bez wprowadzania w błąd.** Łatwo jest coś uprościć, ale równie łatwo jest to zrobić do takiego stopnia, że powstałe uproszczenie wprowadza w błąd. Takich sytuacji należy zdecydowanie unikać, ponieważ znacznie ciężej jest oduczyć się czegoś błędnego i nauczyć tego ponownie, niż rozszerzać definicję danej koncepcji.
- **Nie wprowadzać rzeczy które w danym etapie kursu nie są wymagane.** Nawiązując do poprzedniego podpunktu, nie powinniśmy wprowadzać czegoś co w danej części kursu nie jest ważne, ponieważ narzut informacji może spowodować trudności w jej przyswojeniu. Lepiej przedstawić najpierw absolutne podstawy, a potem wprowadzać dodatkowe elementy.
- **Zawierać opisy i przykłady znacznie dłuższe niż definicje.** Nawiązując do pierwszego podpunktu, nie wolno oczekiwać że newbie zrozumie coś na podstawie definicji. Każda rzecz która zostaje wprowadzona powinna zostać dość wyczerpująco (i nie przesadnie, nikt nie lubi czytać o tej samej rzeczy 3 razy) wyjaśniona, żeby mieć pewność że czytelnik ją rozumie. Dodatkowe przykłady są również wyjątkowo ważne i powinny iść w parze z opisem.
- **Nie mieszać języka C z C++.** C++ w absolutnie żadnym wypadku nie powinien być traktowany jako rozszerzenie C, jedyne co ma wspólnego z nim to składnia i wsteczna kompatybilność C++'a z C (w dużym stopniu). Są to dwa diametralnie różne od siebie języki. Uczenie C (używanie praktyk z niego oraz bibliotek przeznaczonych pod ten język) w kursie poświęconym C++owi bez żadnego wspomnienia o tym dyskwalifikuje kurs z miejsca.

Powyższych podpunktów (może oprócz czwartego) nie musi spełniać kurs dla kontynuujących naukę, lub zorientowany pod specyficzne rozwiązania i elementy języka, ale zawsze jest milej jeśli jest przystosowany dla

ludzi, a nie tylko dla programistów.

Dodatkowo, każdy dobry kurs powinien

- **Być jak najbardziej aktualny w momencie tworzenia go.** Moim zdaniem, wprowadzanie przestarzałych rozwiązań bez konkretnych powodów (a takie mogą się pojawić, na przykład brak wsparcia kompilatora dla danego rozwiązania w danym momencie) jest co najmniej niepoprawne. Jeśli już coś tworzymy, postarajmy się pisać o rzeczach rekomendowanych oraz stosunkowo nowych.
- **Nie wprowadzać w błąd.** Na to narażeni są głównie newbies, którzy w 99% nie będą weryfikować poprawności materiałów tylko ślepo im uwierzą. Popełnianie jawnych błędów jest złe, nie poprawianie ich po dostaniu feedbacku jest jeszcze gorsze. Kardynalne błędy merytoryczne dyskwalifikują materiały pod względem jakościowym. Rzecz jasna, każdy popełnia błędy - nie mówię tu o ekstremiźmie w postaci dyskwalifikacji na podstawie niedomówienia, literówki czy też niedopatrzenia autora. Mówię tutaj na przykład o nadmiernym upraszczaniu do momentu w którym zmieniamy sens danej rzeczy, lub rekomendowaniu przestarzałych/niepoprawnych rozwiązań. Oduczenie się niepoprawnego rozwiązania i nauczenie się go od nowa, w poprawny sposób skutecznie hamuje rozwój.
- **Być aktualizowany.** Oczywiście nie wszystko da się aktualizować, niektóre rozwiązania nie będą się zmieniać przez lata, ale jeśli piszemy coś co w pewnym momencie otrzymuje aktualizację, warto wydać poprawkę lub nową wersję materiałów z jej opisem.
- **Brać pod uwagę feedback czytających.** To jest raczej oczywiste, przykładowo jeśli ludzie nie rozumieją pewnej części, lub chcą dokładniejszego opisu, należy wziąć to pod uwagę.

Materiały do nauki

Czas na meritum tego artykułu. Poniżej przedstawiam listę moich ulubionych materiałów dotyczących języka C++

Książki

Książka jest najlepszym źródłem informacji, jeśli zaczynamy z programowaniem w danym języku lub potrzebujemy rozległego źródła informacji

Dla początkujących

- **Bjarne Stroustrup - Programowanie. Teoria i praktyka z wykorzystaniem C++ *(oryginalny tytuł: Programming: Principles and Practice Using C++)*, 2 edycja (2014).** Przyznam się szczerze - akurat tej pozycji (jeszcze) nie przeczytałem (a powinienem!), ale jestem w stanie z czystym sumieniem ją polecić. Książka jednego z twórców języka C++ stanowi genialne wprowadzenie zarówno do programowania jak i samego języka, bardzo dokładnie go opisując (włącznie ze standardami C++11 oraz C++14). Nie ogranicza czytelnika do samego działania w konsoli, ponieważ zawiera kilka rozdziałów przeznaczonych bibliotece FLTK (międzyplatformowej biblioteki pozwalającej tworzyć w prosty sposób GUI). Prowadzi nas od podstaw języka, przez podstawowe biblioteki, do bardziej zaawansowanych elementów STLa (Standard Template Library) oraz algorytmów i kontenerów.

- **Bjarne Stroustrup - Język C++, Kompendium Wiedzy *(oryginalny tytuł: The C++ Programming Language)*, wydanie 4 (2013).** Tą pozycję przerobiłem i absolutnie ją polecam osobom które miały wcześniej jakikolwiek kontakt z programowaniem. Jest to trochę bardziej technicznie napisana książka, która skupia się tylko i wyłącznie na samym języku stanowiąc jednocześnie jedno z najdokładniejszych źródeł wiedzy o nim. Minimalnym mankamentem jest brak opisu standardu C++14, ale nie był on tak

rewolucyjny jak C++11, więc można to tej książce wybaczyć. Zawiera pełny opis języka, najważniejszych części biblioteki standardowej oraz technik i idiomów stosowanych w tym języku. Osoby które są kompletnie zielone w programowaniu nie powinny koniecznie sięgać po tą książkę jako pierwszą, ponieważ techniczny język może zniechęcić - ale jeśli ktoś taki woli, to jest to idealna pozycja.

- **Stanley Lippman, Josée Lajoie i Barbara E. Moo - C++ Primer, wydanie 5 (2012).** Niedostępna niestety w języku polskim (na tyle wykazało moje googlowanie, ale nie dam sobie ręki uciąć) gorąco polecana przez środowisko Stack Overflow książka, która *"jest bardzo szczegółowym wprowadzeniem do języka C++, która oferuje przystępne i szczegółowe wyjaśnienie wszystkiego co znajduje się w tym języku"*. Jeśli ktoś umie czytać w miarę płynnie po angielsku i nie przypadną mu do gustu powyższe pozycje, to warto po nią sięgnąć.

Dla kontynuujących naukę, chcących podszlifować swoje umiejętności

- **Scott Meyers - C++. 50 efektywnych sposobów na udoskonalenie Twoich programów *(oryginalny tytuł: Effective C++)* (2008).** Troszkę stara książka, która mimo swojego wieku jest nadal aktualna. Przedstawia 55 praktyk które pomogą nam tworzyć elastyczniejszy i wydajniejszy kod. Warto ją przeczytać po przerobieniu dobrego kursu podstaw.
- **Scott Meyers - Skuteczny nowoczesny C++. 42 sposoby lepszego posługiwania się językami C++11 i C++14 *(oryginalny tytuł: Effective Modern C++)* (2014).** Druga, nowsza wersja powyższej książki która skupia się na elementach nowych standardów które można zastosować żeby nasz kod był lepszy. Tak jak wyżej, polecam każdemu kto przerobił kurs podstaw języka.

Jest ogromna ilość książek które są przeznaczone pod konkretne elementy języka, ja przedstawiłem krótką listę tych najbardziej ogólnych. Pełną listę na której bazowałem można znaleźć tutaj, w tym legendarnym wątku na StackOverflow.

Strony i kursy internetowe

Tutaj dla początkujących mam tylko jedną pozycję, pozostałe będą suplementami, referencjami oraz poradnikami dotyczącymi języka. I od razu uprzedzam - wszystkie pozycje będą po angielsku. Nie znam w tej chwili żadnego ukończonego i dobrego kursu C++ po polsku, a stron poświęconych temu językowi w języku polskim nie przeglądam na tyle żeby je tu świadomie polecać.

- **LearnCpp.com.** Najlepszy darmowy i ogólnodostępny kurs języka C++ jaki w tym momencie znam. Bardzo dokładnie opisuje elementy języka oraz techniki w nim stosowane, jak również podstawy STLa. Świetna pozycja na start, żeby przekonać się czy ten język danej osobie się podoba.
- **C++ Reference.** Najlepsze referencje C++ jakie można znaleźć w internecie (lepsze są chyba tylko dokumenty ISO ze standardami, ale żeby je czytać trzeba być cyborgiem). Podstawowe źródło wiedzy na temat języka oraz biblioteki standardowej. Must-know dla każdego programisty C++.
- **C++ Super-FAQ.** Bardzo dobrze napisane FAQ języka C++ które odpowiada na ogromną ilość pytań i niejasności związanych z językiem oraz dobrymi praktykami.
- **C++ Core Guidelines.** Prowadzone przez (między innymi) Bjarne Stroustrup'a guideline'y, BARDZO szczegółowo opisujące zarówno sam język jak i wiele praktyk stosowanych w nim (lub takich, które powinno się stosować)
- **CppCon.** YouTube'owy kanał na którym zamieszczane są nagrania z konferencji CppCon, na której poruszane są przeróżne tematy dotyczące C++a.

- **Bjarne Stroustrup's C++ Style and Technique FAQ.** FAQ stworzone przez Bjarne Stroustrupa dotyczące stylu języka C++ oraz technik w nim wykorzystywanych. Warto przeczytać przy migracji z innego języka.

Materiały których należy unikać

Jak już wcześniej wspomniałem - nie znam żadnych materiałów w języku polskim które mógłbym w tej chwili z czystym sumieniem polecić, ale znam materiały które zdecydowanie mogę odradzić.

- **Mirosław Zelent - Kurs C++ (2013).** Popularność prawopodobnie zawdzięcza formie prowadzenia kursu, bo merytoryki w tym nie ma prawie żadnej. Kilka pierwszych odcinków jest *"znośnych"* (w bardzo dużym cudzysłowie), ale im dalej się w to zagłębiamy tym jest gorzej. Główne grzechy tego kursu to:
    1. **Mieszanie C z C++em.** Nie wiem czy jest to świadomy zabieg, wiem że nigdzie o tym nie mówi. Po prostu miesza techniki z C i C++a, nie patrząc na poprawność merytoryczną.
    2. **Posługiwanie się bibliotekami zależnymi od systemu.** Pal licho *Windows.h*, *conio.h* to biblioteka z MS-DOSu! Tak, rozumiem - kurs przeznaczony dla początkujących, ale czy warto wrzucać takie biblioteki i uczyć ich używania tylko po to żeby mieć *fajny* efekt kliknięcia przycisku bez blokowania w terminalu lub zatrzymywać (na co jest odpowiednia funkcja w standardzie, o której oczywiście nie ma słowa w całym kursie) program na chwilę? Moim zdaniem absolutnie nie warto.
    3. **Praktycznie zerowe omówienie STLa.** Ani słowa o kontenerach czy podstawowych algorytmach, co jest w tym momencie absolutną podstawą przy programowaniu w języku C++.
    4. **Posługiwanie się przestarzałymi rozwiązaniami i standardami.** Mam dziwne wrażenie że nie ma tam wykorzystanego niczego co pojawiło się w (aktualnie obowiązkowym do nauczenia się) standardzie C++11, nie wspominając nawet o C++14 ani C++17. Kurs powstał w roku 2013. Dwa lata po wydaniu C++11. Nie wiem co powoduje że pan Zelent nie wspomniał o tym słowem.
    5. **Brak chęci poprawienia kursów.** Słyszałem wiele opinii na temat Zelenta, słyszałem również że on sam miał z nimi styczność i zrobił dokładnie nic żeby to poprawić. Kurs jak był, tak jest fatalnej jakości.
    6. **Nauka przestarzałych i niepoprawnych technik.** takich jak nadmierne używanie wskaźników oraz ręcznej alokacji pamięci lub nieużywanie kontenerów ze standardu. Gdybym dokładnie obejrzał cały kurs to prawdopodobnie znalazłbym takich przykładów o wieeele więcej
    7. **Używanie przestarzałych kompilatorów i narzędzi.** Uczenie tworzenia GUI przy pomocy Borland C++ Buildera w wersji 6, wydanego w roku 2002. W roku 2018. Rozumiałbym jeszcze używanie nowszej wersji, wybaczyłbym gdyby kurs był merytoryczny. Ale nie jest.
    8. **Nudna forma kursu.** Może to tylko moje subiektywne odczucie, ale tłumaczenie tej samej rzeczy trzykrotnie, przeciągając temat, nie trafia do mnie w żaden sposób.
- **Cpp0x.** Ten kurs był swego czasu dobry, nie przeczę, ale patrząc na niego w tej chwili wydaje mi się że kolejność rzeczy w nim jest troszkę toporna, oraz brakuje zintegrowanych w kursie nowości (które de facto pojawiają się, ale na koniec, jako osobne rozdziały). Mimo wszystko, nie polecałbym.
- **Karol Kuczmarski - Kurs C++ (aka. Od zera do gier kodera) (2004).** Data wydania tej książki wiele mówi nam o tym jaki język C++ będzie tam opisywany. Na ten moment, zdecydowanie przestarzała. Znacznie lepiej jest sięgnąć po pozycję Stroustrupa, jeśli chcemy mieć grubszą lekturę.
- **Jerzy Grębosz - Symfonia C++ (1993).** Tak jak wyżej - WYJĄTKOWO przestarzała. Nie jestem w stanie się obiektywnie wypowiedzieć na temat Opus Magnum, w którym podobno jest C++11 (a powinno być i C++14 biorąc pod uwagę datę wydania, ale z jakiegoś powodu nie ma), ale jestem nieprzekonany. Stroustrup lepszy.

- **Stephen Prata - Język C++. Szkoła programowania. (oryginalny tytuł: C++ Primer Plus) (2013)**. Nie polecam jej z dwóch powodów - nie ma zbyt dobrych opinii (patrz tutaj), oraz widziałem jaki kod piszą osoby które się z niej uczyły. Nie warto.

Dodatkowo, polecam unikać kursów które

- **Mają słabe opinie w internecie** - głównie należy patrzeć na merytorykę i dowiedzieć się dlaczego ludzie nie polecają danej książki/kursu, zamiast patrzeć na to dlaczego go polecają. Jest duża szansa że ktoś kto go nie poleca wie o czym mówi, a ktoś kto go poleca jeszcze nie wie o czym mówi (patrz: efekt Krugera-Dunninga)
- **Oferują naukę w pewnym określonym czasie (C++ OD ZERA DO PRO W 30 DNI!!!11one)** - nie. To tak nie działa. Szczególnie z C++em, gdzie trzeba spędzić kilka miesięcy i pokonać dość stromą krzywą nauki żeby zacząć w nim pisać cokolwiek rozsądnego.
- **Są stosunkowo stare** - szczególnie należy na to zwracać uwagę jeśli dane materiały powstały przed rokiem 2011 i nie ma do nich aktualizacji. Warto wtedy poszukać czegoś nowszego.
- **Nie mówią o bibliotece standardowej** - To straszna zmora wielu kursów słabej jakości, zamiast wprowadzić udogodnienia, powodują że krzywa uczenia się jest jeszcze bardziej stroma

Podsumowanie

Na koniec mam kilka porad których chętnie udzieliłbym sobie, gdybym mógł, w momencie zaczynania z C++em

- **Zastanów się czy ten język jest dla ciebie** - *obviously*, ja go polubiłem i w końcu, po wielu mękach, umiem go na tyle żeby się nim płynnie posługiwać. Ale jest wiele osób które usłyszały o tym języku i uznały że chcą się go nauczyć "bo tak", lub "bo słyszały że jest dobry na start". To, że jest dobry na start osobiście demonizuję, bo moim zdaniem nie jest - jeśli ktoś jest kompletnie zielony w temacie to lepiej sięgnąć po coś znacznie prostszego i przekonać się czy programowanie jest dla niego (ja zaczynałem od kochanego Autolta ❤️), a potem wkroczyć w miarę potrzeby w inne języki. Zdaję sobie sprawę że w szkołach cały czas nauczyciele próbują (bo znaczącej większości to nie wychodzi) uczyć tego języka, więc osobom które mają informatykę (lub co gorsza poszły na studia informatyczne bez absolutnie żadnych zapędów do programowania) i z przymusu się go uczą serdecznie współczuję i sugeruję brać Pythona na maturę z informatyki kiedy tylko stanie się to możliwe.
- **Nie ucz się ze słabych źródeł** - to był mój główny problem podczas nauki. Na początku przerobiłem całe cpp0x, które wtedy jeszcze nie było rozszerzone o rzeczy z C++11, więc mój stan wiedzy był dość słaby. Przez wiele miesięcy próbowałem coś pisać, z marnym skutkiem, ponieważ brak znajomości znaczącej części języka oraz trudności ze znajdowaniem materiałów na moim poziomie powodowały że nie mogłem się rozwijać w pełnym tempie. Na dodatek, oduczenie się złych praktyk zajęło mi drugie tyle czasu, co nauczenie sie ich, więc finalnie zmarnowałem ogromną ilość czasu na nauce metodą prób i błędów.
- **Przeczytaj książkę na start** - tak naprawdę dopiero kilka lat po rozpoczęciu nauki języka sięgnąłem po pierwszą dobrą książkę (C++ Programming Language) i w miarę jak ją czytałem, uświadamiałem sobie jak mało znałem ten język. Przeczytanie dobrej książki na start jest najlepszym co można zrobić.
- **Nie daj się złapać za mocno efektowi Dunninga-Krugera** - Nie wiem ile razy byłem na szczycie tej krzywej, ale na pewno za dużo. Przy nauce programowania, na początku trzeba być świadomym dwóch rzeczy. Po pierwsze - **zawsze znajdzie się ktoś lepszy od ciebie**. Po drugie - **nic nie umiesz i nie oczekuj że to szybko się zmieni**, bo nawet jeśli wydaje ci się że coś umiesz, to tak naprawdę tego nie umiesz jeśli nie potrafisz tego efektywnie zastosować w praktyce. Jeśli nie uświadomisz sobie że czegoś

nie umiesz, tylko od razu po poznaniu danej rzeczy uznasz "aaa, ale ja to umiem, olać" to nigdy się tego nie nauczysz. I po trzecie, **nie da się nauczyć czegoś w 100%, więc musisz cały czas się uczyć**. To może być trochę przesadzone, ale tak to wygląda w 90% przypadków - nigdy nie wolno przestać się uczyć, bo żeby poznać i zostać absolutnym ekspertem języka C++ trzeba by było prawdopodobnie spędzić dziesiątki lat na nauce i praktyce. Nawet Bjarne Stroustrup nie uważa że zna ten język w pełni.

I na tym kończę ten artykuł. Mam nadzieję że pomoże wielu osobom które chcą wejść w świat języka C++, jak i również tym które starają się w nim odnaleźć. Jeśli chcesz mnie wesprzeć oraz zobaczyć więcej takich artykułów, możesz rzucić mi kilka złotych tutaj (PayPal). Jeśli chcesz umieścić ten wpis na swojej stronie/blogu/gdziekolwiek, to prosiłbym o uprzedni kontakt i konsultację tego ze mną - wolałbym o tym wiedzieć, niż potem robić komuś problemy. Jeśli chcesz wstawić ten wpis w jakiś wątek lub post, to skopiuj ten link (PPM -> kopiuj adres linku, lub skorzystaj z poniższego pola)

```
https://bit.ly/2lfftNV
```

Dostępna jest również wersja PDF w moim repozytorium.

Wojciech Olech