Wykład 1: Standardy języka C. Cykl budowania oprogramowania.

dr inż. Andrzej Stafiniak

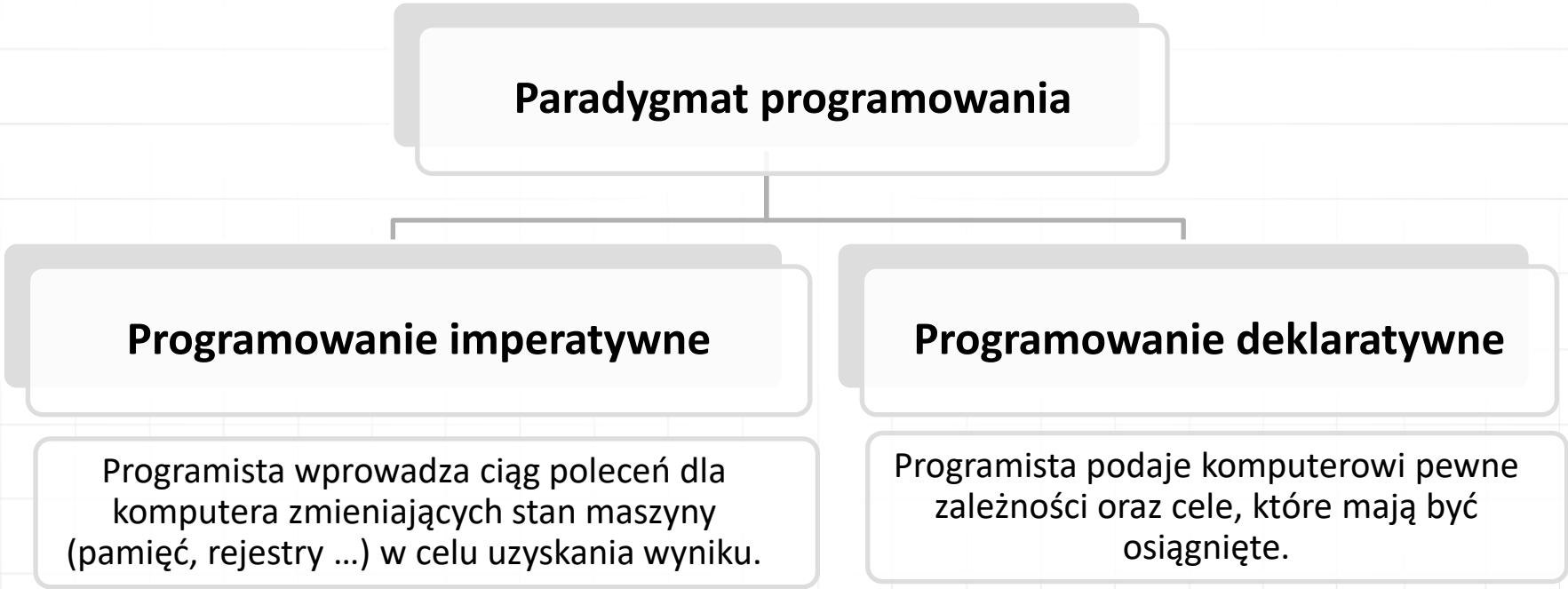
Wrocław 2024

Paradygmat programowania

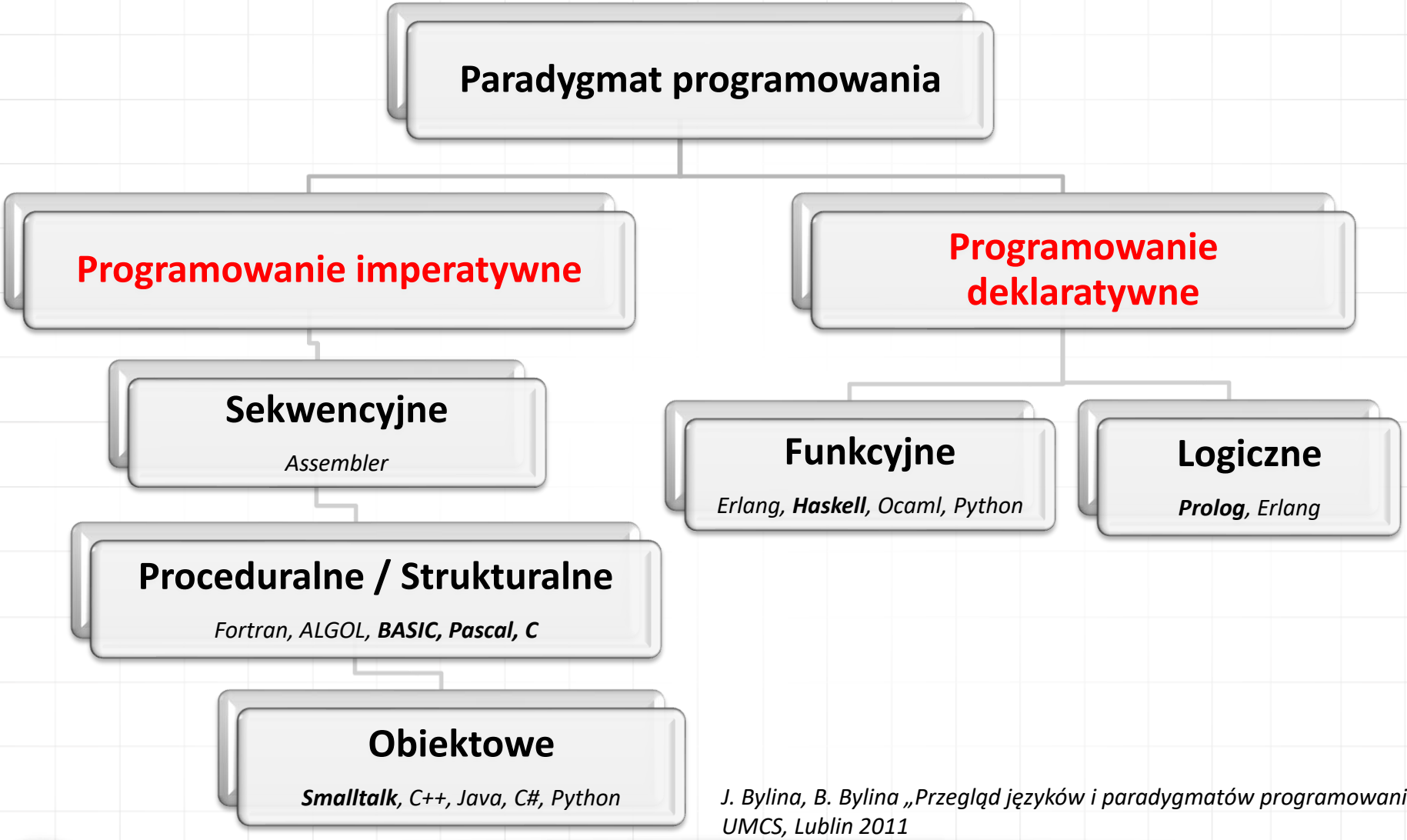
Paradygmat - przyjęty sposób widzenia rzeczywistości w danej dziedzinie, doktrynie.

(*Słownik Języka Polskiego, PWN*)

Paradygmat programowania – zbiór zasad, mechanizmów jakich używa programista pisząc program i w jaki sposób postrzega sterowanie nim oraz jak program jest wykonywany.



Paradygmat programowania



J. Bylina, B. Bylina „Przegląd języków i paradygmatów programowania”, UMCS, Lublin 2011

Paradygmat programowania

Programowanie proceduralne / Programowanie strukturalne

*Fortran, ALGOL, **BASIC, Pascal, C***

- podprogramy (procedury, funkcje …)
- wywoływane z różnymi parametrami, również rekurencyjnie
- programowanie zespołowe
- biblioteki oprogramowania

- sekwencja
- wybór/selekcja
- pętla
- podprogram wywoływany wielokrotnie z różnymi parametrami
- rekurencyjnia

J. Bylina, B. Bylina „Przegląd języków i paradygmatów programowania”, UMCS, Lublin 2011

Paradygmat programowania

Programowanie proceduralne / Programowanie strukturalne

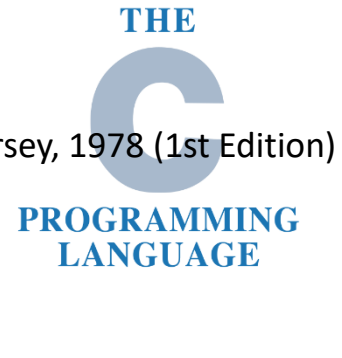
*Fortran, ALGOL, **BASIC, Pascal, C***

Time-line języków programowania wyższego poziomu

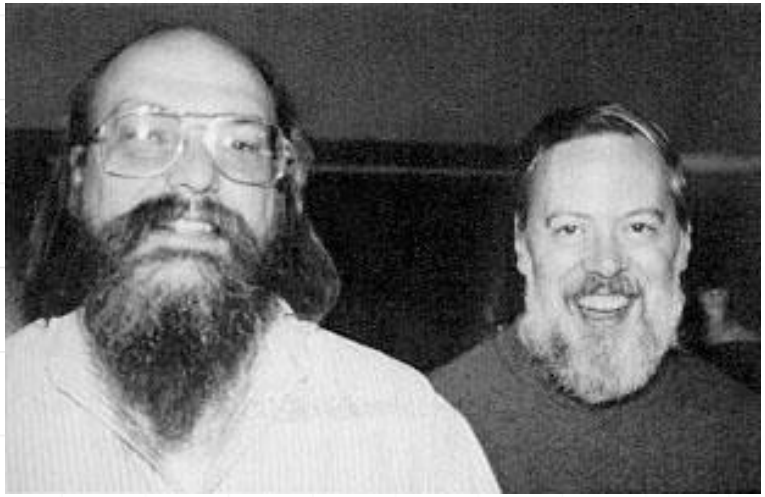
https://wikipedia.org/

The C Programming Language

Brian Kernighan, Dennis Ritchie, *„The C Programming Language”,* Prentice Hall, Hoboken, New Jersey, 1978 (1st Edition)



Bell Labs
Język B -> Język C -> UNIX



Ken Thompson Dennis Ritchie

Kolejne wersje/standardy języka C

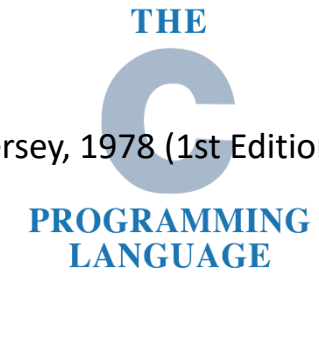
Year	C Standard
1972	Birth
1978	K&R C
1989/1990	ANSI C and ISO C
1999	C99
2011	C11
2017	C17
TBD	C2x

Język C – stworzony dla programistów, celem głównym użyteczność, w opozycji do Pascal (baza do nauczania zasad programowania) i BASIC (podobny do j. angielskiego łatwy do przyswojenia)

https://wikipedia.org/

The C Programming Language

Brian Kernighan, Dennis Ritchie, *„The C Programming Language”,* Prentice Hall, Hoboken, New Jersey, 1978 (1st Edition)



K&R C – (C78)

- kwalifikatory short i long
- specyfikator unsigned

Kolejne wersje/standardy języka C

Year	C Standard
1972	Birth
1978	K&R C
1989/1990	ANSI C and ISO C
1999	C99
2011	C11
2017	C17
TBD	C2x

https://wikipedia.org/

The C Programming Language

Brian Kernighan, Dennis Ritchie, *„The C Programming Language”,* Prentice Hall, Hoboken, New Jersey, 1978 (1st Edition)

K&R C – (C78)

- kwalifikatory short i long
- specyfikator unsigned

ANSI C / ISO C

- typ wyliczeniowy enum
- prototyp funkcji
- void*
- #elif
- biblioteka standardowa języka C

Kolejne wersje/standardy języka C

Year	C Standard
1972	Birth
1978	K&R C
1989/1990	ANSI C and ISO C
1999	C99
2011	C11
2017	C17
TBD	C2x

https://wikipedia.org/

The C++ Programming Language

Bell Labs (79-83)
Język C + klasy -> Język C ++

Bjarne Stroustrup

https://wikipedia.org/

https://www.stroustrup.com/

- Nazwa C++ została zaproponowana w 1983 roku, kiedy to po raz pierwszy użyto tego języka poza laboratorium naukowym.
- Wcześniej używano nazwy „C z klasami”.
- Wysoka efektywność kodu.
- Kompatybilność z językiem C, poprawnie napisany program w języku C będzie poprawnie napisanym programem w C++

The C++ Programming Language

Bell Labs (79-83)
Język C + klasy -> Język C ++

Bjarne Stroustrup

https://wikipedia.org/

https://www.stroustrup.com/

C++ standards

Year	C++ Standard	Informal name
1998	ISO/IEC 14882:1998	C++98
2003	ISO/IEC 14882:2003	C++03
2011	ISO/IEC 14882:2011	C++11, C++0x
2014	ISO/IEC 14882:2014	C++14, C++1y
2017	ISO/IEC 14882:2017	C++17, C++1z
2020	ISO/IEC 14882:2020	C++20, C++2a

https://wikipedia.org/

Język programowania niskiego vs. wysokiego poziomu

- Język programowania **niskiego poziomu** – **?**

Język programowania niskiego vs. wysokiego poziomu

- Język programowania **niskiego poziomu** – **Języki asemblera**
  - wysoki stopień powiązania z hardware'm
  - kontrola programisty nad działaniem procesora, jedno polecenie - jeden rozkaz procesora
  - operują na rejestrach procesora i pośrednio na komórkach pamięci
  - brak abstrakcji takich jak pętle czy złożone struktury danych

Język programowania niskiego vs. wysokiego poziomu

- Język programowania **niskiego poziomu** – **Języki asemblera**
  - wysoki stopień powiązania z hardware'm
  - kontrola programisty nad działaniem procesora, jedno polecenie - jeden rozkaz procesora
  - operują na rejestrach procesora i pośrednio na komórkach pamięci
  - brak abstrakcji takich jak pętle czy złożone struktury danych

- Język programowania **wysokiego poziomu** – ?

Język programowania niskiego vs. wysokiego poziomu

- Język programowania **niskiego poziomu** – **Języki asemblera**
  - wysoki stopień powiązania z hardware'm
  - kontrola programisty nad działaniem procesora, jedno polecenie - jeden rozkaz procesora
  - operują na rejestrach procesora i pośrednio na komórkach pamięci
  - brak abstrakcji takich jak pętle czy złożone struktury danych

- Język programowania **wysokiego poziomu** – np. **Java, Python, Erlang, Ocaml**
  - brak potrzeby wyrażania się w postaci kodu maszynowego czy mnemonikami
  - wykorzystanie abstrakcji programistycznych w celu ułatwienia tworzenia zaawansowanych programów
  - bardziej naturalny i intuicyjny sposób programowania – zbliżony do tego jak myśli człowiek

Język programowania C

- Czy języki programowania **C/C++,** są językami **wysokiego poziomu?**

Język programowania C

- Czy języki programowania **C/C++,** są językami **wysokiego poziomu?**
- Języki **C/C++** to języki **kompilowane.** Czym jest **kompilator?**

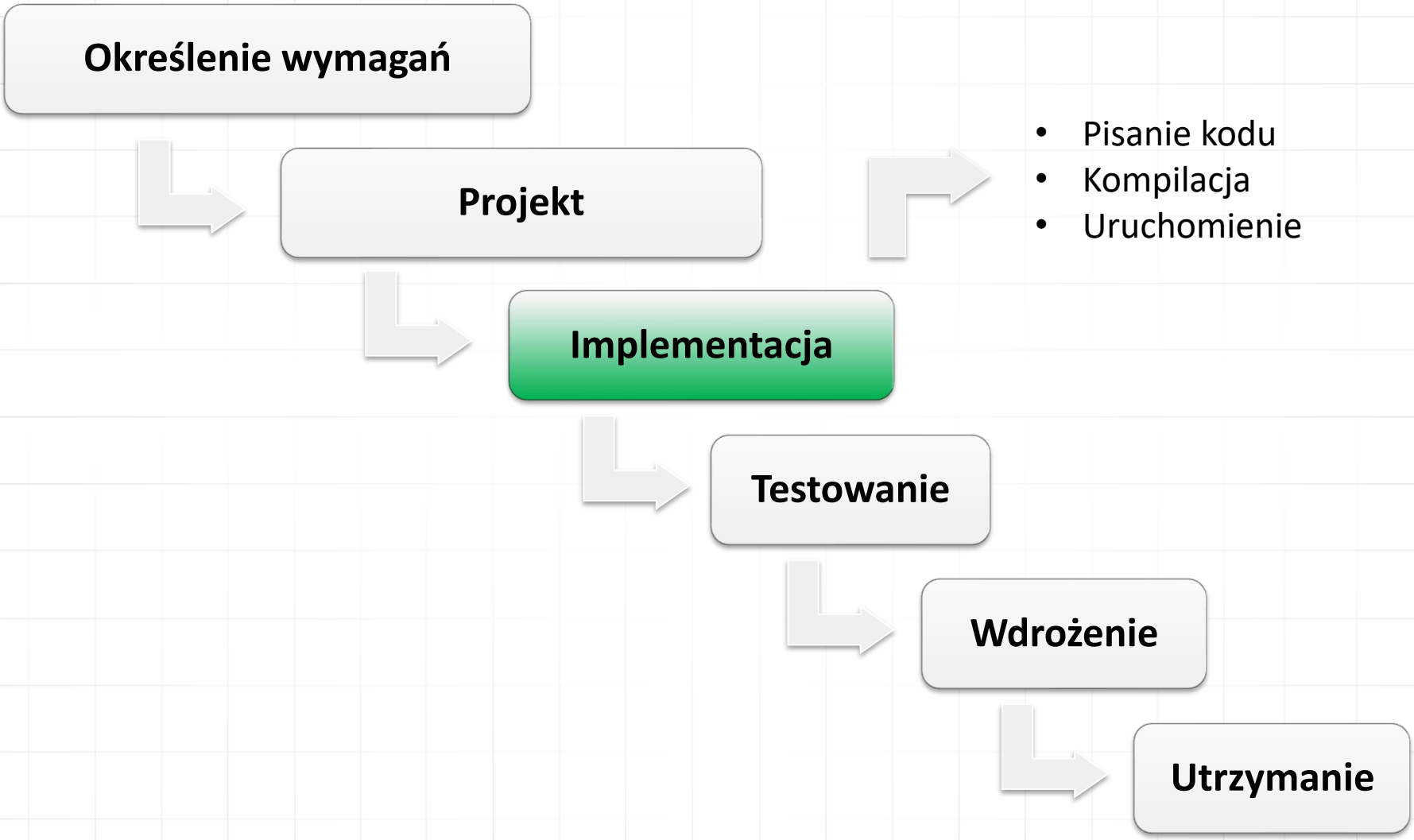
Język programowania C

- Czy języki programowania **C/C++,** są językami **wysokiego poziomu?**
- Języki **C/C++** to języki **kompilowane.** Czym jest **kompilator?**
  - Tłumaczy **instrukcje wysokiego poziomu** na **język maszynowy**.
  - Każdy komputer/procesor może mieć własny **język maszynowy** i to do **kompilatora** należy wybór odpowiedniego języka.
  - Jeden **kod źródłowy** programu napisanego w **języku wysokiego poziomu** -> wiele programów napisanych języku maszynowym na różne urządzenia / systemy / architektury.

Język programowania C

- Czy języki programowania **C/C++,** są językami **wysokiego poziomu?**
- Języki **C/C++** to języki **kompilowane.** Czym jest **kompilator?**
- Czy języki **C/C++** to języki **przenośne?**

Cykl tworzenia oprogramowania



Proces tworzenia pliku wykonywalnego z kodu źródłowego

Preprocesor

Preproceesing

- Usuwanie komentarzy
- Rozwijanie makr
- Dołączanie i rozwijanie plików nagłówkowych i źródłowych (*.h, *.c)

Kompilator

Kompilacja

- Pobiera dane wyjściowe z preprocesora i generuje kod w języku asemblera (specyficzny dla docelowego procesora)
- Pliki *.s, *.asm

Asemblacja

- Konwersja kodu asemblera do kodu maszynowego (binarnego)
- Pliki obiektowe *.o

Asembler

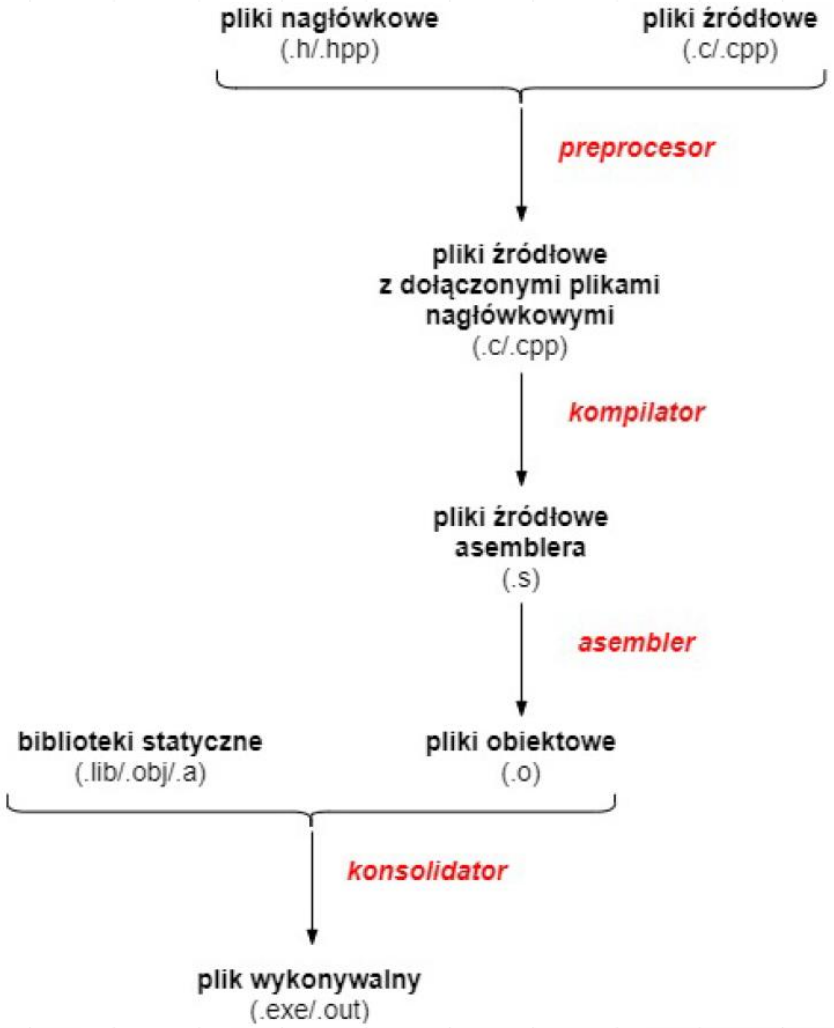
Konsolidacja (Linkowanie)

- Pliki obiektowe + kod bibliotek + kod startowy (interfejs miedzy programem a systemem operacyjnym)
- Scalanie wszystkich plików z kodem maszynowym z różnych modułów w jeden plik wykonywalny (*.exe, *.out)

Linker

Proces tworzenia pliku wykonywalnego z kodu źródłowego

- czyli proces budowania aplikacji



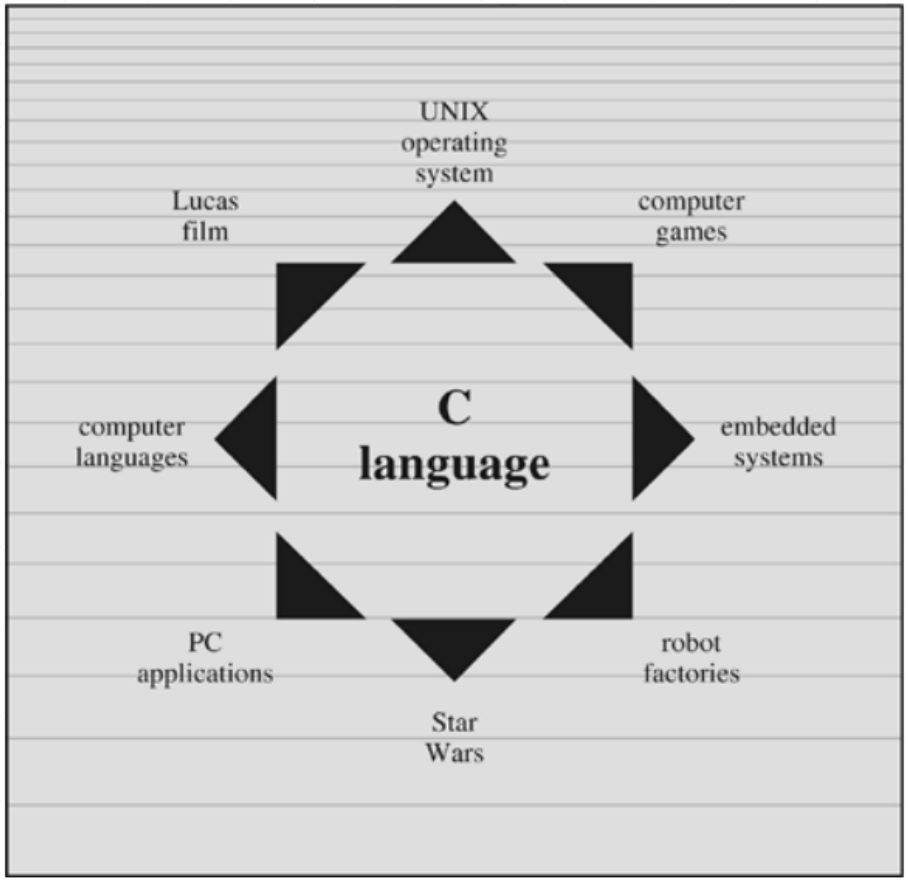
M. Stępniak, Laboratorium Informatyki

Zalety języka C

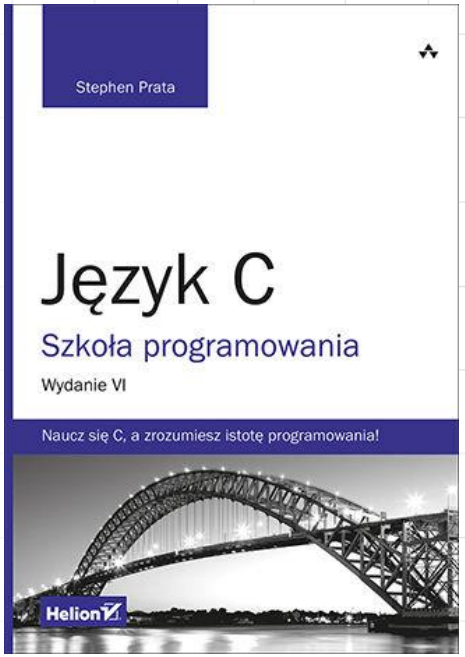
- Język **C** jest językiem **efektywnym**, programy w nim napisane charakteryzują się:
  - niewielką objętością
  - duża szybkość działania

  Wynika to z wysokiego stopnia precyzji utożsamianego z językami asemblera.
- **Przenośność** – kompilatory od najmniejszych mikroprocesorów do superkomputerów.
- **Elastyczność** – zaleta/wada – więcej swobody przy obarczeniu większą odpowiedzialnością.

Obszary zastosowań języka C



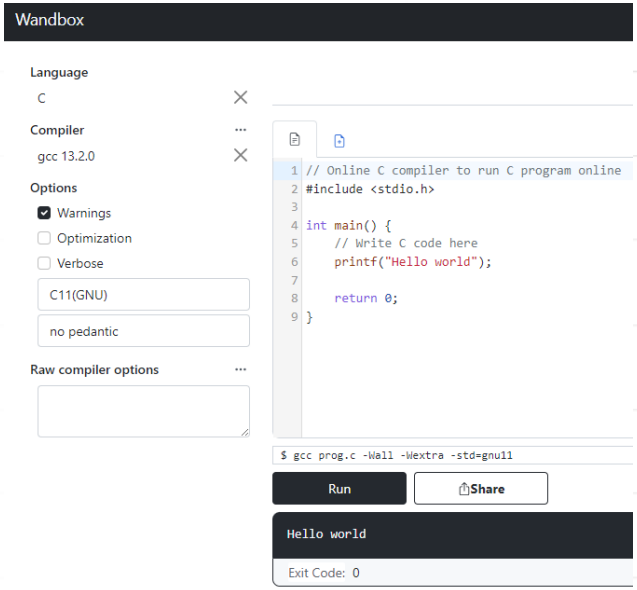
Stephen Prata, Język C. Szkoła programowania.



Edytor + narzędzia + kompilator = środowisko

- **IDE** (ang. integrated development environment) zintegrowanie środowisko do tworzenia, modyfikowania i testowania oprogramowania:
  - *Microsoft Visual Studio*
  - *Code::Blocks*
  - *Dev-C++*
  - *CodeLite*
  - *Eclipse*

- Kompilatory i narzędzia **on-line**:
  - *mycompiler.io*
  - *onlinegdb.com*
  - *onecompiler.com*
  - *programiz.com*
  - *wandbox.org*



Edytor + narzędzia + kompilator = środowisko

- **IDE** (ang. integrated development environment) zintegrowanie środowisko do tworzenia, modyfikowania i testowania oprogramowania:
  - *Microsoft Visual Studio* 
  - *Code::Blocks* 
  - *Dev-C++* 
  - *CodeLite* 
  - *Eclipse* 

- Kompilatory i narzędzia **on-line**:
  - *mycompiler.io* 
  - *onlinegdb.com* 
  - *onecompiler.com* 
  - *programiz.com* 
  - *wandbox.org*

- Zajęcia laboratoryjne:
  - *Notepad++ – edytor* 
  - *MinGW – pakiet GNU zawierający m.in. kompilator GCC oraz zestaw narzędzi*