

Programowanie obiektowe

Biocybernetyka i inżynieria biomedyczna
Semestr letni 2020/2021

Materiały wstępne do zajęć

mgr inż. Piotr Bączyk

Środowisko do zajęć

1. Zajęcia prowadzone są w oparciu o język C++20.
2. Mają Państwo możliwość rozliczania zadań w wybranym przez siebie innym języku poznanym w ramach samodzielnej pracy wspierającym paradygmat obiektowy spośród:
 - Python
 - Java
 - C#
 - Inny język po uzgodnieniu z prowadzącym

Środowisko do zajęć

1. Przygotowanie środowiska programistycznego dla C++ wymaga instalacji:
 - Kompilatora i linkera (zwykle są w jednym pakiecie)
 - Zintegrowanego środowiska programistycznego
2. Mają Państwo pełną dowolność w wyborze elementów środowiska, nie mniej zajęcia oparte są o zalecane środowisko. Dalej również zawarta jest instrukcja instalacji zalecanego środowiska.

Środowisko dla języka C++

	Zalecane	Inne możliwości
Kompilator	gcc w wersji $\geq 10.2.0$	Clang w wersji $\geq 12.0.0$ MSVC w wersji ≥ 14.29 (1929)
Zintegrowane środowisko programistyczne	CLion 2021.1.1	Microsoft Visual Studio (wszystkich opcji wymienić się nie da, teoretycznie da się pisać z wykorzystaniem edytora tekstu)

Instalacja gcc pod systemem Windows

1. Możliwości instalacji

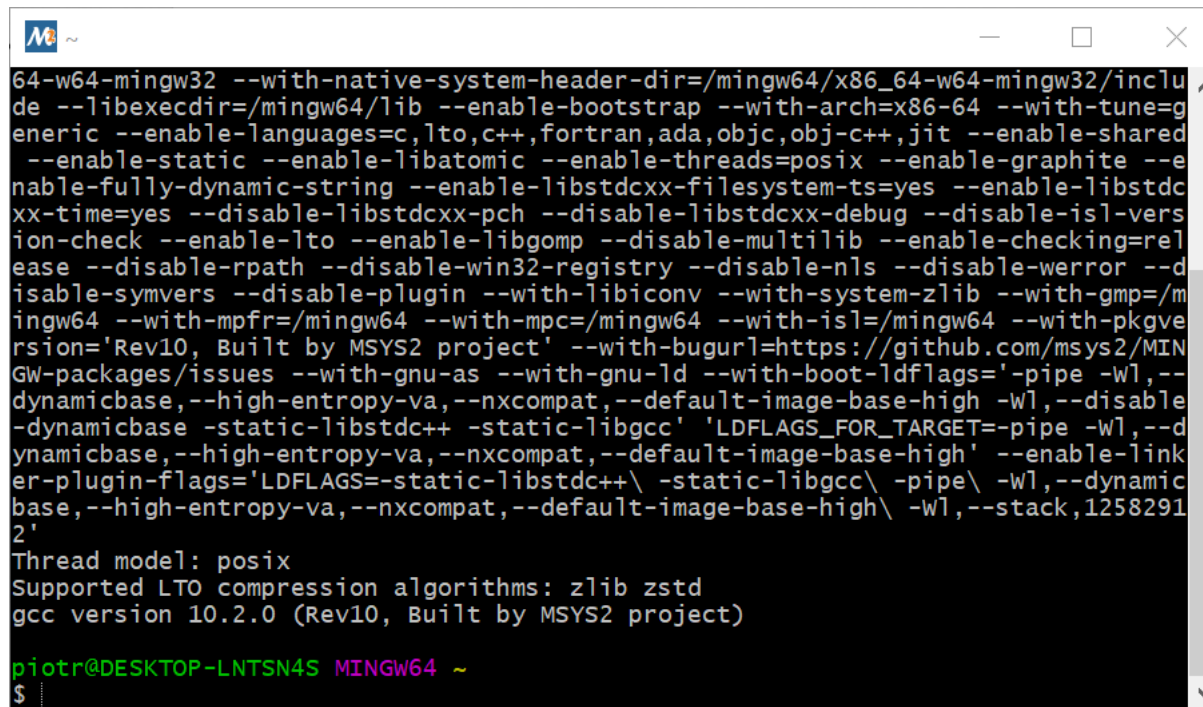
- Windows Subsystem for Linux (trudniejsza instalacja, większa elastyczność)
- MSYS2 (łatwiejsza instalacja)

Instalacja MSYS2

1. Pobierz i zainstaluj https://repo.msys2.org/distrib/x86_64/msys2-x86_64-20210419.exe
2. Uruchom MSYS2
3. Pobierz aktualną listę pakietów MSYS2. W konsoli wpisz: `pacman -Syu`
4. Krok 3 może wymagać powtórzenia, gdy pojawi się komunikat o konieczności zamknięcia okna, zamknij okno i powtórz krok 3.
5. Zaktualizuj pakiety. W konsoli wpisz: `pacman -Su`
6. Zainstaluj GCC: W konsoli wpisz: `pacman -S --needed base-devel mingw-w64-x86_64-toolchain`

Instalacja MSYS2

1. Potwierdź wynik instalacji (np. sprawdzając wersję gcc wpisując w konsolę `gcc -v`). Wynik powinien być podobny do:



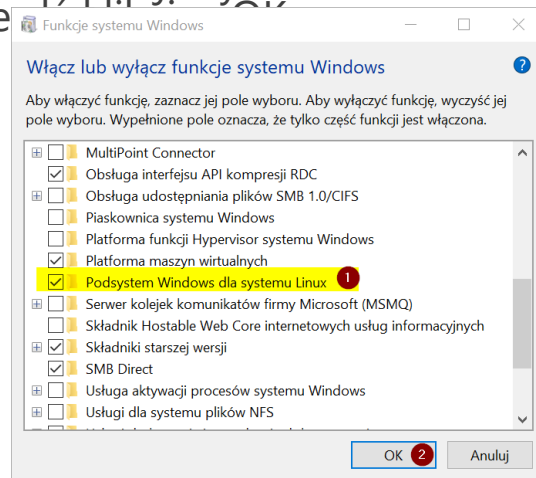
```
64-w64-mingw32 --with-native-system-header-dir=/mingw64/x86_64-w64-mingw32/inclu
de --libexecdir=/mingw64/lib --enable-bootstrap --with-arch=x86-64 --with-tune=g
eneric --enable-languages=c,lto,c++,fortran,ada,objc,obj-c++,jit --enable-shared
--enable-static --enable-libatomic --enable-threads=posix --enable-graphite --e
nable-fully-dynamic-string --enable-libstdcxx-filesystem-ts=yes --enable-libstdc
xx-time=yes --disable-libstdcxx-pch --disable-libstdcxx-debug --disable-lsl-vers
ion-check --enable-lto --enable-libgomp --disable-multilib --enable-checking=rel
ease --disable-rpath --disable-win32-registry --disable-nls --disable-werror --d
isable-symvers --disable-plugin --with-libiconv --with-system-zlib --with-gmp=/m
ingw64 --with-mpfr=/mingw64 --with-mpc=/mingw64 --with-lsl=/mingw64 --with-pkgve
rsion='Rev10, Built by MSYS2 project' --with-bugurl=https://github.com/msys2/MIN
GW-packages/issues --with-gnu-as --with-gnu-ld --with-boot-ldflags='-pipe -Wl,--
dynamicbase,--high-entropy-va,--nxcompat,--default-image-base-high -Wl,--disable
-dynamicbase -static-libstdc++ -static-libgcc' 'LDFLAGS_FOR_TARGET=-pipe -Wl,--d
ynamicbase,--high-entropy-va,--nxcompat,--default-image-base-high' --enable-link
er-plugin-flags='LDFLAGS=-static-libstdc++\ -static-libgcc\ -pipe\ -Wl,--dynamic
base,--high-entropy-va,--nxcompat,--default-image-base-high\ -Wl,--stack,1258291
2'
Thread model: posix
Supported LTO compression algorithms: zlib zstd
gcc version 10.2.0 (Rev10, Built by MSYS2 project)

piotr@DESKTOP-LNTSN4S MINGW64 ~
$
```

Instalacja Windows Subsystem For Linux

Instrukcja pochodzi z <https://docs.microsoft.com/en-us/windows/wsl/install-win10>

1. Upewnij się, że masz aktualną wersję Windows.
2. W menu Start wpisz „Włącz lub wyłącz funkcje systemu Windows”. W otwartym oknie zaznacz „Podsystem Windows dla systemu Linux” i zatwierdź.



To samo uzyskać można również uruchamiając PowerShell w trybie Administratora (Prawy przycisk na Start i Program PowerShell (Administrator)) wpisując komendę:

```
dism.exe /online /enable-feature /featurename:Microsoft-Windows-Subsystem-Linux /all /norestart
```


Instalacja Windows Subsystem For Linux

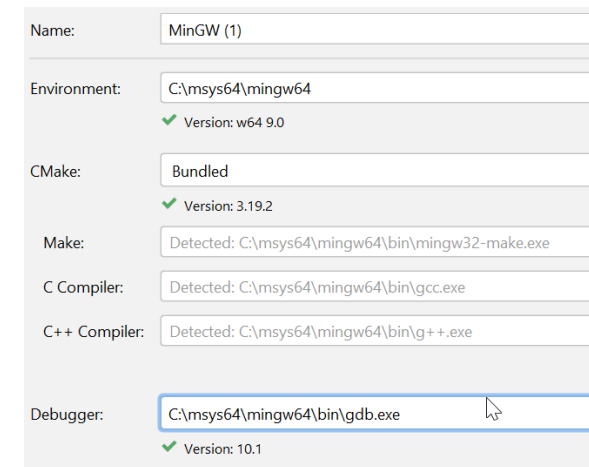
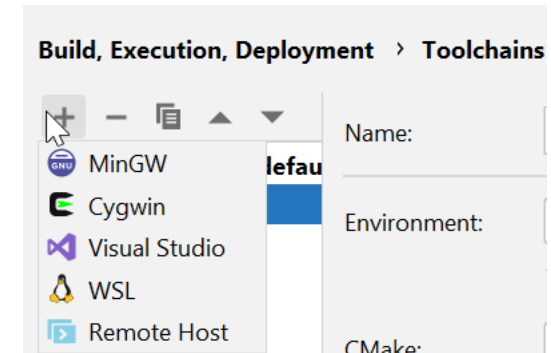
1. Włącz wsparcie dla maszyn wirtualnych. Uruchom PowerShell w trybie Administratora i wpisz komendę: `dism.exe /online /enable-feature /featurename:VirtualMachinePlatform /all /norestart`
2. Zainstaluj Linux kernel update package ze strony https://wslstorestorage.blob.core.windows.net/wslblob/wsl_update_x64.msi
3. Ustaw domyślną wersję WSL=2. Uruchom PowerShell w trybie dla administratora i wpisz komendę `wsl --set-default-version 2`
4. Zainstaluj aplikację Ubuntu 20.04 LTS z Microsoft Store.

Instalacja CLion

1. Zarejestruj się na <https://www.jetbrains.com/shop/eform/students> jeżeli dotychczas nie korzystałeś z narzędzi firmy JetBrains. Skorzystaj z konta studenckiego (darmowa licencja na wszystkie narzędzia).
2. Na przyszłość zalecam instalację JetBrains Toolbox (jeszcze na co najmniej jednych zajęciach będziemy korzystać z narzędzi JetBrains). Proponuję zatem instalację poprzez to narzędzie, które najpierw należy pobrać ze strony <https://www.jetbrains.com/toolbox-app/>
3. Po zalogowaniu kontem utworzonym z wykorzystaniem maila studenckiego wystarczy kliknąć Install przy CLion.

Łączenie MSYS2 z CLion

1. Wybieramy z menu: File -> Settings -> Build, Execution, Toolchains a następnie Toolchains
2. Klikamy przycisk + i wybieramy MinGW
3. Wybieramy ścieżkę (podfolder mingw64 w katalogu instalacyjnym MSYS2). Domyślnie C:\msys64\mingw64
4. Trzy zielone ptaszki oznaczają, że się powiodło

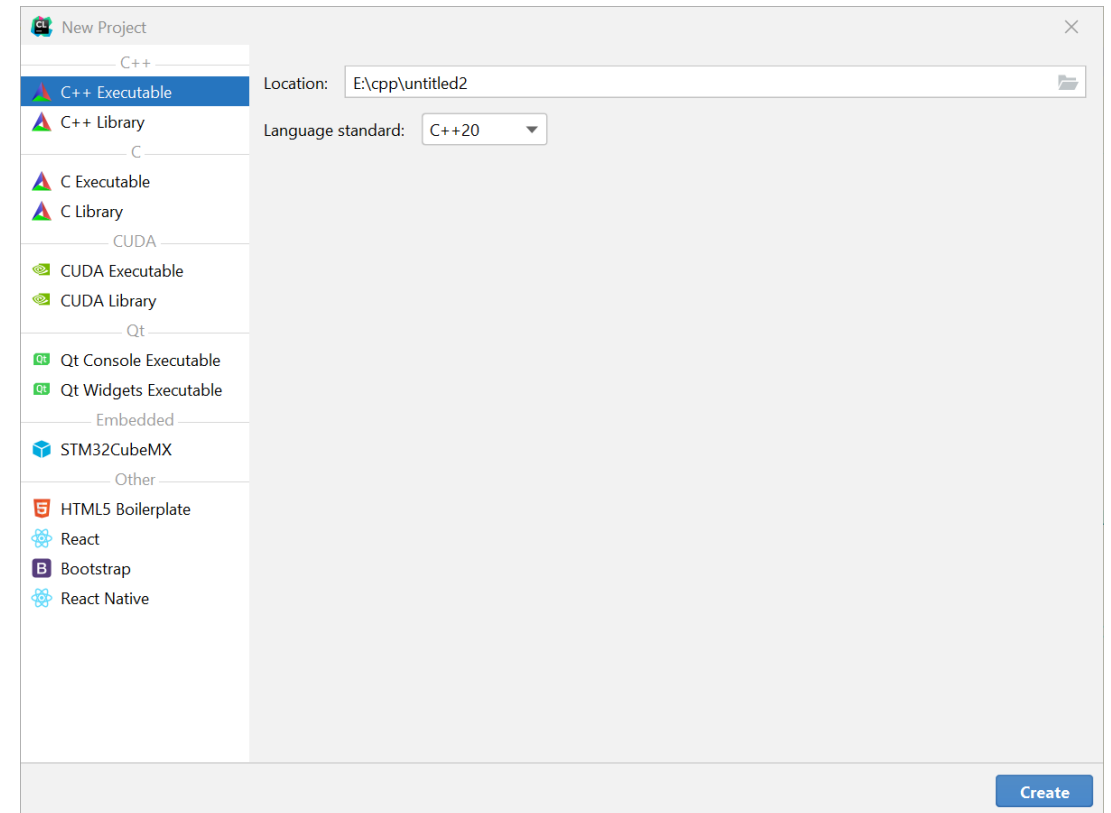


Łączenie WSL2 z CLion

1. Instrukcja: <https://www.jetbrains.com/help/clion/how-to-use-wsl-development-environment-in-product.html#wsl-general>
Uruchamiamy aplikację Ubuntu i przechodzimy od punktu 3. sekcji Configure WSL oraz sekcję Configure a WSL toolchain for your project

Tworzenie nowego projektu w CLion

1. Z Menu wybieramy File -> New Project
2. W oknie wybieramy C++ Executable, lokalizację projektu oraz standard języka C++20
3. Klikamy Create



Plik CMakeLists.txt

1. Plik CMakeLists.txt jest konfiguracją CMake, który zarządza procesem kompilacji programu.
2. Po linii `set(CMAKE_CXX_STANDARD 20)` sugeruję dodać linie:
`set(CMAKE_RUNTIME_OUTPUT_DIRECTORY "${CMAKE_CURRENT_SOURCE_DIR}/bin")`
`add_compile_options(-Wall -Wextra -pedantic -Werror)`
i potraktować te linie jako domyślne dla projektów w ramach tych zajęć.

Pierwsza z nich zapewnia, że plik wykonywalny powstały w wyniku kompilacji umieszczony zostanie w podkatalogu projektu o nazwie `bin`.

Druga z nich dodaje dodatkowe opcje kompilatora gcc, przydatne przy nauce programowania w C++. Z tymi flagami będę również kompilował Państwa projekty.

Protipy

1. Nauka programowania jest podobna do uprawiania sportu - wymaga częstych i samodzielnych ćwiczeń. Poprzez przepisywanie kodu innych, czy oglądanie tutoriali nie da się nauczyć programowania. Brak samodzielnej pracy jest główną przyczyną niepowodzeń w nauce programowania.
2. Warto często kompilować programy i je sprawdzać. Błędy programisty są czymś naturalnym (zdarzają się najbardziej doświadczonym programistom). Im częściej kompilujemy program, tym mniej błędów będzie do poprawienia na sam koniec.
3. Kompilator nie jest wrogiem programisty. Jeżeli kompilator zwraca błąd kompilacji, to wynika on z błędu popełnionego przez programistę. Gcc bardzo dokładnie wskazuje lokalizację i przyczynę błędu i należy traktować je jako wskazówki przy poprawianiu kodu.
4. Warto również zwracać uwagę na podpowiedzi CLion, który potrafi zaznaczyć błędne linie a duża część jego sugestii stylistycznych jest trafna.

Materiały do nauki

Lista zawiera pozycje, z których korzystam przy opracowywaniu zajęć i które polecam do zapoznania (przynajmniej jedna-dwie wybrane pozycje szczególnie te nowe, gdyż większość z nich jest dość obszerna). Jak widać lista zawiera pozycje wydane w tym oraz w ubiegłym roku – została przeze mnie zaktualizowana.

C++:

1. Stroustrup B., „Język C++. Kompendium wiedzy”, Helion 2014
2. Gregoire M., „Professional C++”, wydanie V 2021 (po angielsku)
3. Hortsmann C., „Wprowadzenie do C++. Efektywne nauczanie”, Helion 2021
4. Josuttis N. „C++. Biblioteka standardowa. Podręcznik programisty.”, Helion 2014
5. Browning J. B., Sutherland B., „C++20 Biblioteka techniczna. Problemy i rozwiązania”, Apress 2020
6. Stroustrup B., „Programowanie. Teoria i praktyka z wykorzystaniem C++”, wydanie III, Helion 2020
7. Meyers S. „Skuteczny nowoczesny C++”, 2015

Materiały do nauki

UML:

1. Miles R., Hamilton K. „Język UML 2.0. Wprowadzenie”, 2007
2. Bruegge B., Dutoit A. „Inżynieria oprogramowania w ujęciu obiektowym. UML, wzorce projektowe i Java”, 2011

Materiały do nauki

Sprawdzone źródła z sieci

1. <https://en.cppreference.com/w/> [(prawie) kompletna dokumentacja C++ w przyjaznej postaci, jest też wersja polska ale bardzo uboga]
2. <https://books.goalkicker.com/CPlusPlusBook/> [całkiem dobry materiał w postaci PDF, aktualny i pokrywa najważniejsze zagadnienia z zakresu składni języka i biblioteki standardowej, bardzo dużo przykładów kodu]