# POLITECHNIKA POZNAŃSKA

## Wydział Informatyki i Telekomunikacji

**Produkt cyfrowy**
WIiT-Inf-niest-II-zti-sem2 grupa L1

**Wojciech Błażejewski 93299**
**wojciech.blazejewski@student.put.poznan.pl**

**Esej “Ewolucja języków programowania: historia - daty, kluczowe nazwiska. Przełomowe produkty programistyczne”**

**Wrzesień 2023**

Czym jest język programowania? To pytanie wydaje się proste, jednak odpowiedź na nie może być znacznie bardziej złożona, niż byśmy mogli przypuszczać. Języki programowania można rozumieć szerzej niż tylko jako narzędzia do tworzenia oprogramowania dla komputerów. Patrząc na zagadnienie przez pryzmat teorii sterowania to językiem programowania może być każda opracowana metoda i technika, która pozwalają na kontrolowanie zachowań i właściwości w pożądany przez programistę sposób, a więc będą to także sposoby wyrażania i realizowania różnych idei, celów i zadań w różnych dziedzinach wiedzy i życia. Języki programowania pozwalają więc na tworzenie i modyfikowanie sekwencji instrukcji lub danych, które sterują zachowaniem lub funkcjonowaniem pewnego systemu. Systemem tym może być maszyna cyfrowa, ale także organizm żywy, układ elektroniczny, zjawisko fizyczne czy nawet ludzki umysł.

W tym eseju przyjrzymy się bliżej historii i zastosowaniom języków programowania, ale nie tylko w kontekście oprogramowania dla komputerów. Oczywiście, dowiemy się o kluczowych momentach w historii informatyki i programowania, takich jak rola Ady Lovelace czy Johna von Neumanna. Jednak chcielibyśmy także poszerzyć nasze pojęcie o językach programowania o obszary, które mogą wydawać się zupełnie odległe od kodowania.

Przykładem może być Programowanie neurolingwistyczne - zbiór technik komunikacji i pracy z wyobrażeniami, które mają na celu tworzenie i modyfikowanie wzorców postrzegania i myślenia u ludzi. Oczywiście, na pierwszy rzut oka nie wydaje się to mieć wiele wspólnego z tradycyjnym programowaniem. Jednak, jak pokazuje film "Arrival" z 2016 roku, różne tłumaczenia i interpretacje mogą wprowadzać nieporozumienia, tak samo jak błędy w kodzie mogą wprowadzać chaos w działaniu programu. Taką perełką w tym filmie było przetłumaczenie wiadomości obcych jako “broń” z którym nie zgadzała się ściągnięta na miejsce lingwistka tłumacząc, że obcym może chodzić po prostu o przekazanie ludzkości “narzędzia”, które niekoniecznie jest tożsame z bronią.

Podobnie, poznamy obszary takie jak Programowanie DNA - proces tworzenia i modyfikowania kodu genetycznego organizmów żywych za pomocą technik biotechnologicznych, czy Programowanie muzyczne - proces tworzenia i modyfikowania dźwięków i utworów muzycznych za pomocą różnych technik i narzędzi, które mogą przypominać programowanie komputerowe.

Przyjrzymy się także przykładom nieszablonowego myślenia w programowaniu, takim jak w filmach z serii o Bournie, gdzie scenarzyści zainspirowali się prawdziwymi eksperymentami prowadzonymi przez CIA w ramach projektu MKUltra, aby stworzyć intrygującą historię o manipulacji ludzkimi zdolnościami i emocjami.

Warto zrozumieć, że języki programowania mają znacznie szerszy zakres zastosowań, niż mogłoby się wydawać na pierwszy rzut oka. Dlatego w tym eseju eksplorujemy te różnorodne aspekty, aby lepiej zrozumieć, jak języki programowania wpływają na różne sfery naszego życia i działalności.

Większość osób powiedziałaby, że nowożytna historia programowania zacznie się na Adzie Lovelace. Osobiście uważam jednak, że Krosno tkackie, które zostało zaprojektowane w 1805 roku przez Josepha Marie Jacquarda, oparte na metodzie sterowania nitkami za pomocą kart perforowanych wpisuje się w definicje, które podałem we wstępie. Karty perforowane znalazły też zastosowanie dla organów (XIX wiek), a dopiero potem użyto ich do kontrolowania komputerów. Karty perforowane zawierały otwory, które wskazywały, które

dźwięki miały być odtwarzane na organach. Z kronikarskiego obowiązku w kolejnych akapitach chronologicznie wymienione zostaną kluczowe według mnie wydarzenia.

1842 - Ada Lovelace, uważana za pierwszą programistkę w historii, zaproponowała metodę obliczania liczb Bernoulliego na silniku analitycznym Charlesa Babbage'a, który był prototypem komputera mechanicznego. Jej praca jest uznawana za pierwszy program komputerowy na świecie. Publikacja "Sketch of the analytical engine invented by Charles Babbage, Esq" po angielsku nastąpiła rok później w 1843[1]

1945 - John von Neumann zaprezentował koncepcję architektury von Neumanna, która stała się podstawą dla współczesnych komputerów elektronicznych. Zaproponował on, aby program był przechowywany w pamięci komputera w postaci binarnej, co umożliwiło tworzenie uniwersalnych maszyn zdolnych do wykonywania różnych zadań.

"1954-1957 - John Backus i jego zespół w IBM" [2] stworzyli pierwszy język programowania wysokiego poziomu, czyli takiego, który jest bardziej zrozumiały dla ludzi niż dla maszyn. Był to Fortran (Formula Translation), który był przeznaczony do obliczeń naukowych i inżynierskich. Fortran jest nadal używany, zwłaszcza w dziedzinach takich jak fizyka, chemia czy meteorologia.

1959 - Grupa CODASYL (Conference on Data Systems Languages) opracowała język COBOL (Common Business Oriented Language), który był skierowany do zastosowań biznesowych i administracyjnych. COBOL był jednym z najpopularniejszych i najbardziej rozpowszechnionych języków programowania w latach 60. i 70. XX wieku. Jego zaletą była łatwość czytania i zrozumienia kodu przez ludzi, a także przenośność między różnymi systemami operacyjnymi i sprzętem.

1964 - John Kemeny i Thomas Kurtz stworzyli język BASIC (Beginner's All-purpose Symbolic Instruction Code), który miał na celu ułatwić naukę programowania osobom nieposiadającym specjalistycznej wiedzy matematycznej lub informatycznej. BASIC był prosty w użyciu i szybko zyskał popularność wśród hobbystów, studentów i nauczycieli. Był to także jeden z pierwszych języków dostępnych na komputerach osobistych.

1972 - Dennis Ritchie zaprojektował język C, który był następcą języka B stworzonego przez Kena Thompsona. C był językiem niskiego poziomu, bliskim maszynie, ale jednocześnie oferował możliwości abstrakcji i strukturyzacji kodu typowe dla języków wysokiego poziomu. C stał się podstawą dla wielu innych języków programowania, takich jak C++, Java czy C#. Był także użyty do napisania systemu operacyjnego Unix, który miał duży wpływ na rozwój informatyki. Sam napisałem też w nim proste gry na konsole wii.

1980 - Bjarne Stroustrup rozszerzył język C o elementy programowania obiektowego, tworząc język C++. Programowanie obiektowe polega na organizowaniu kodu w postaci klas i obiektów, które reprezentują abstrakcyjne lub konkretne byty i ich właściwości oraz zachowania. C++ jest jednym z najczęściej używanych języków programowania, zwłaszcza w dziedzinach takich jak gry komputerowe, grafika, symulacje czy systemy wbudowane.

1983 - Kary Mullis wynalazł reakcję łańcuchową polimerazy (PCR), co stało się kluczowym narzędziem w biologii molekularnej do kopiowania i amplifikowania fragmentów DNA.

1990 - Rozpoczęcie projektu Human Genome Project, który miał na celu zsekwencjonowanie całego ludzkiego genomu. Projekt ten trwał do roku 2003.

1991 - Guido van Rossum stworzył język Python, który charakteryzuje się prostotą, elegancją i czytelnością kodu, którą wręcz wymusza jego składnia. Python jest językiem interpretowanym, co oznacza, że nie wymaga kompilacji przed uruchomieniem. Python jest językiem wieloparadygmatowym, co oznacza, że umożliwia stosowanie różnych stylów i technik programowania, takich jak programowanie obiektowe, funkcyjne, imperatywne czy deklaratywne. Python jest szeroko stosowany w różnych dziedzinach, takich jak nauka danych, sztuczna inteligencja, web development czy edukacja.

1995 - James Gosling i jego zespół w Sun Microsystems zaprezentowali język Java, który był inspirowany językiem C++, ale uproszczony i ulepszony pod względem bezpieczeństwa, przenośności i wielowątkowości. Java jest językiem kompilowanym do kodu bajtowego, który jest następnie interpretowany przez maszynę wirtualną Javy (JVM), co umożliwia uruchamianie programów na różnych platformach i systemach operacyjnych. Java jest językiem obiektowym, który wprowadził pojęcie klas abstrakcyjnych i interfejsów. Java jest jednym z najpopularniejszych i najbardziej wpływowych języków programowania, używanym w aplikacjach desktopowych, mobilnych, webowych i korporacyjnych.

1995 - Rasmus Lerdorf stworzył język PHP (Hypertext Preprocessor), który był pierwotnie zbiorem skryptów napisanych w C do obsługi formularzy na stronach internetowych. PHP jest językiem interpretowanym, który jest osadzony w kodzie HTML i wykonywany po stronie serwera. PHP jest językiem słabo typowanym, co oznacza, że nie wymaga deklarowania typów zmiennych. PHP jest także językiem obiektowym, choć początkowo nie posiadał tej funkcjonalności. PHP jest jednym z najczęściej używanych języków programowania do tworzenia dynamicznych stron i aplikacji internetowych.

1995 - Brendan Eich stworzył język JavaScript, który był pierwotnie nazywany Mocha, a następnie LiveScript. JavaScript jest językiem interpretowanym, który jest osadzony w kodzie HTML i wykonywany po stronie klienta (przeglądarki internetowej). JavaScript jest językiem słabo typowanym i dynamicznym, co oznacza, że typy zmiennych mogą się zmieniać w trakcie działania programu. JavaScript jest także językiem wieloparadygmatowym, który umożliwia stosowanie programowania obiektowego, funkcyjnego i zdarzeniowego. JavaScript jest nieodłącznym elementem rozwoju stron i aplikacji internetowych, zwłaszcza w połączeniu z technologiami takimi jak HTML, CSS czy AJAX.

1997 - Craig Venter i zespół ogłosili sukces w sekwencjonowaniu ludzkiego genomu, rywalizując z projektem Human Genome Project.

1999 - Układy FPGA (Field Programmable Gate Array) to rodzaj układów scalonych, które można zaprogramować po ich wyprodukowaniu. Oznacza to, że można zmieniać funkcje i logikę realizowaną na poziomie hardware w zależności od potrzeb użytkownika. Układy FPGA składają się z wielu bloków logicznych, które można połączyć ze sobą za pomocą programowalnej macierzy połączeń. Układy FPGA są więc przykładem programowania sprzętu, czyli modyfikowania fizycznej struktury i działania urządzenia elektronicznego już po jego wyprodukowaniu.

2003 - Zakończenie projektu Human Genome Project, który dokładnie zsekwencjonował ludzki genom, identyfikując 20 000-25 000 genów w genomie ludzkim. Można powiedzieć, że jest to wstęp do programowania DNA czyli procesu tworzenia i modyfikowania kodu genetycznego organizmów żywych za pomocą technik biotechnologicznych, takich jak inżynieria genetyczna, edycja genomu czy syntetyczna biologia. Programowanie DNA umożliwia tworzenie zmian w cechach fenotypowych organizmów, takich jak kolor skóry, włosów, oczu, odporność na choroby, inteligencja czy długość życia. Można także tworzyć nowe gatunki lub formy życia, które nie występują w naturze. Programowanie DNA jest więc podobne do programowania komputerowego pod względem używania języka kodowania i algorytmów.

2012 - Ogłoszenie pierwszego sukcesu w inżynierii genetycznej człowieka z użyciem techniki CRISPR-Cas9, co otworzyło nowe możliwości modyfikacji DNA.

2013 - W filmach z serii o Bournie scenarzyści użyli nieszablonowego myślenia do stworzenia fabuły o programowaniu zachowań żołnierzy tajnych służb. Zainspirowali się prawdziwymi eksperymentami prowadzonymi przez CIA w ramach projektu MKUltra, który miał na celu opracowanie metod kontroli umysłu i modyfikacji osobowości. Scenarzyści połączyli elementy psychologii, farmakologii, genetyki i szpiegostwa, aby stworzyć intrygującą historię o sterowaniu ludzkimi zdolnościami i emocjami.

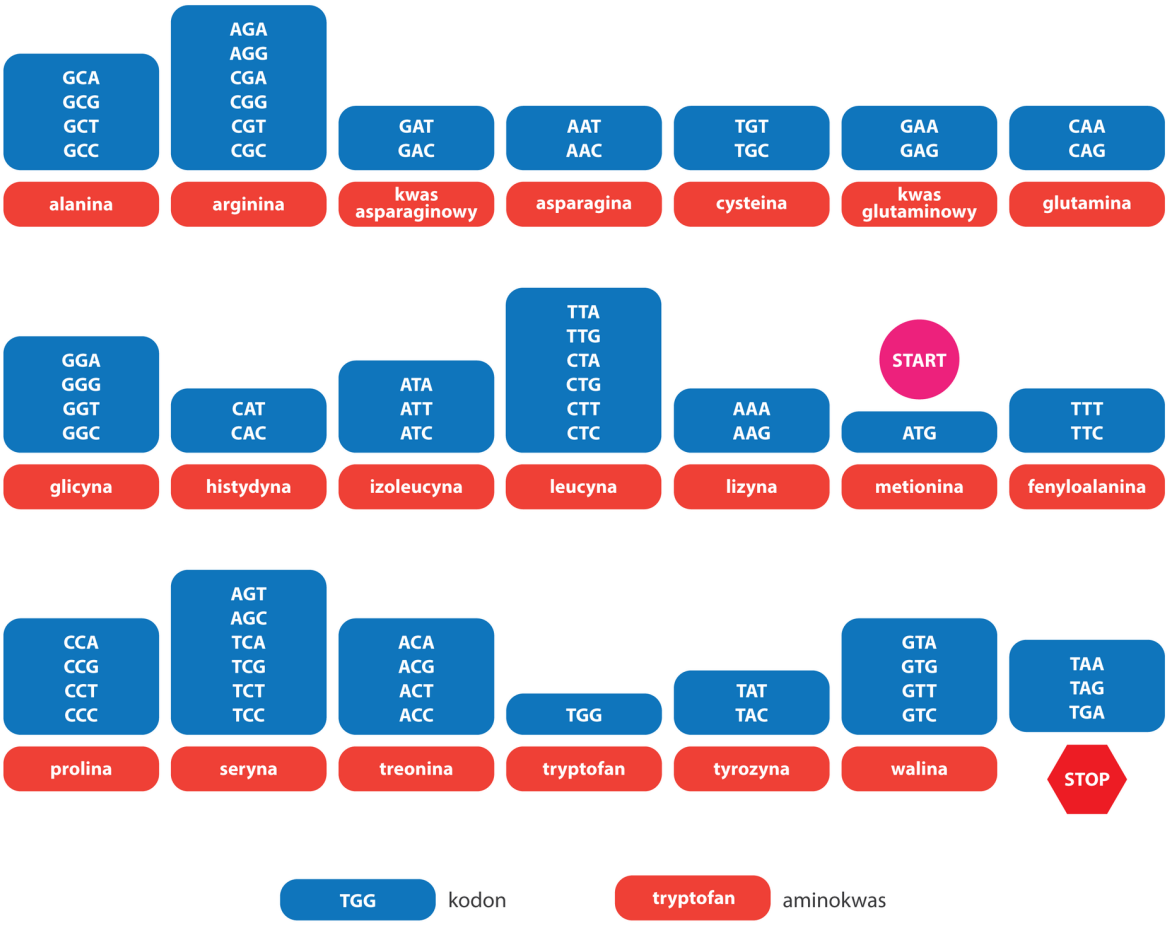
2016 - “zespół Google DeepMind AlphaGo pokonał mistrza świata w grze w Go Lee Sadola. Złożoność starożytnej chińskiej gry była wcześniej uznawana za główne wyzwanie dla sztucznej inteligencji.” [3]

2016 - “firma Hanson Robotics opracowała pierwszego humanoidalnego robota o nazwie Sophia, który był zdolny do rozpoznawania twarzy, komunikacji werbalnej i ekspresji wyrazu twarzy.” [3]

2018 - “firma Waymo uruchomiła usługę Waymo One, która pozwalała użytkownikom znajdującym się na obszarze metropolitalnym miasta Phoenix w stanie Arizona na korzystanie z samochodów autonomicznych.” [3]

2020 - “firma Baidu opracowała algorytm LinearFold AI pomagający zespołom naukowców w pracach nad szczepionką przeciwko wirusowi SARS-CoV-2. Algorytm jest zdolny do przewidywania mutacji sekwencji wirusa RNA w czasie 27 sekund, to znaczy 120 szybciej niż dotychczas stosowane metody.” [3]

To tylko niektóre z wielu języków programowania, które powstały i ewoluowały na przestrzeni lat. Każdy z nich ma swoją historię, zalety i wady, zastosowania i społeczność użytkowników. Nie ma jednego idealnego języka programowania dla wszystkich celów i sytuacji. Na aktualnym etapie naszego rozwoju ważne jest więc, aby poznać różne opcje i wybrać tę, która najlepiej odpowiada naszym potrzebom i preferencjom. Niektórych może zdziwić, że w niektórych akapitach pojawiły się daty wydarzeń na pierwszy rzut oka kompletnie niezwiązane z programowaniem. Pojawiły się one tam celowo aby spojrzeć na zagadnienie szerzej. Czy poniższy rysunek przedstawiający DNA razem ze start i stop nie przypomina kodu programu w asemblerze?




Rysunek 1 — Przykładowy kod DNA [4]

Podsumowując naszą podróż przez historię i różnorodność języków programowania oraz ich nieoczywiste zastosowania, nie możemy zapominać o przyszłości tego fascynującego obszaru. Świat programowania i kompilacji dynamicznie ewoluuje, a jednym z głównych kierunków tej ewolucji jest rosnąca rola sztucznej inteligencji.

AI obecnie przekształca sposób, w jaki piszemy, kompilujemy i optymalizujemy kod. Narzędzia oparte na AI pozwalają na tworzenie kodu w sposób bardziej intuicyjny, automatyczne wykrywanie błędów, optymalizację kodu pod kątem konkretnych platform sprzętowych i wiele innych. Jednym z przełomowych przykładów jest rosnąca popularność GPT-3 i podobnych modeli, które pomagają w generowaniu kodu na podstawie opisów tekstowych, a to tylko wierzchołek góry lodowej.

W przyszłości możemy oczekiwać jeszcze bardziej zaawansowanych narzędzi AI do kompilacji i programowania. Procesy kompilacji będą bardziej inteligentne, dostosowując się do konkretnej architektury sprzętowej, co przyspieszy wydajność programów. Narzędzia do wykrywania błędów będą bardziej skuteczne, a optymalizacje kodu będą bardziej zindywidualizowane, co pozwoli na lepsze dostosowanie aplikacji do potrzeb użytkowników.

Jednak AI nie tylko usprawni techniczne aspekty programowania i kompilacji, ale także otworzy nowe horyzonty. Narzędzia oparte na AI pozwolą na bardziej intuicyjne i dostępne dla każdego podejście do programowania. Sztuczna inteligencja będzie wspierać nas w tłumaczeniu naszych myśli na kod, eliminując bariery językowe i techniczne.

Programowanie stanie się bardziej kreatywne, a także bardziej zintegrowane z innymi dziedzinami wiedzy, jak biologia czy neurologia.

Podsumowując, przyszłość języków programowania i kompilacji jest pełna obietnic, dzięki rosnącej roli sztucznej inteligencji. Kiedy to się stanie, ograniczeniem będzie tylko nasza wyobraźnia. Sztuczna inteligencja oparta o GPT może stać się kompilatorem przyszłości który zastąpi lub będzie miał wbudowaną większość aktualnie znanych języków programowania dzięki czemu programista będzie mógł wydawać polecenia w języku naturalnym. Można więc zaryzykować stwierdzenie, że kompilatory dzięki modelom GPT wyewoluują w uruchamianie programów za pomocą języka naturalnego. Co więcej nie można wykluczyć, że na zapleczu firm takich jak OpenAI już się to dzieje i jest tylko kwestią czasu kiedy ta technologia będzie ogólnodostępna dla każdego. Przełom w tej kwestii już nastąpił, niektóre dostępne wtyczki do dużych modeli językowych dają namiastkę programowania za pomocą języka naturalnego.

Idąc dalej środowisko programistyczne zmierza w kierunku, w którym programowanie za pomocą języka naturalnego, a może nawet myśli staje się realnością. To ekscytujący rozwój, który może znacząco obniżyć bariery wejścia do świata programowania i uczynić go bardziej dostępnym dla różnych grup ludzi.

Na zakończenie nawiążę do odcinków programu starożytnych kosmitów. Co jeżeli mechanizm z Antykithiry rzekomo z czasów Starożytnej Grecji (190–50 r. p.n.e.)[5] nie jest modelem opracowanym na podstawie obserwacji naszego układu słonecznego tylko jest znacznie starszy i został zbudowany jako “proof of concept” - wizualizacja projektu takiego układu słonecznego, a nasz układ słoneczny został na jego podstawie stworzony tak jak najpierw tworzy się projekt budynku a dopiero potem realizuje się jego budowę. Czy programowanie rozwinie się do tego stopnia aby prostym poleceniem-słowem, a może nawet samą myślą kreować świat? Czy opis stworzenia świata, który można przeczytać np. w biblii, już niedługo dzięki rozwojowi technologi będziemy czytać dosłownie? To, że ktoś “powiedział” czyli wydał polecenie i stało się, wcale nie jest niemożliwe. To, że na obecnym poziomie technologicznym nie dysponujemy jeszcze urządzeniem końcowym, które może tworzyć galaktyki, układy słoneczne czy organizmy żywe nie znaczy, że już wcześniej nie użyto tego rodzaju technologii aby stworzyć np. nas.

## Bibliografia

[1] https://pl.wikipedia.org/wiki/Ada_Lovelace

[2] https://pl.wikipedia.org/wiki/Fortran

[3] Fazlagić, J. (red.) (2022). Sztuczna inteligencja (AI) jako megatrend kształtujący edukację. Jak przygotowywać się na szanse i wyzwania społeczno-gospodarcze związane ze sztuczną inteligencją? Warszawa: Instytut Badań Edukacyjnych.

[4] Źródło: Dariusz Adryan, licencja: CC BY 3.0.

[5] https://pl.wikipedia.org/wiki/Mechanizm_z_Antykithiry