Materiały wprowadzające: „Środowisko CLion”

dr inż. Arkadiusz Chrobot

4 września 2026

1. Wprowadzenie

Do tworzenia programów w ramach zajęć laboratoryjnych z Podstaw Programowania 1 będzie używane zintegrowane środowisko do programowania (ang. *Integrated Development Environment*) *CLion* firmy *JetBrains*.

Jest to profesjonalne, komercyjne oprogramowanie, ale dostępne bez opłat dla studentów, po wcześniejszym zarejestrowaniu się na stronie producenta i aktywowaniu licencji.

Celem tej instrukcji jest objaśnienie, jak wykonać podstawowe czynności związane ze wspomnianą rejestracją, a także instalacją i aktywacją środowiska oraz tworzeniem przy jego pomocy oprogramowania.

Proszę zwrócić uwagę, że nie jest to wyczerpujący opis tego oprogramowania. Ten zająłby zbyt wiele miejsca i prawdopodobnie nie byłby tak dobry, jak dokumentacja dostępna na stronie producenta.

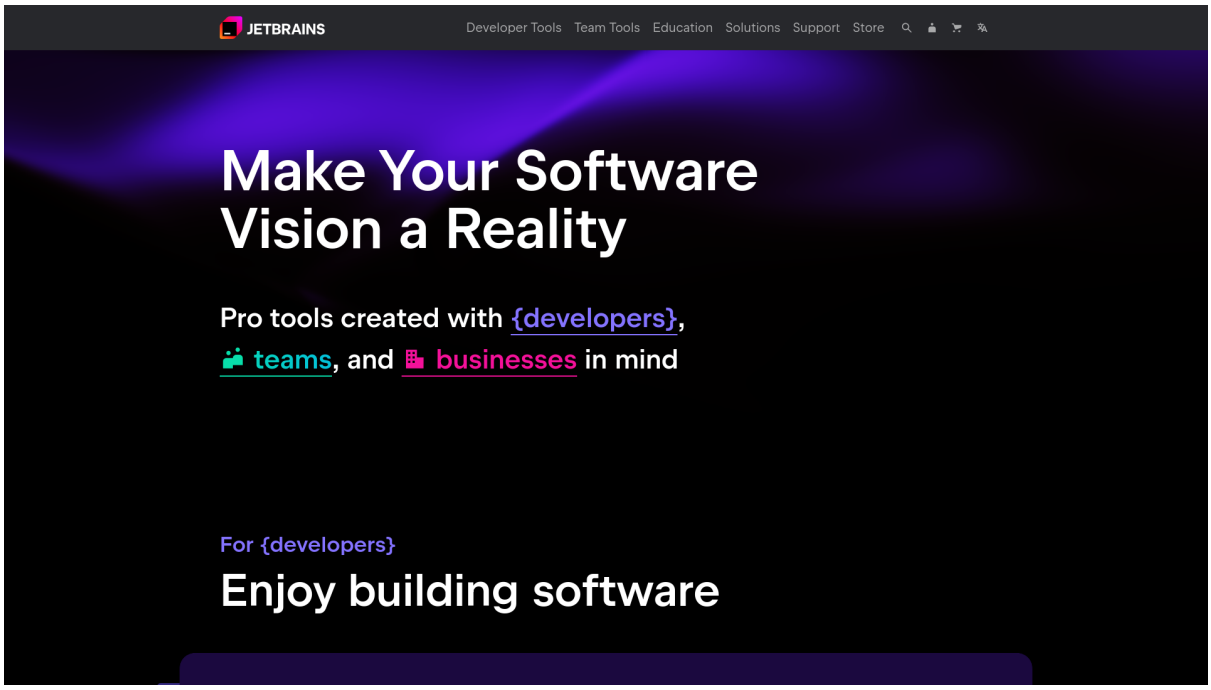
Pierwsze trzy rozdziały instrukcji wyjaśniają odpowiednio, jak pobrać (2), zarejestrować (3), a następnie skonfigurować (4) środowisko CLion.

Trzy kolejne opisują tworzenie projektu (5), kompilację i uruchomienie programu (6) oraz debugowanie (ang. *debugging*) (7).

Przedostatni rozdział zawiera kod źródłowy programu wykorzystywanego w jednym z zadań do samodzielnego wykonania na zajęciach w laboratorium. Zadania są zawarte w ostatnim rozdziale.

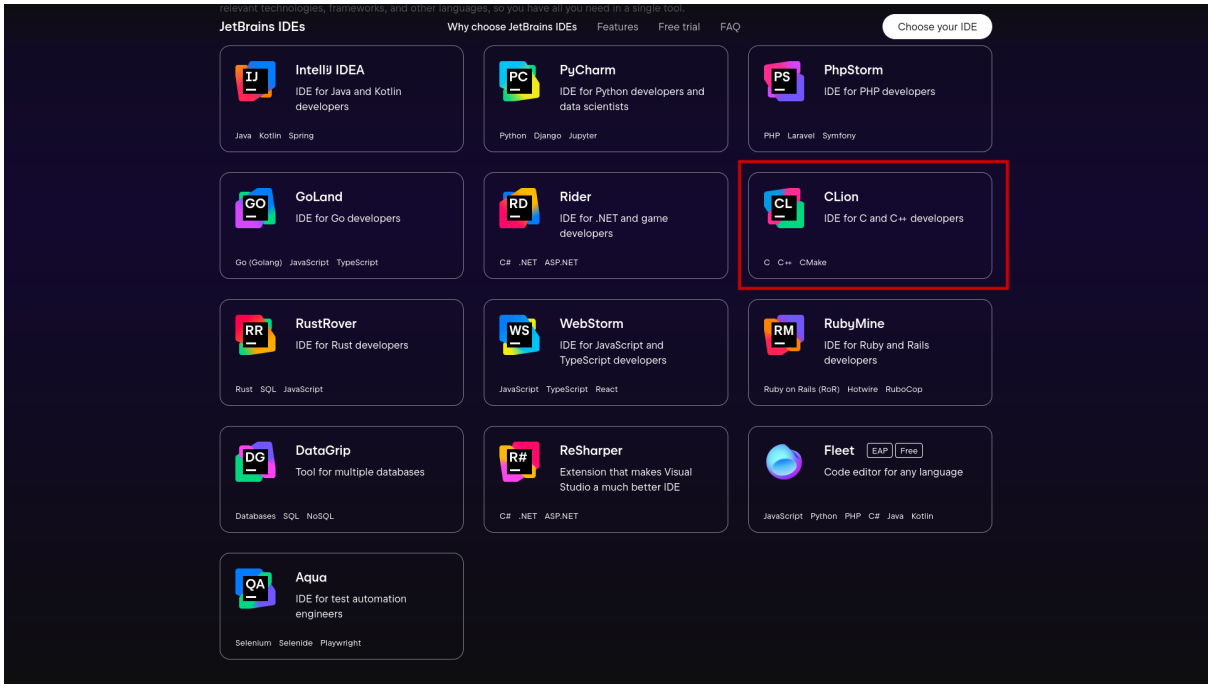
2. Pobranie oprogramowania

Aby pobrać środowisko CLion w wersji instalacyjnej, należy w przeglądarce WWW wpisać adres `https://www.jetbrains.com` i wejść na stronę producenta tego oprogramowania, jak pokazano na Rysunku 1.



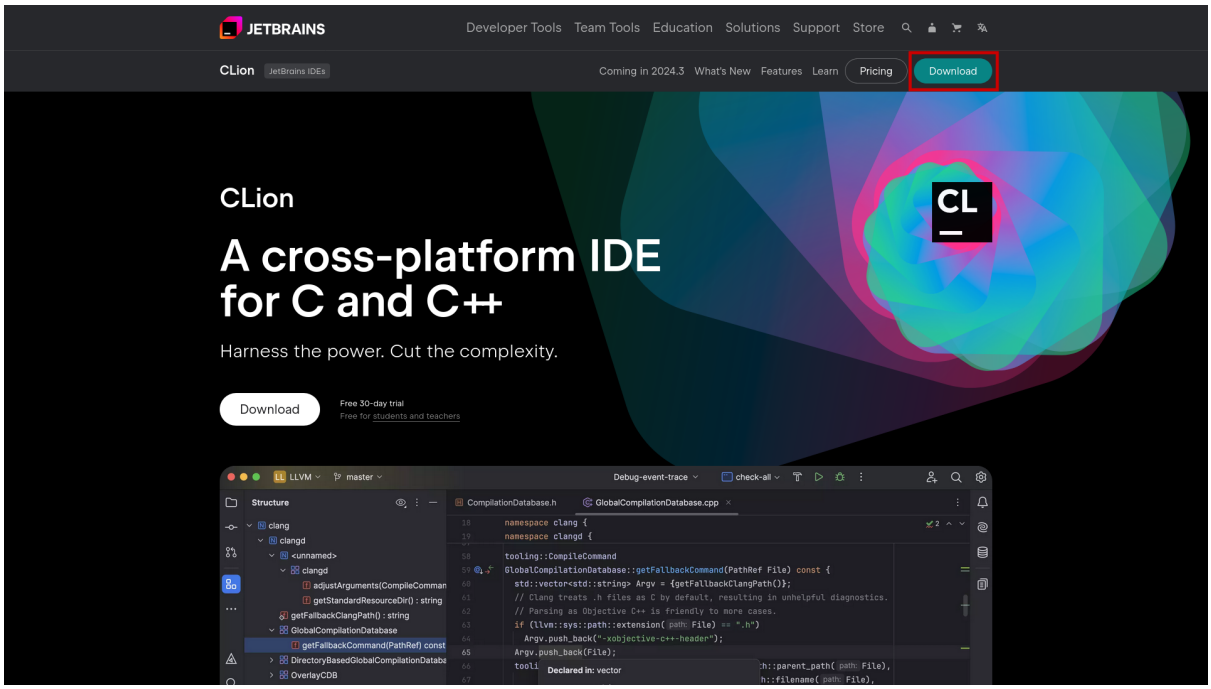
Rysunek 1: Strona producenta oprogramowania

Następnie należy przewinąć tę stronę, aż pokaże się widok podobny do tego, jaki przedstawiono na Rysunku 2, i kliknąć zaznaczony czerwonym prostokątem przycisk.



Rysunek 2: Wybór oprogramowania do pobrania

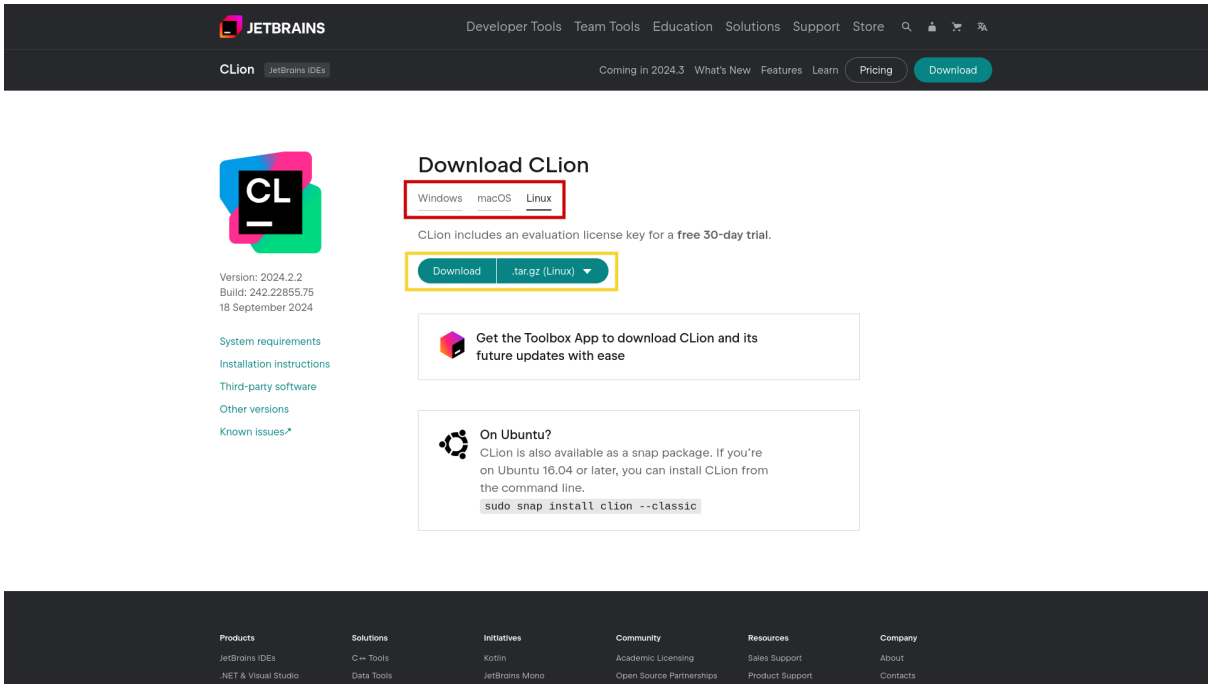
Po wykonaniu tej czynności w oknie przeglądarki powinna pokazać się strona podobna do tej, którą przedstawia Rysunek 3.



Rysunek 3: Strona pobierania środowiska CLion

Tu również trzeba kliknąć przycisk zaznaczony na czerwono, tym razem nazywający się po prostu „*Download*”.

Po wykonaniu tej czynności pojawi się kolejna strona z formularzem wyboru wersji instalacyjnej (Rysunek 4).



Rysunek 4: Wybór wersji instalacyjnej CLion

Na tej stronie, przy pomocy zakładek oznaczonych czerwonym prostokątem, można dokonać wyboru systemu operacyjnego, na którym CLion ma być zainstalowany. Dostępne są trzy opcje: Windows, macOS i Linux.

Aby pobrać środowisko, należy kliknąć napis „*Download*" z lewej strony podzielonego przycisku (ang. *split button*), zaznaczonego żółtym prostokątem.

Aby wybrać inny format pliku z wersją instalacyjną niż domyślnie oferowany, należy skorzystać z prawej części opisywanego przycisku.

Proszę zwrócić uwagę, że osoby korzystające z dystrybucji Ubuntu systemu Linux mogą zainstalować CLion, używając podanego na stronie polecenia lub za pomocą aplikacji *Centrum Oprogramowania*.

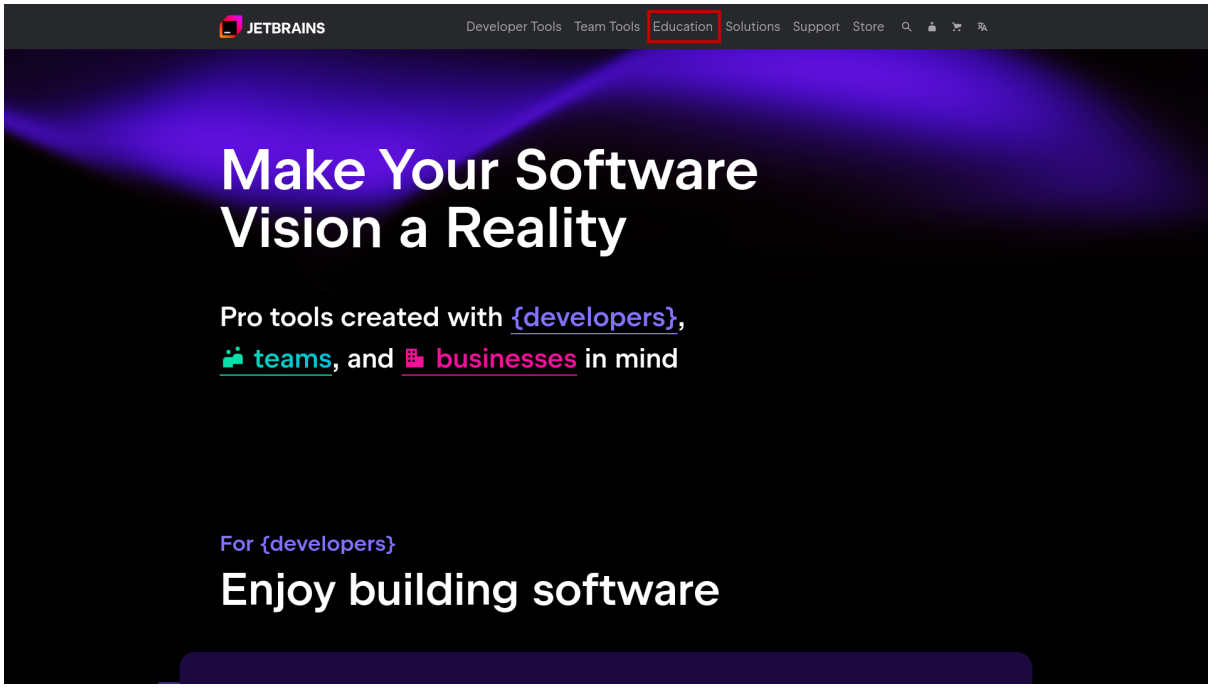
3. Rejestracja

Po zainstalowaniu środowisko domyślnie będzie pracowało tylko przez 30 dni.

Studenci mogą jednak skorzystać z darmowej licencji, która uprawnia ich do korzystania z CLion i innych narzędzi oferowanych przez JetBrains przez cały okres studiowania.

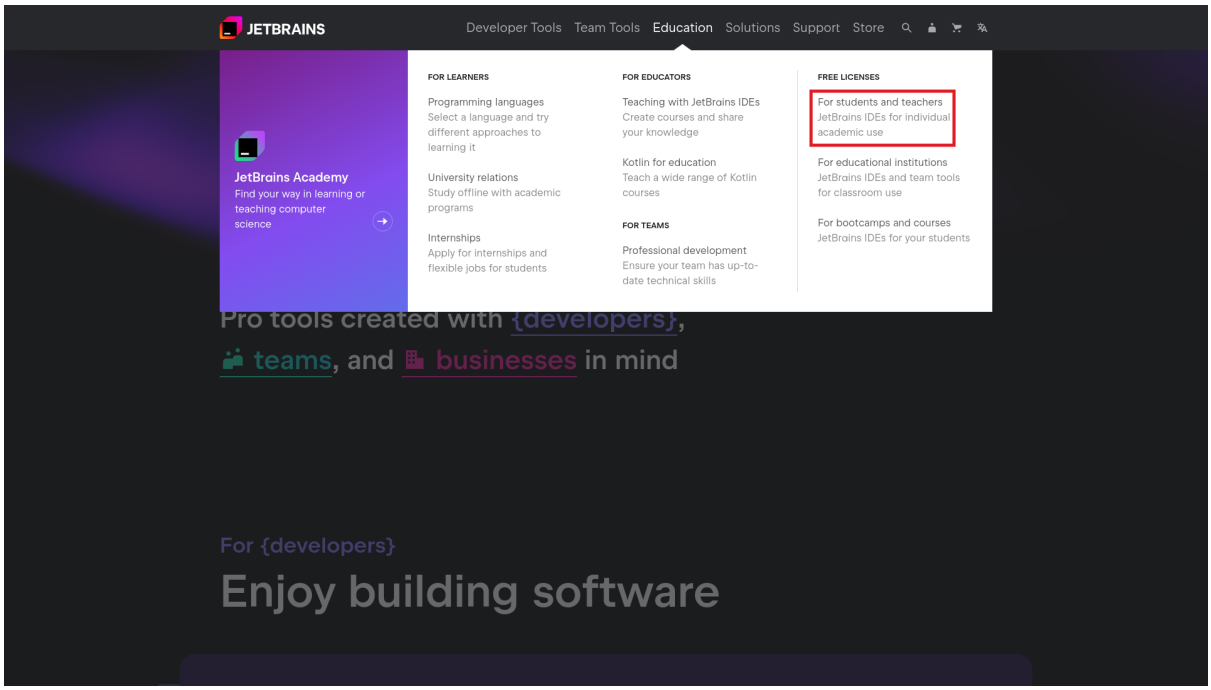
Aby ją uzyskać, należy zarejestrować się na stronie producenta.

Podobnie jak w przypadku pobierania oprogramowania, najpierw trzeba przy pomocy przeglądarki wejść na stronę firmy i kliknąć przycisk *Education*, oznaczony czerwonym prostokątem na Rysunku 5.



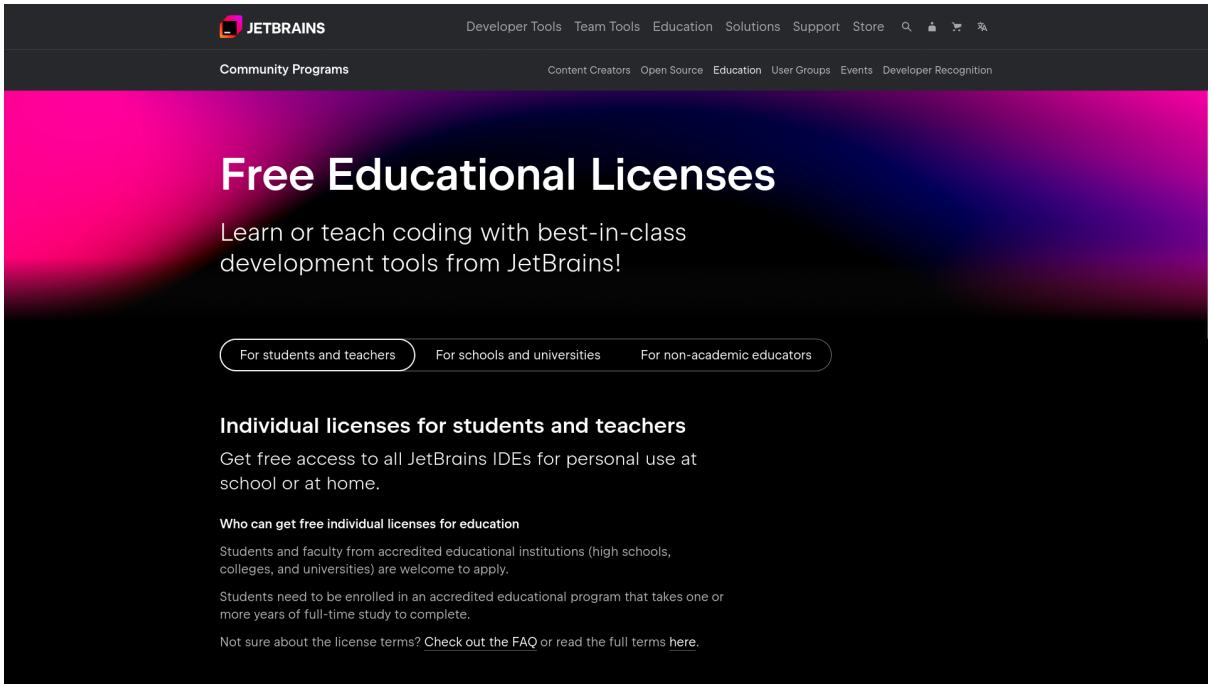
Rysunek 5: Pierwszy krok rejestracji

Po wykonaniu tych czynności pokaże się okienko, w którym należy wybrać link zaznaczony na czerwono na Rysunku 6.

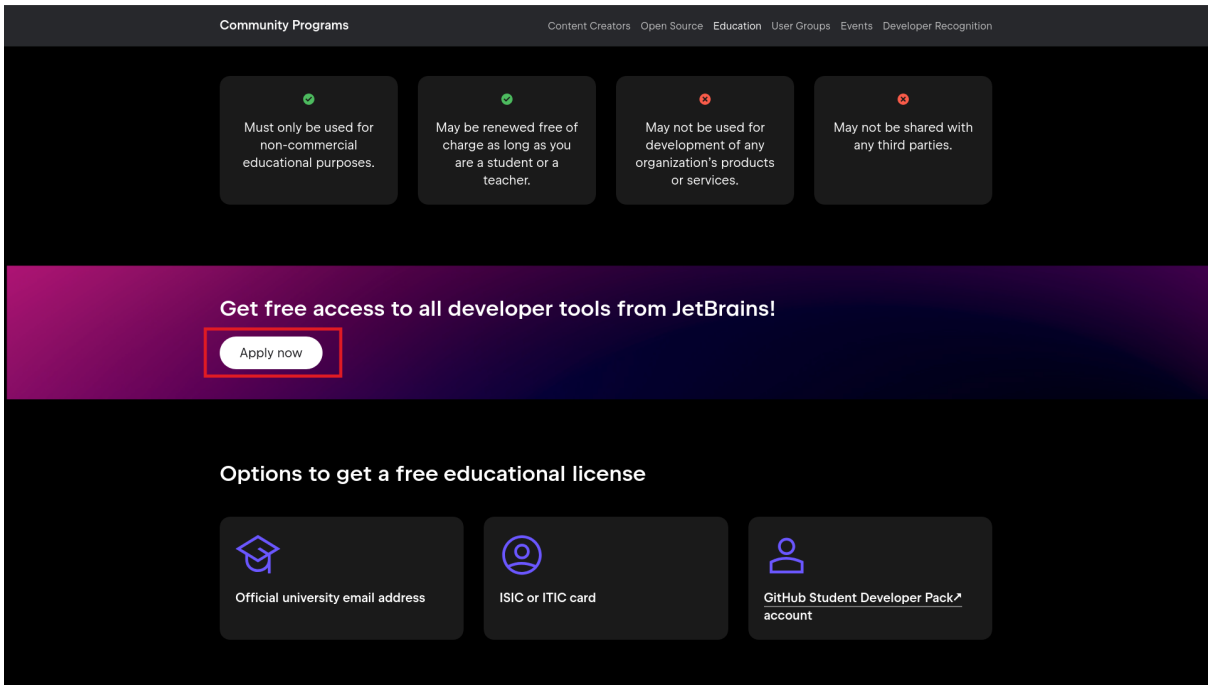


Rysunek 6: Link z wyborem licencji

Pojawi się nowa strona (Rysunek 7), którą trzeba przewinąć w dół, aż pojawi się fragment podobny do tego, który jest widoczny na Rysunku 8.

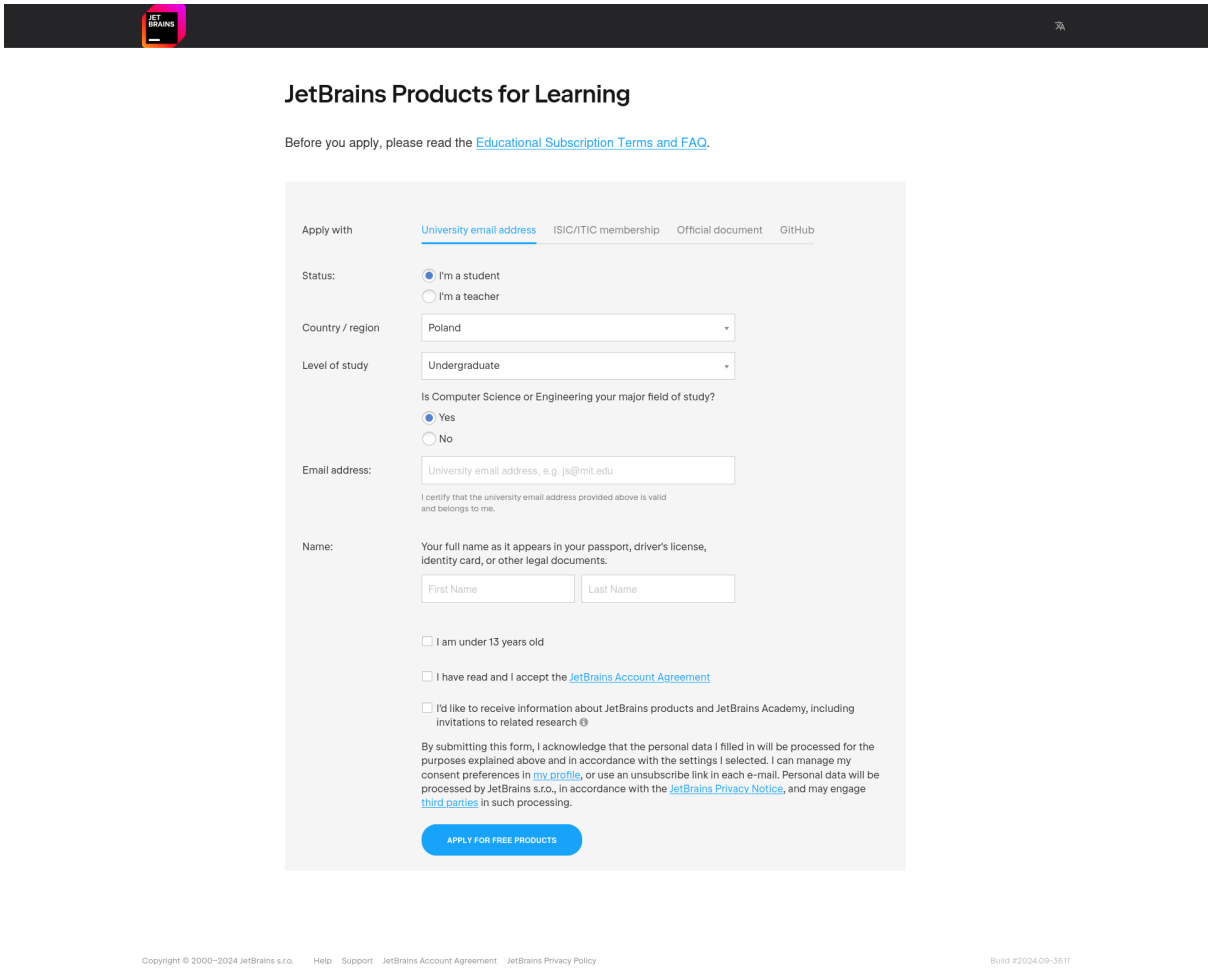


Rysunek 7: Strona z informacją o darmowych licencjach



Rysunek 8: Przycisk do rejestracji

Należy kliknąć przycisk „*Apply now*”.
Spowoduje to przejście do formularza (Rysunek 9), który należy wypełnić.
W polu e-mail należy wpisać adres poczty elektronicznej przydzielony przez uczelnię. Inny adres nie zadziała.

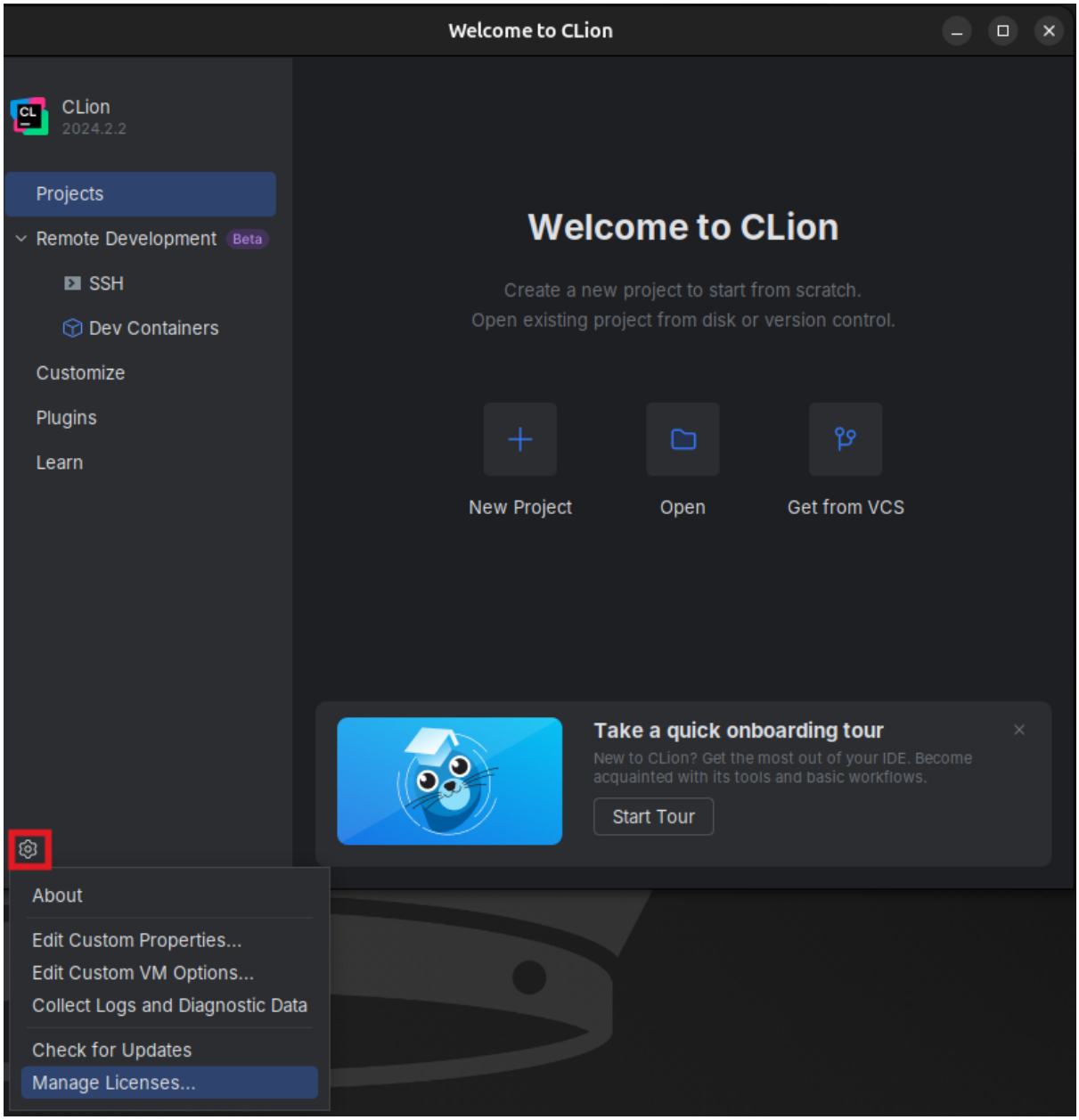


Rysunek 9: Formularz do rejestracji

Po otrzymaniu wiadomości od producenta oprogramowania należy wykonać opisane w niej kroki i aktywować przydzieloną licencję w środowisku CLion.

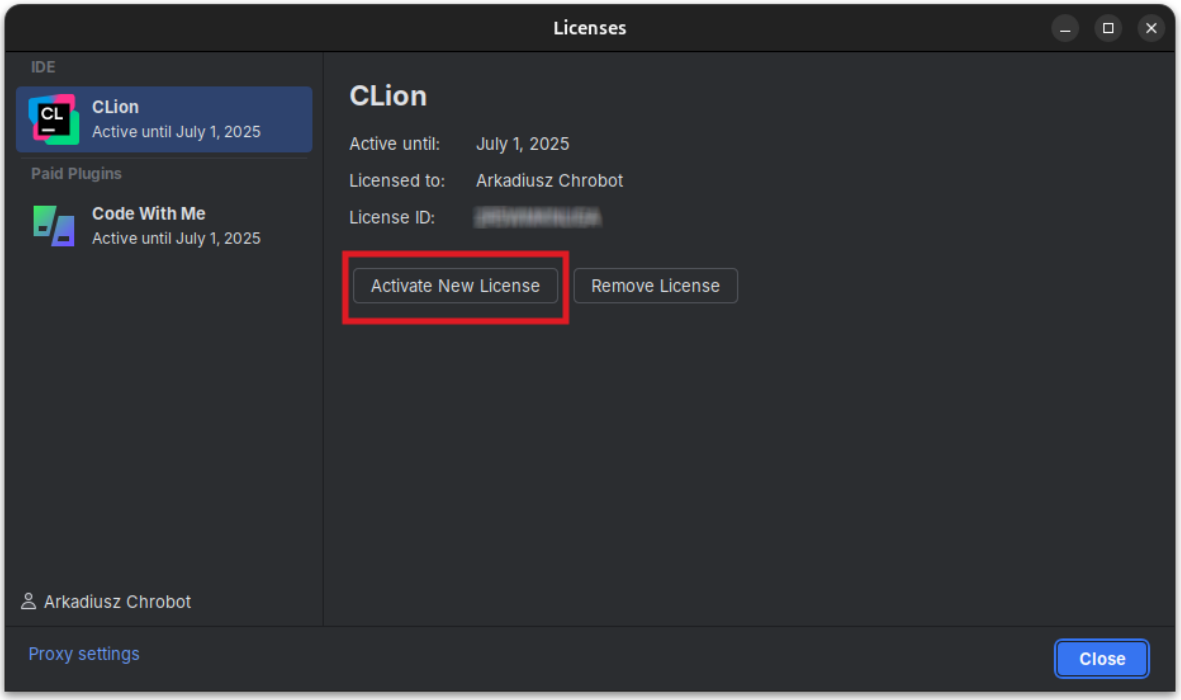
Tę ostatnią czynność należy rozpocząć od uruchomienia programu.

Kiedy pojawi się główne okno, należy kliknąć zaznaczony na Rysunku 10 „trybik” i po rozwinięciu menu wybrać z niego opcję „*Manage licenses…*”.



Rysunek 10: Sposób uruchomienia menedżera licencji

W nowym oknie (Rysunek 11) trzeba kliknąć przycisk opisany jako „*Activate New License*" i postępować zgodnie z kolejnymi instrukcjami wyświetlanymi przez aplikację.



Rysunek 11: Okno aktywacji licencji

Po zakończeniu tej operacji w oknie pojawi się informacja o nowej, aktywnej licencji, która będzie ważna przez rok.

Potem można ją odnowić, korzystając z założonego na stronie producenta konta.

4. Konfiguracja

Środowisko programistyczne CLion wyposażone jest w szereg udogodnień, które mogą być bardzo pomocne dla doświadczonego programisty.

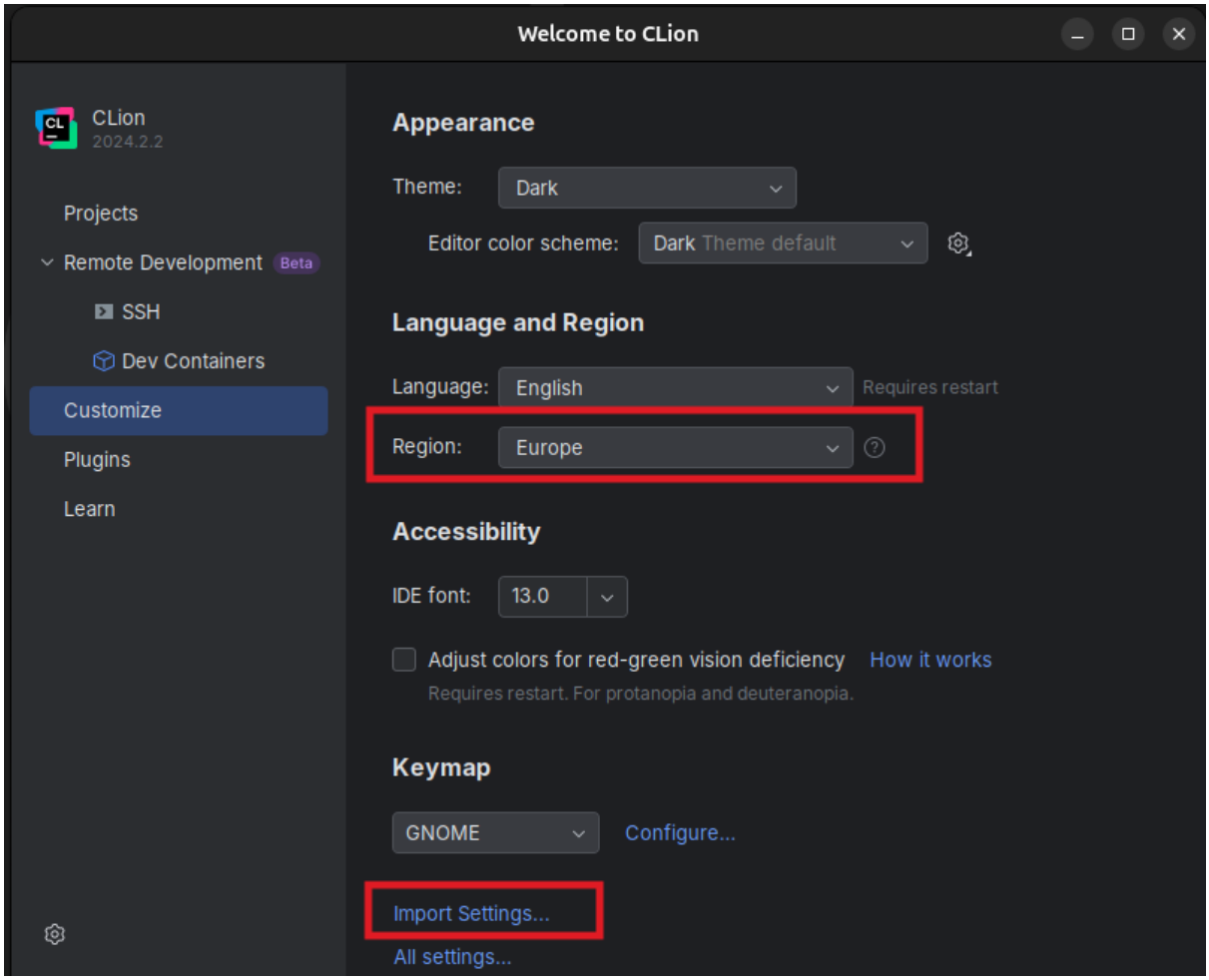
Niestety początkujących mogą one dezorientować i rozpraszać lub w inny sposób przeszkadzać adeptom programowania w uczeniu się.

Dlatego ze strony, na której znajduje się ta instrukcja, można pobrać plik konfiguracyjny o nazwie *settings.zip*, który niektóre z tych udogodnień wyłącza, na przykład automatyczne generowanie kodu bazujące na uczeniu maszynowym oraz etykiety dla parametrów i argumentów funkcji.

Plik ten włącza również opcje, które mogą być przydatne, na przykład generowanie ostrzeżeń przez kompilator.

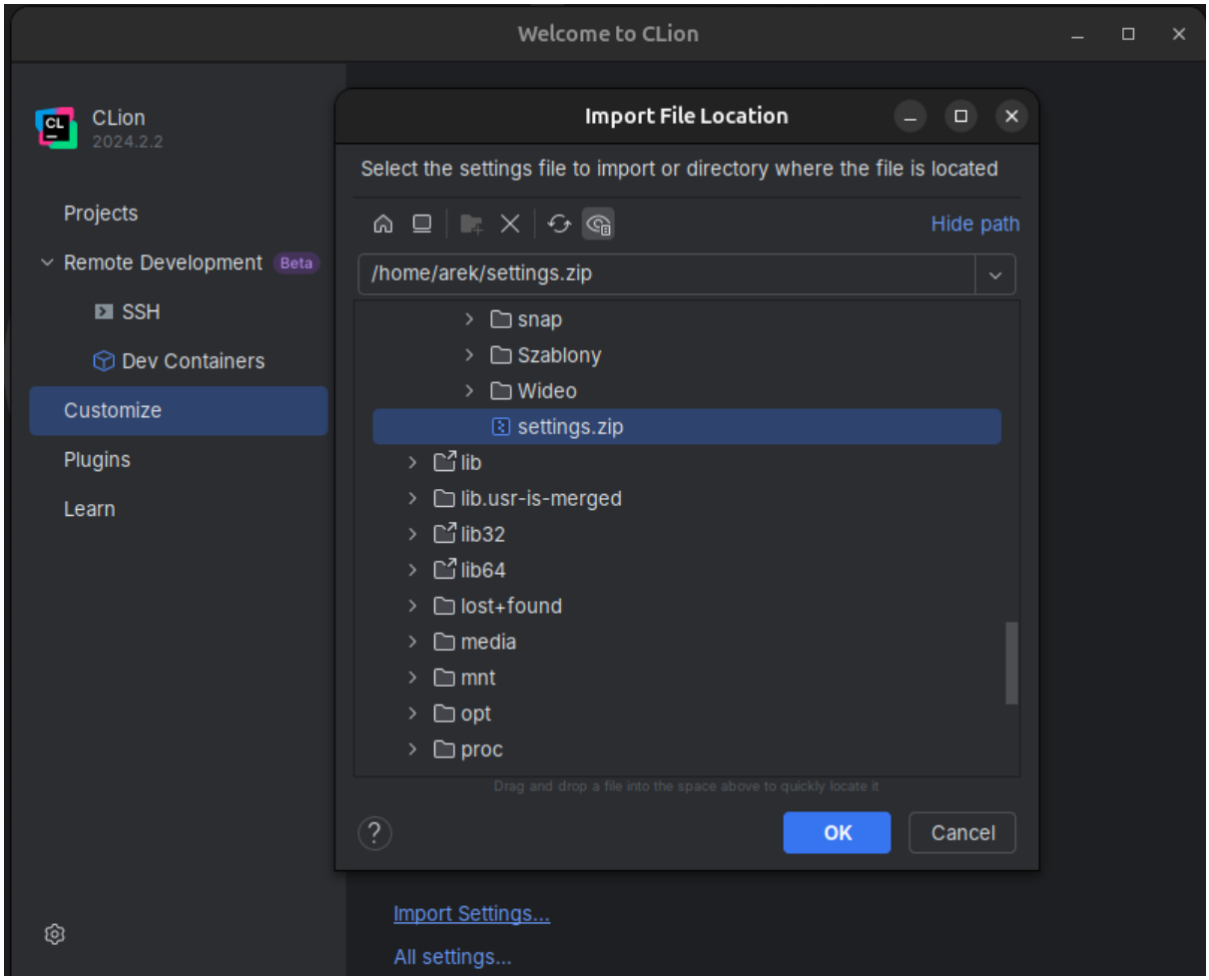
Aby skonfigurować CLion z użyciem wspomnianego pliku, należy uruchomić to środowisko i w oknie głównym po lewej stronie wybrać opcję „*Customize*”.

Następnie po prawej stronie można ustawić region na „*Europe*”, jeśli nie jest jeszcze skonfigurowany, a potem kliknąć link „*Import settings...*” (Rysunek 12).



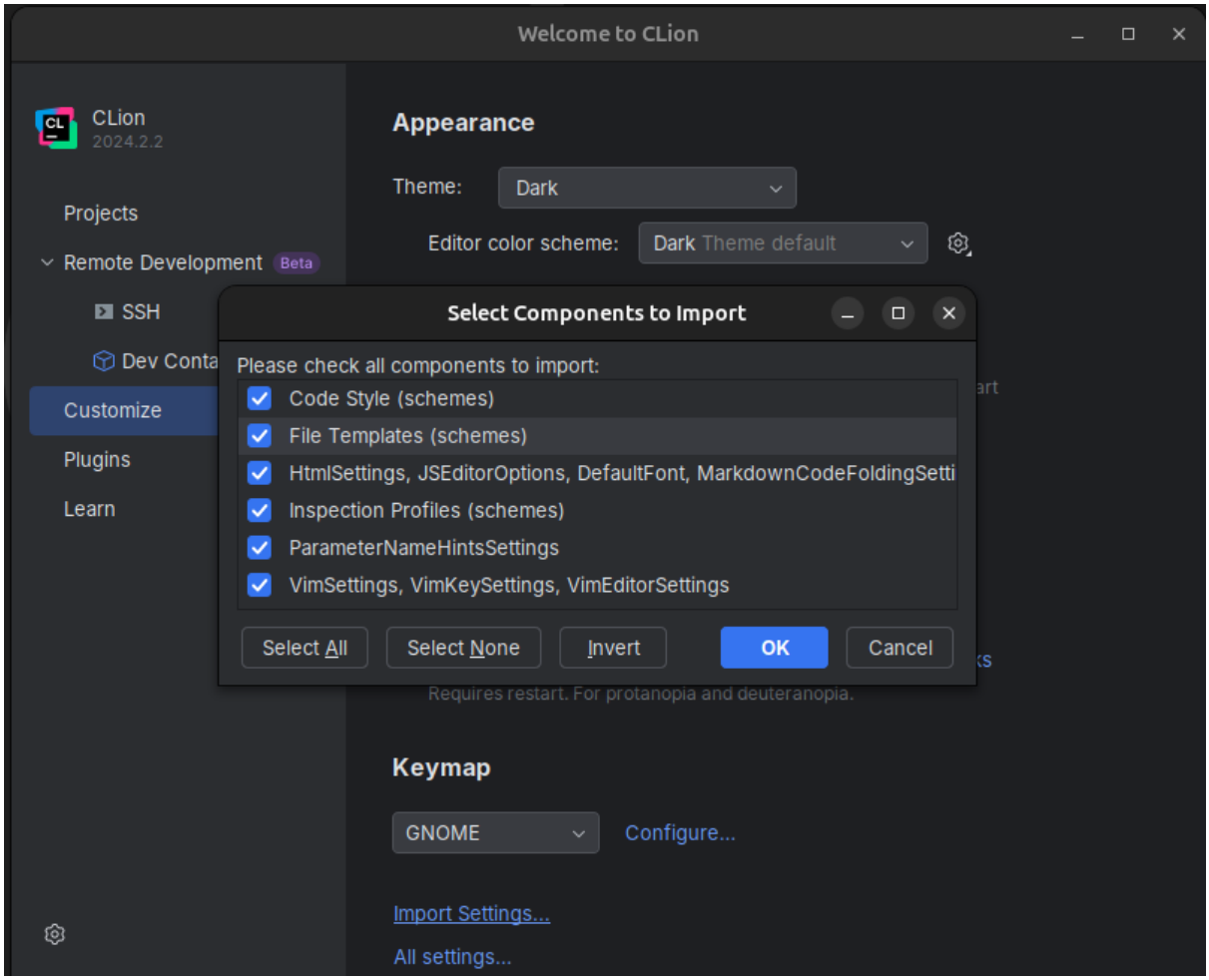
Rysunek 12: Konfigurowanie środowiska

Po pokazaniu się okna umożliwiającego wybór pliku, podobnego do tego na Rysunku 13, należy wskazać lokalizację pliku *settings.zip* i kliknąć *OK*.



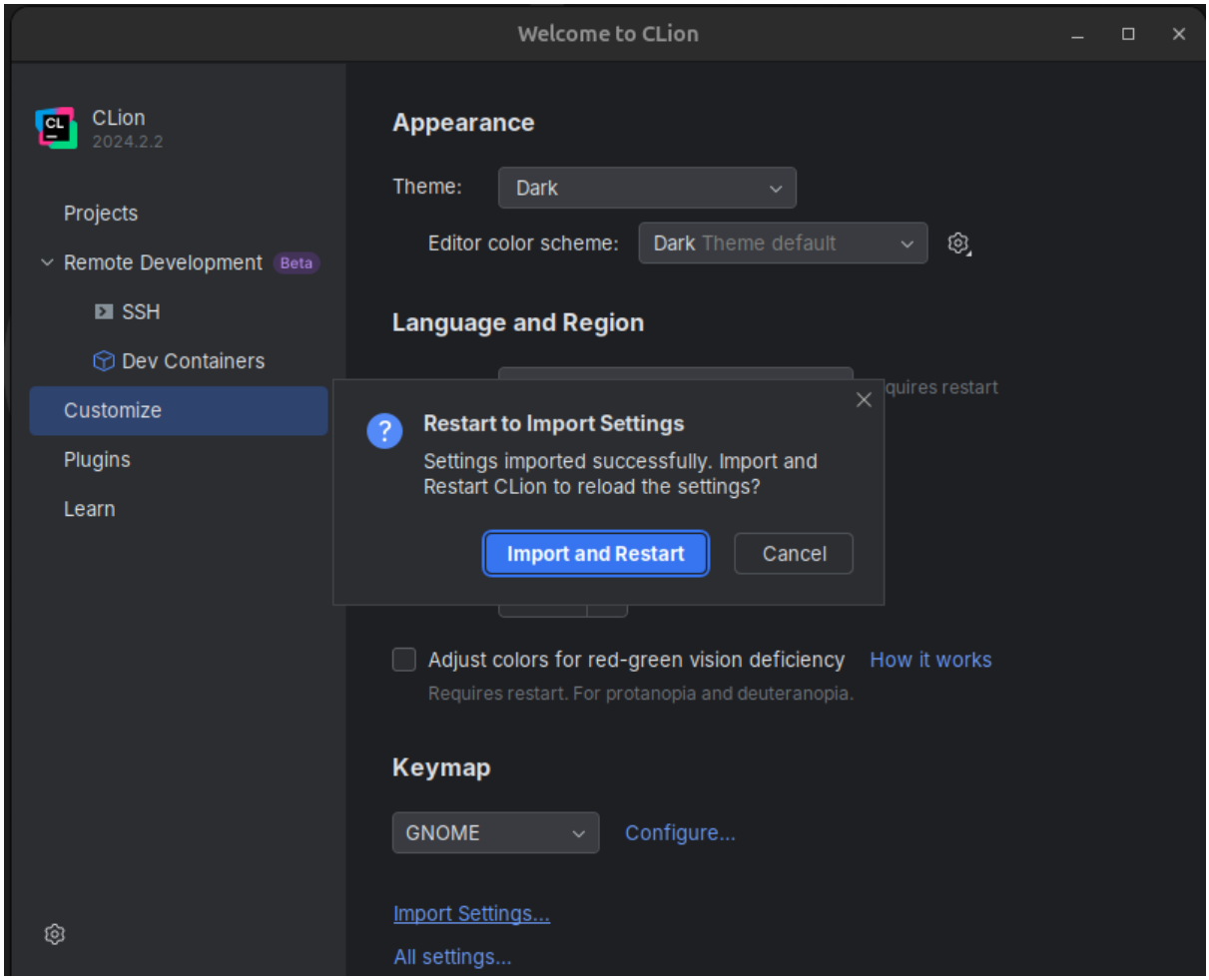
Rysunek 13: Wybór pliku konfiguracyjnego

W nowym oknie (Rysunek 14) należy zaznaczyć wszystkie opcje, jeśli nie są oznaczone, a następnie ponownie kliknąć przycisk *OK*.



Rysunek 14: Wybór importowanych opcji

Po ukazaniu się kolejnego okna wystarczy kliknąć przycisk „*Import and Restart*” (Rysunek 15) i poczekać na ponowne uruchomienie się środowiska.



Rysunek 15: Import pliku konfiguracyjnego

5. Tworzenie nowego projektu

Środowisko CLion umożliwia wygodne programowanie między innymi dzięki temu, że integruje pracę zewnętrznych narzędzi, takich jak kompilator GCC lub narzędzie do sterowania kompilacją o nazwie *CMake*.

Te narzędzia również mogą wymagać plików, w których na przykład jest zapisana ich konfiguracja.

Dlatego CLion gromadzi wszystkie pliki związane z tworzonym programem w *projekcie*, który zapisywany jest domyślnie w katalogu *CLionProjects*.

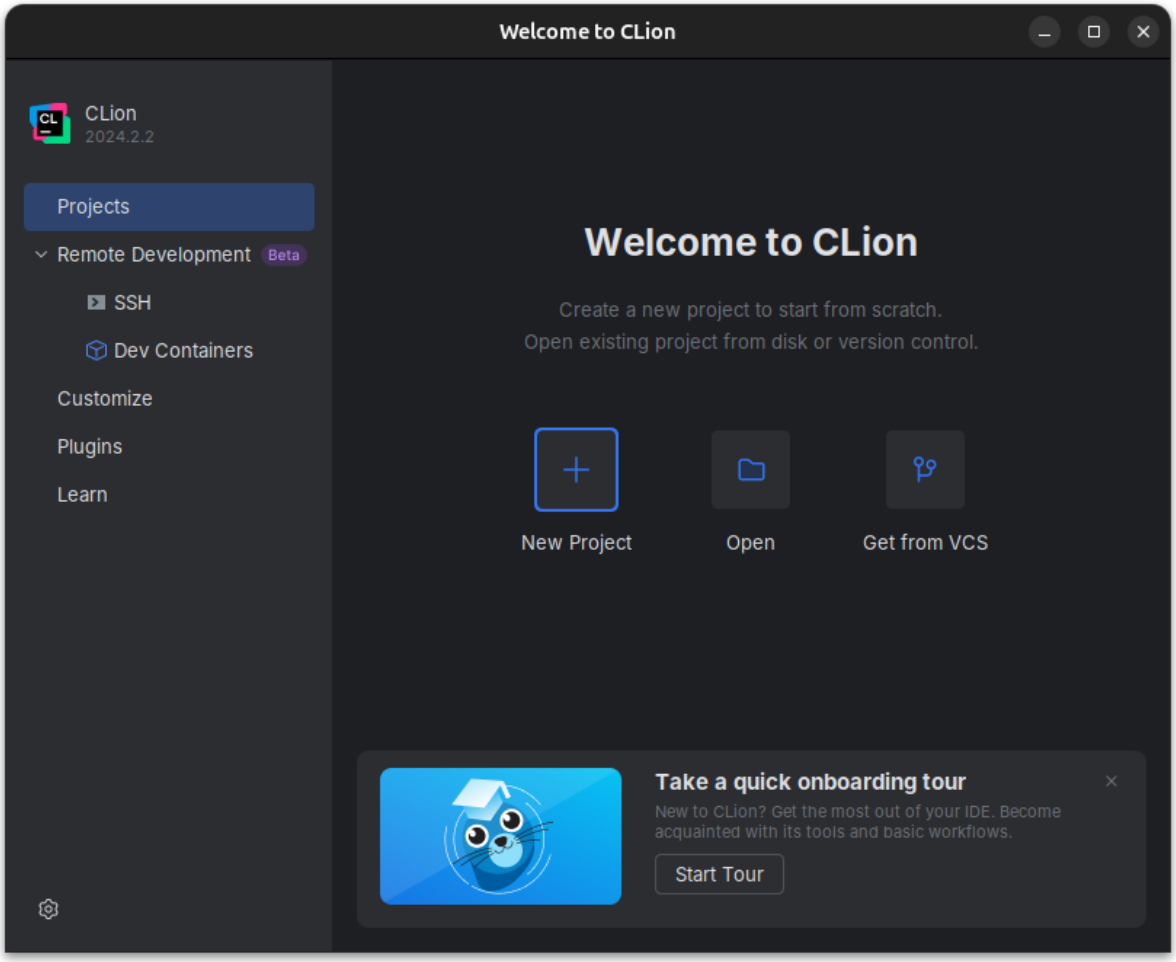
Najważniejszym z tych plików jest ten, który ma rozszerzenie `.c`, gdyż zawiera on kod źródłowy programu.

Środowisko zwykle nazywa go `main.c`.

Zatem przed rozpoczęciem pracy nad programem należy stworzyć nowy projekt.

Jeśli CLion nie był wcześniej używany, to po uruchomieniu będzie wyglądał podobnie jak na Rysunku 16.

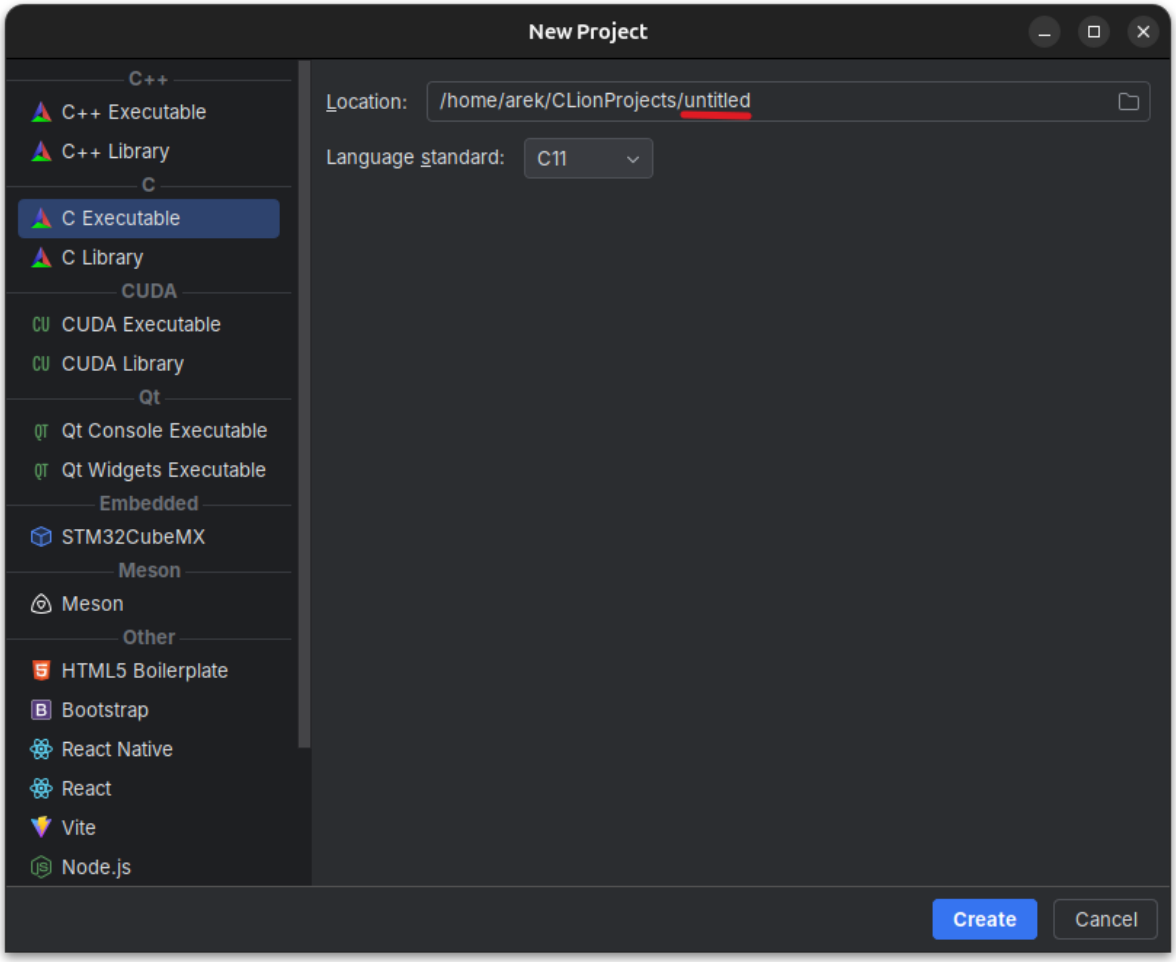
Aby utworzyć nowy projekt, należy kliknąć przycisk z plusem podpisany „*New project*”.



Rysunek 16: Główne okno środowiska CLion

Po otwarciu nowego okna (Rysunek 17) trzeba wybrać po jego lewej stronie opcję „*C Executable*”.

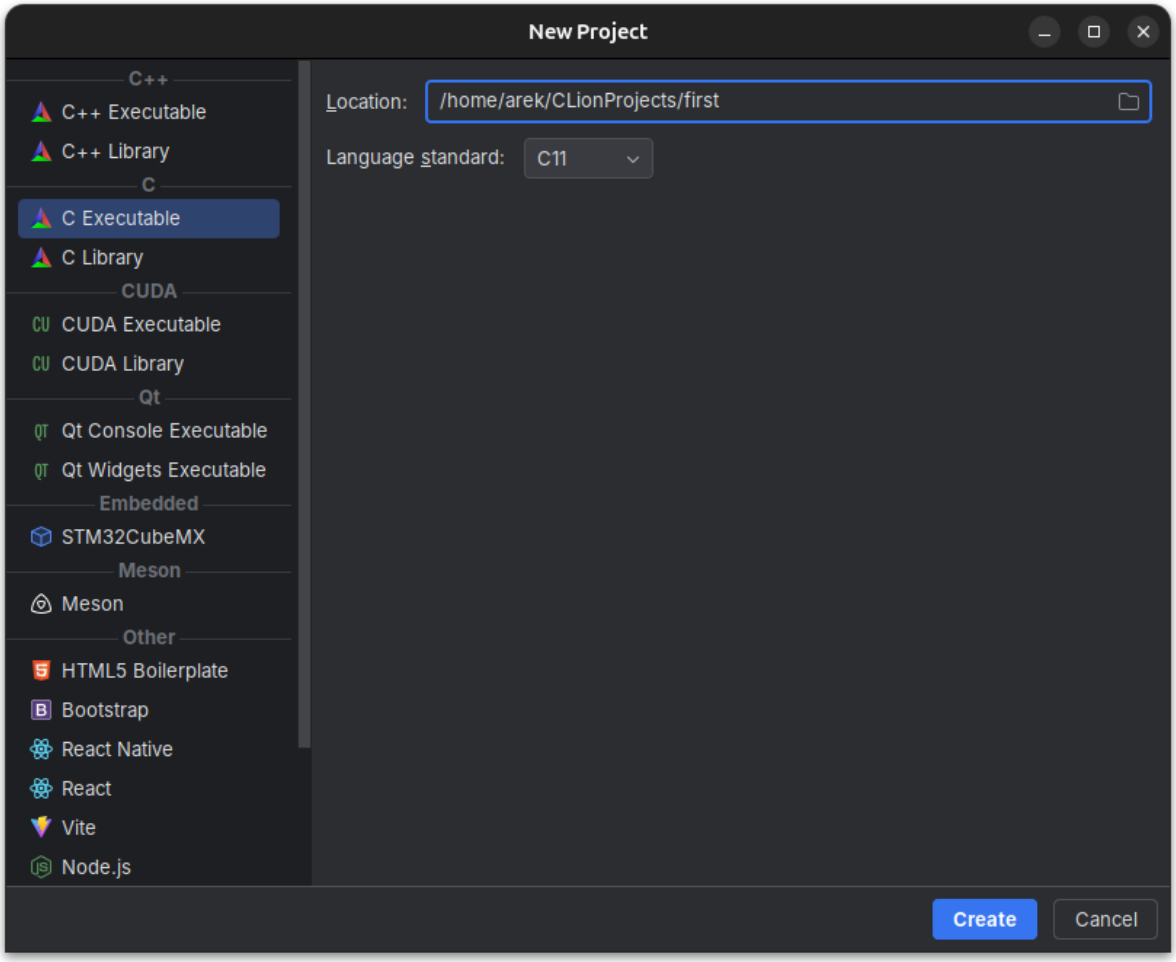
Następnie należy zmienić ostatnią nazwę w ścieżce znajdującej się u góry, po prawej stronie ekranu, chyba że domyślna nazwa nam nie przeszkadza.



Rysunek 17: Okno tworzenia projektu

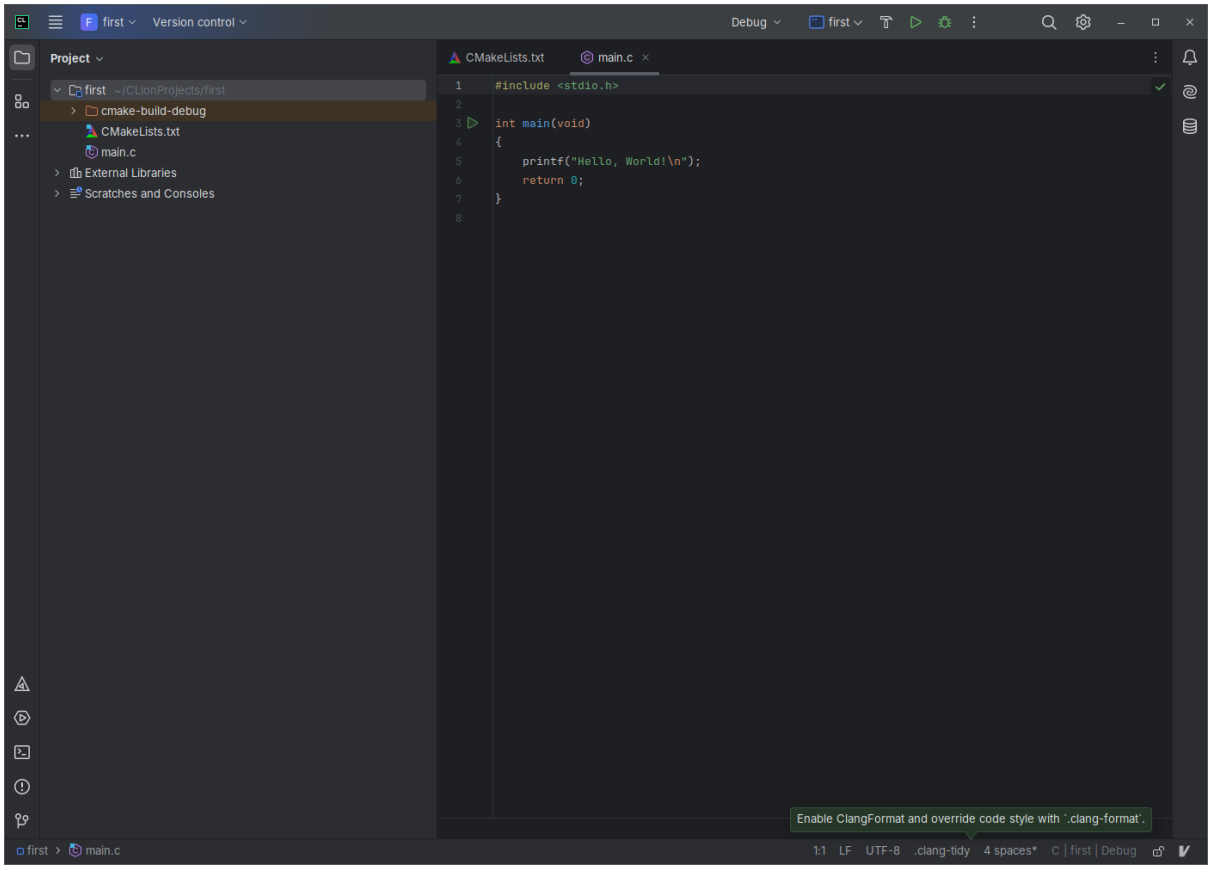
Untitled to domyślna nazwa, którą CLion nadaje nowemu projektowi oraz podkatalogowi w katalogu *CLionProjects*, gdzie ten projekt będzie zapisany.

Proponuję zmienić tę nazwę na inną, na przykład *first*, tak jak pokazano to na Rysunku 18.



Rysunek 18: Nadanie projektowi nowej nazwy

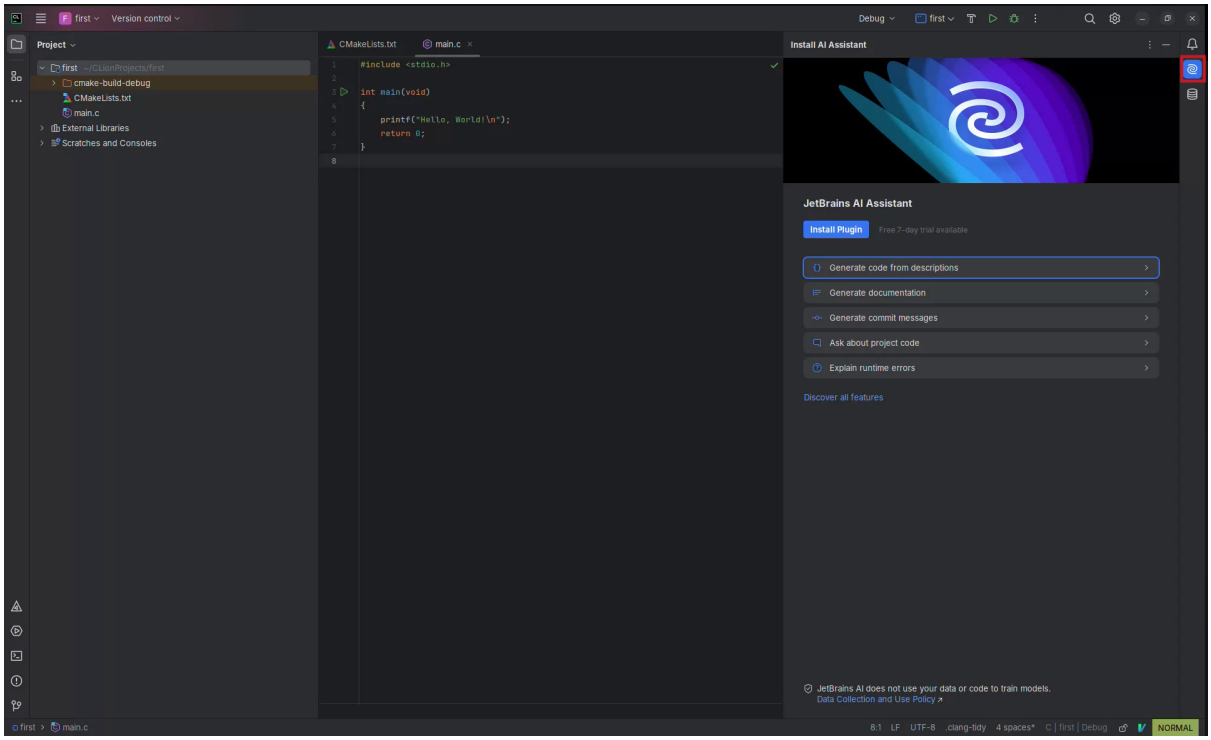
Po kliknięciu przycisku *Create* powinno pojawić się okno podobne do tego, jakie jest widoczne na Rysunku 19.



Rysunek 19: Okno robocze środowiska CLion

Jeśli w tym oknie widniałoby okno modalne z ustawieniami środowiska, to trzeba je zamknąć, klikając przycisk OK.

Z kolei okno sugerujące zainstalowanie wtyczki *AI Assistant* należy zamknąć przyciskiem znajdującym się po prawej stronie okna roboczego (Rysunek 20).



Rysunek 20: Zamknięcie okna dotyczącego wtyczki *AI Assistant*

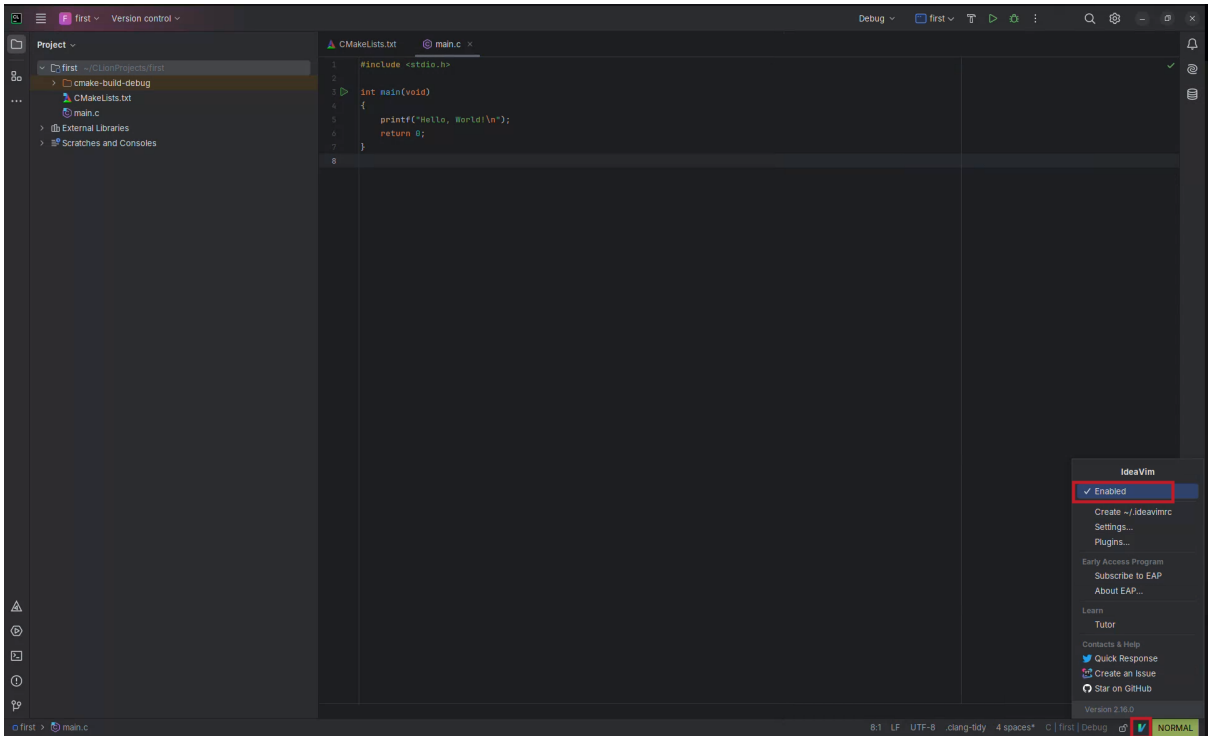
Podobnie należy postąpić, jeśli pojawi się okno z poradą dnia.

W tym ostatnim przypadku można, ale nie trzeba, zaznaczyć opcję zapobiegającą pojawieniu się tego okna przy ponownym uruchomieniu środowiska.

Osoby, które nie są zaznajomione ze skrótami klawiaturowymi edytora Vim, mogą wyłączyć także wtyczkę IdeaVim.

Ułatwia ona edycję kodu, ale dla początkujących jej obsługa może być trudna.

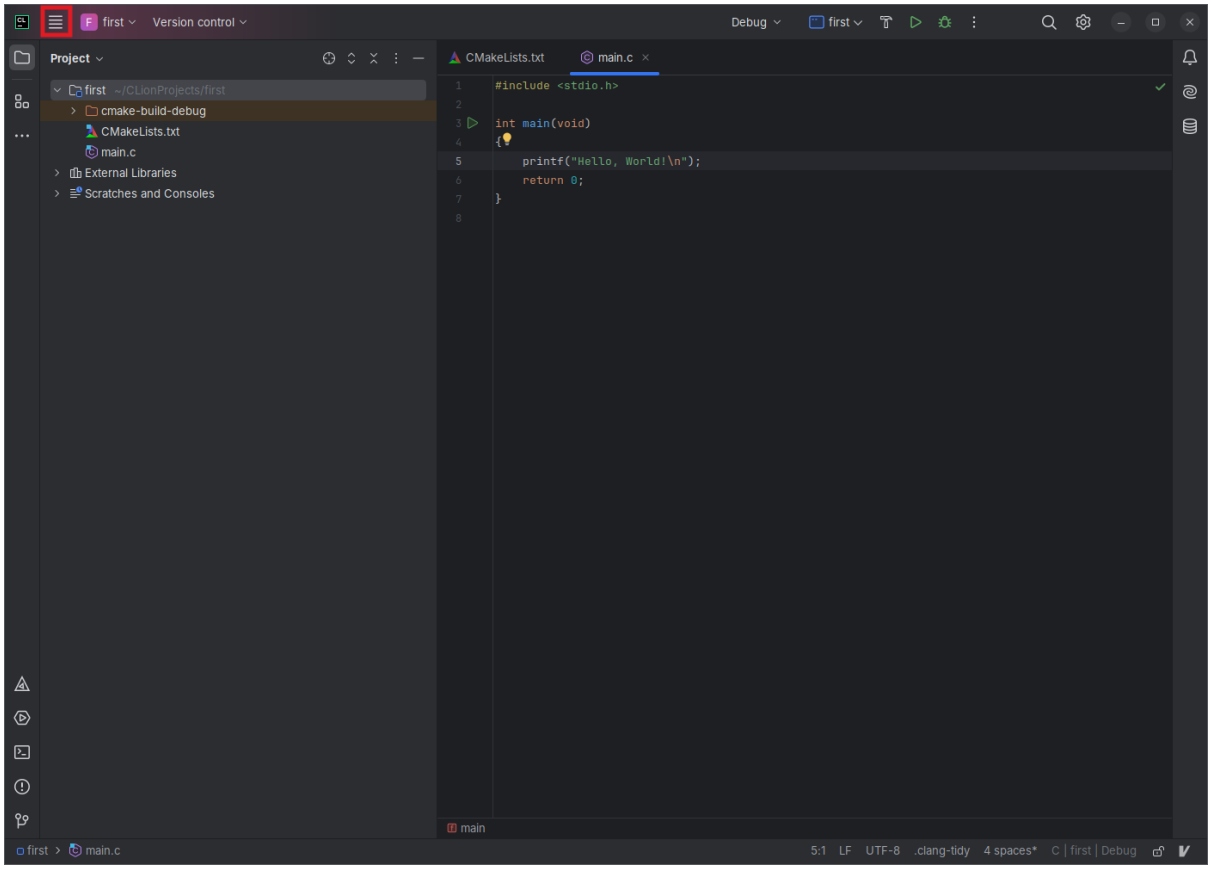
Aby zablokować jej działanie, należy kliknąć ikonę z kolorową literą V znajdującą się na dole okna roboczego, po prawej stronie, a następnie w menu kliknąć opcję „*Enable*” (Rysunek 21).



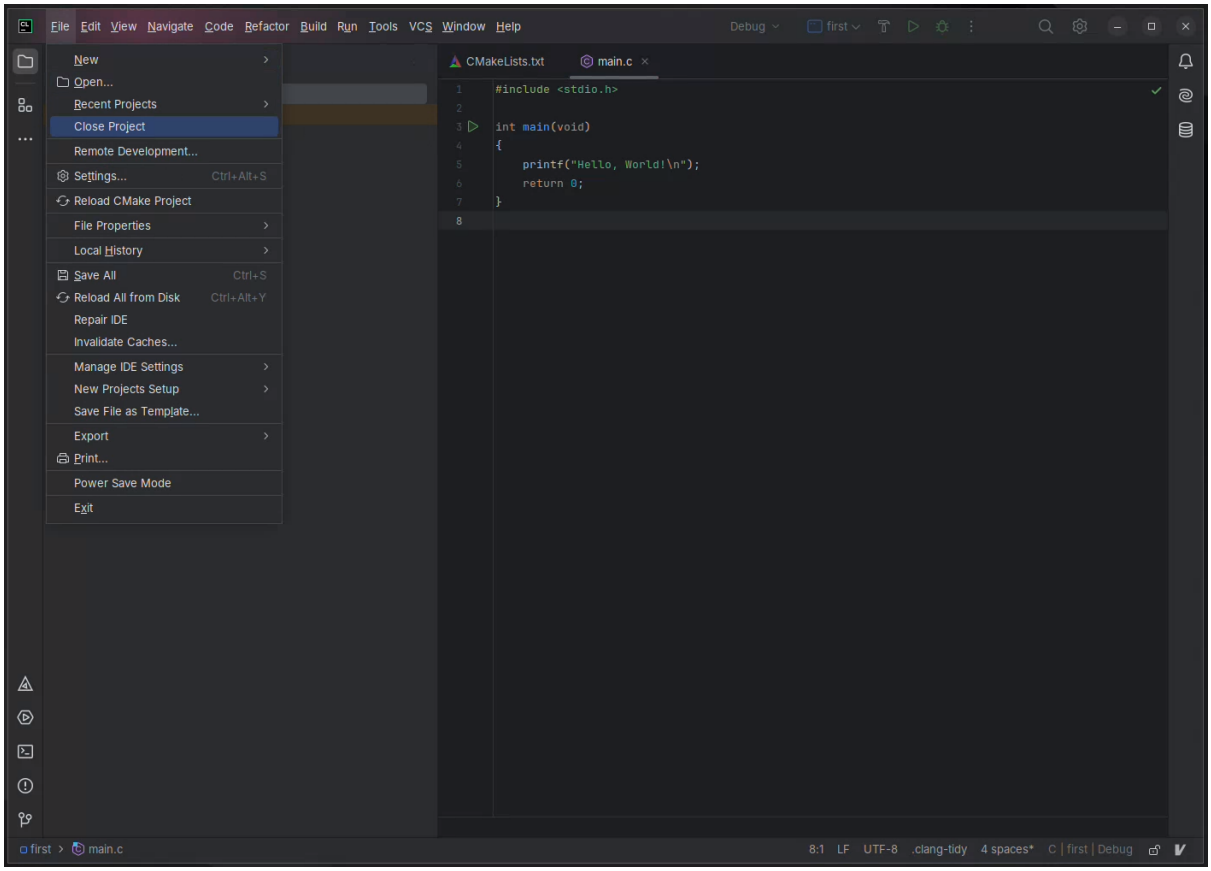
Rysunek 21: Wyłączenie wtyczki IdeaVim

Główną część okna roboczego zajmuje kod źródłowy zapisany w pliku `main.c`, przedstawiający program typu *Hello, World!*

Projekt można zamknąć, klikając przycisk głównego menu (Rysunek 22), a następnie wybierając opcję `File` → *Close project* (Rysunek 23).



Rysunek 22: Przycisk głównego menu

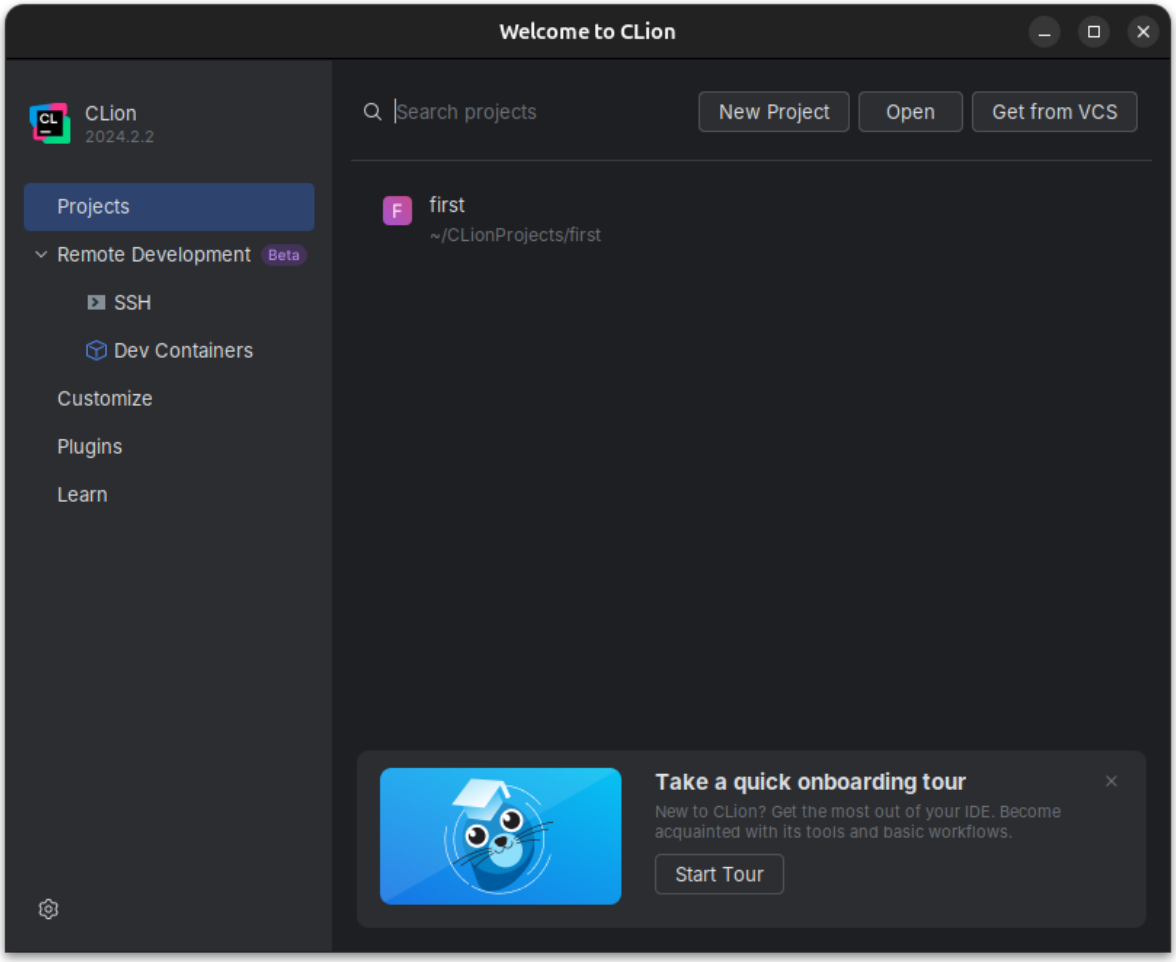


Rysunek 23: Opcja zamknięcia projektu w głównym menu

Po zamknięciu projektu pokaże się okno główne środowiska podobne do tego widocznego na Rysunku 24.

Proszę zwrócić uwagę, że teraz jest w nim widoczna lista, na której znajduje się zamknięty projekt. Umożliwia ona jego szybkie ponowne otwarcie.

Warto również zauważyć, że przycisk tworzenia nowego projektu jest nadal dostępny, choć ma nieco zmieniony wygląd.



Rysunek 24: Główne okno środowiska po zamknięciu projektu

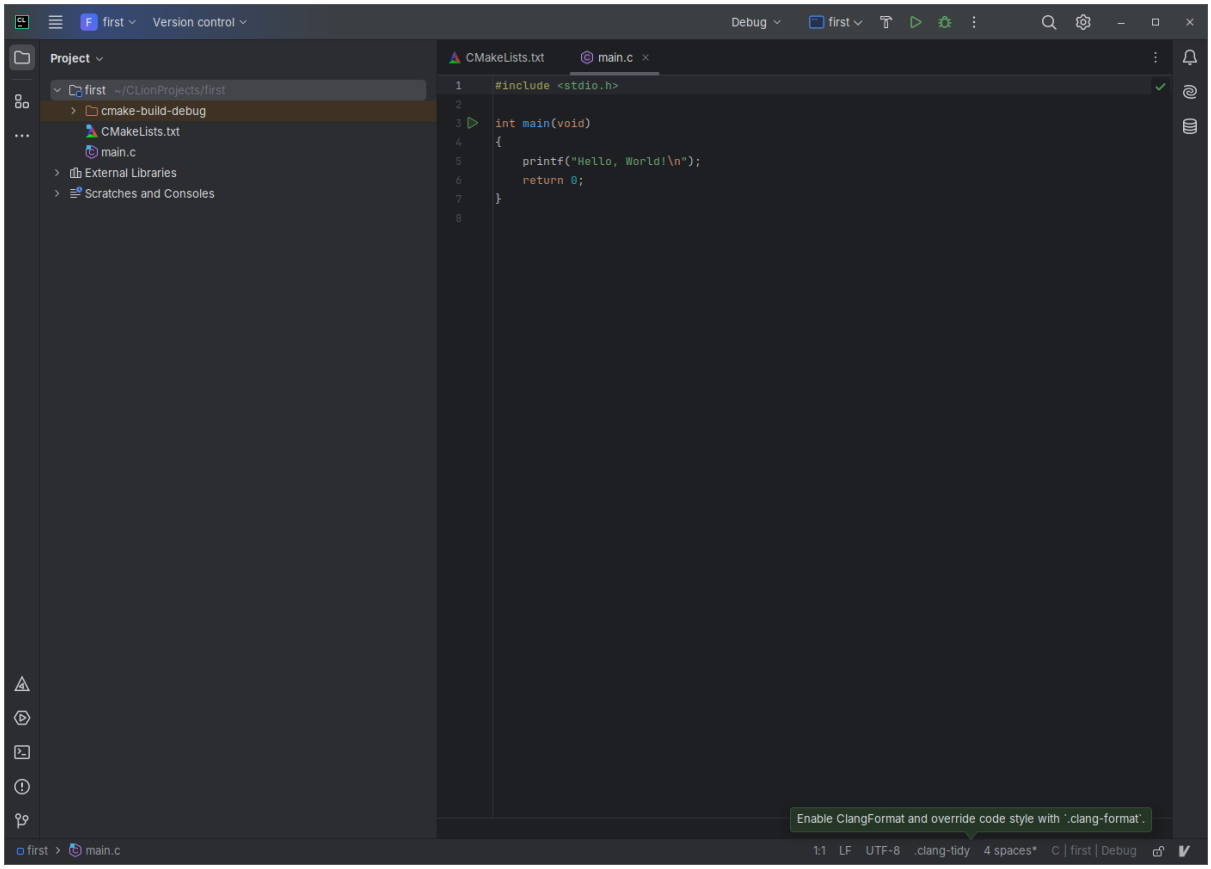
6. Kompilacja i uruchomienie programu

Widoczny na Rysunkach 19 i 25 tekst w centralnej części okna roboczego to zapis programu *Hello, World!* w języku C.

Jest on zrozumiały dla programisty, który zna ten język, ale nie bezpośrednio dla komputera.

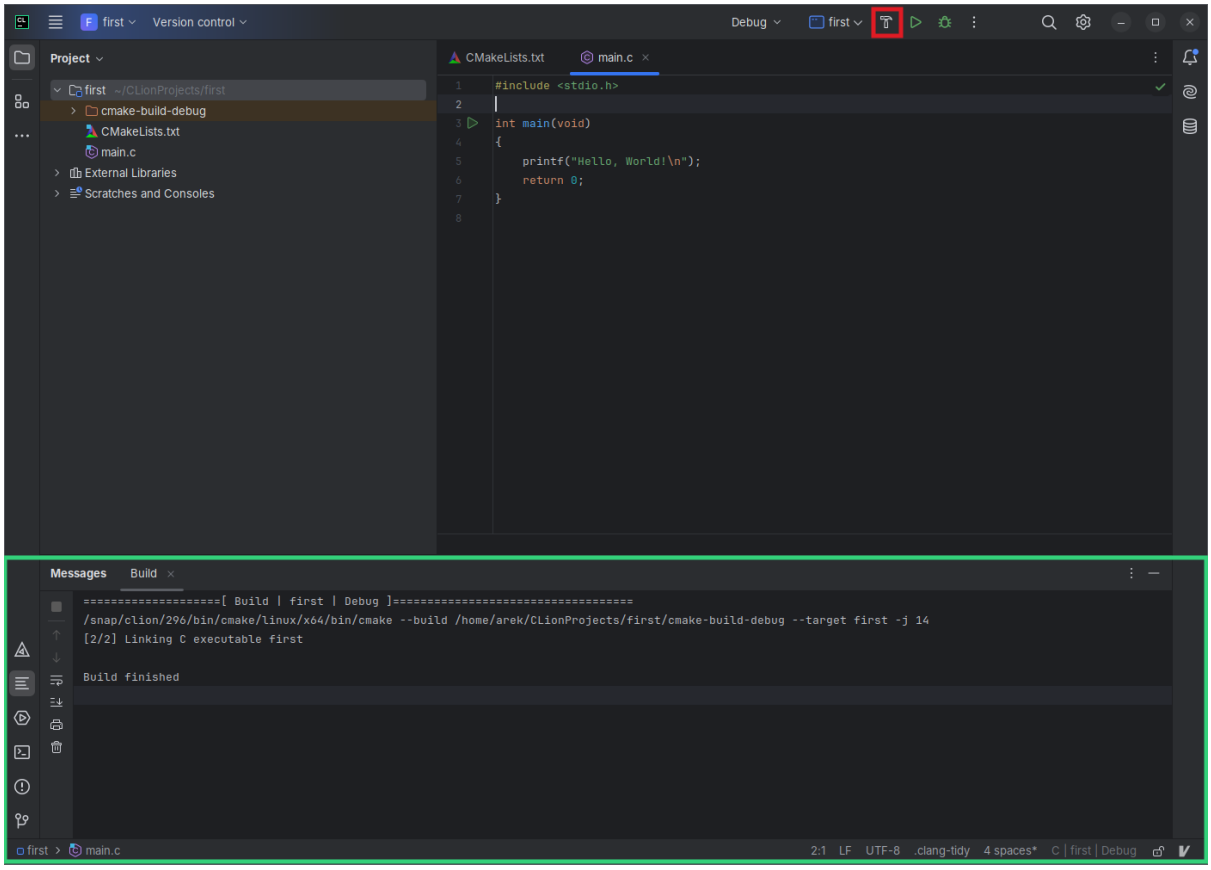
Zanim program zostanie wykonany, musi zostać przetłumaczony na język maszynowy przez kompilator.

CLion umożliwia uruchomienie tej operacji w bardzo prosty sposób.



Rysunek 25: Kod źródłowy programu w środowisku CLion

Wystarczy w tym celu kliknąć zaznaczony na Rysunku 26 przycisk z ikoną młotka.



Rysunek 26: Kompilacja programu w środowisku CLion

Informacje o przebiegu kompilacji zostaną umieszczone w oknie, które otworzy się na dole ekranu.

Tym razem kompilacja zakończyła się powodzeniem, czyli kompilatorowi udało się bez przeszkód przetłumaczyć kod źródłowy na maszynowy i zapisać wynik w pliku wykonywalnym, który został umieszczony w podkatalogu *first* katalogu *CLionProjects*.

Nie zawsze się jednak tak dzieje.

Kompilator może wygenerować dwa rodzaje informacji o napotkanych problemach.

Jeśli wykryje on jakieś rozbieżności z gramatyką języka, to może wygenerować ostrzeżenie (ang. *warning*).

Jest ono informacją od kompilatora, że napotkał niekrytyczny defekt w kodzie źródłowym, ale mimo to był w stanie utworzyć na jego podstawie postać wykonywalną programu.

Najczęściej problem ten związany jest z wieloznacznością lub nieścisłością zapisu fragmentu programu i skutkuje tym, że kompilator musi przyjąć pewne założenia, które nie muszą być zgodne z zamiarami programisty lub programistki.

W takiej sytuacji program wynikowy może działać inaczej, niż zakładał jego twórca.

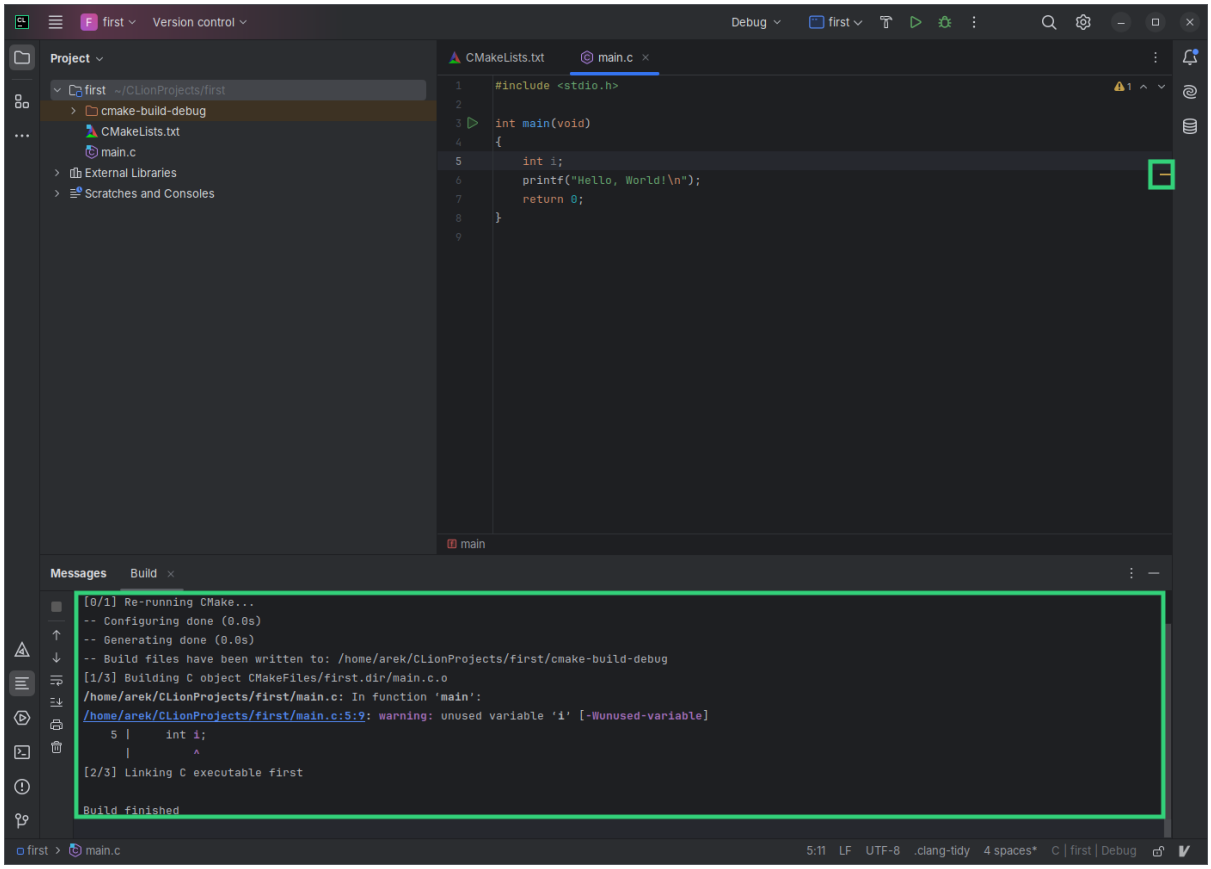
Innym rodzajem defektu, który może znajdować się w kodzie źródłowym, jest *błąd składni* (ang. *syntax error*).

Oznacza on całkowitą niezgodność zapisu fragmentu programu z gramatyką języka i powoduje przerwanie kompilacji.

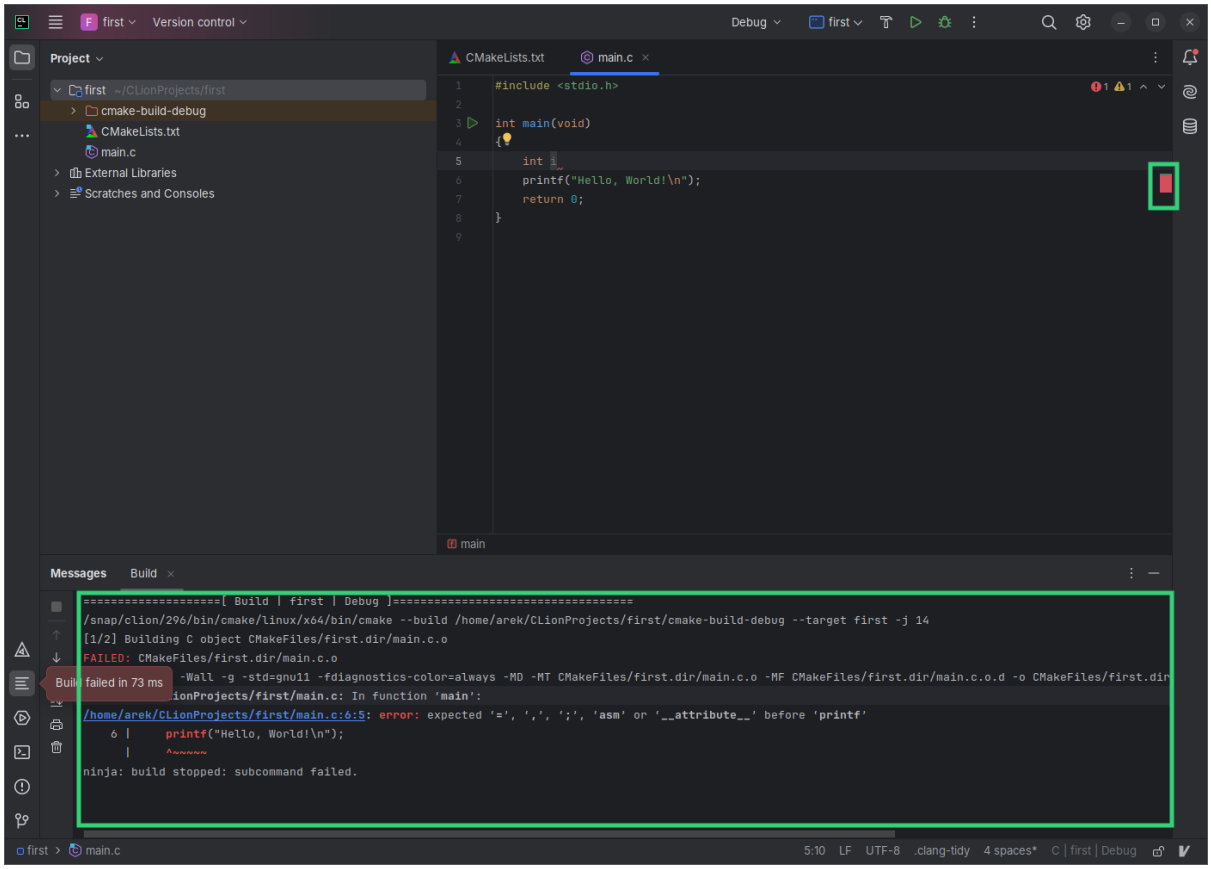
W takiej sytuacji plik z programem wykonywalnym, czyli kod wynikowy, nie powstaje, zatem programu w ogóle nie można uruchomić.

Wykonana w rozdziale 4 konfiguracja środowiska nakazuje kompilatorowi generowanie wszystkich podstawowych ostrzeżeń.

Dzięki temu można porównać, jakie komunikaty wyświetla CLion w przypadku, gdy kompilator zgłasza ostrzeżenie (Rysunek 27), a jakie, gdy znajdzie błąd składni (Rysunek 28).



Rysunek 27: Komunikat będący ostrzeżeniem kompilatora



Rysunek 28: Komunikat kompilatora o błędzie składni

Aby kompilator wygenerował ostrzeżenie, do programu została dodana deklaracja *zmiennej lokalnej*, czyli dostępnej tylko w funkcji `main()`, o nazwie `i`, która może przechowywać liczby całkowite.

Nie jest ona jednak wykorzystywana, stąd pojawiający się w dolnym oknie komunikat o nieużywanej zmiennej.

W jego treści pojawia się angielskie słowo „*warning*”, aby wyraźnie zaznaczyć, że kompilatorowi nie chodzi o błąd składni, a jedynie ostrzeżenie.

Proszę również zwrócić uwagę na żółtą kreskę zaznaczoną w oknie edycji na Rysunku 27, w wierszu[1] nr 5 i znajdujący się u góry okna znak ostrzegawczy.

Jeśli na tę kreskę zostanie skierowany kursor myszy, środowisko wyświetli podobny komunikat do tego pochodzącego od kompilatora, ale tym razem jego źródłem będzie edytor kodu.

CLion ma wbudowany mechanizm, który sprawdza kod źródłowy programu *w trakcie jego tworzenia* i *zanim* zostanie on skompilowany.

Jest to wygodne rozwiązanie, pozwalające wyeliminować określone defekty, zanim znajdzie je kompilator.

Na Rysunku 28 został do kodu źródłowego wprowadzony błąd składniowy, polegający na usunięciu średnika po deklaracji zmiennej lokalnej.

Według gramatyki języka C w poprawnym programie ten średnik *musi* się tam znajdować.

Jego brak powoduje przerwanie kompilacji i pojawienie się komunikatu o defekcie.

Tym razem w komunikacie użyte jest angielskie słowo *error*, co jednoznacznie wskazuje, że chodzi o błąd składni i że w związku z tym nie powstała wykonywalna wersja programu.

Proszę zwrócić uwagę, że również w tym przypadku edytor wykrył problem już wtedy, gdy kod źródłowy programu był tworzony.

Dlatego u góry okna edycji pojawił się znak „*stopu*”, a na końcu wiersza, gdzie występuje defekt, pojawił się wypełniony czerwony prostokąt.

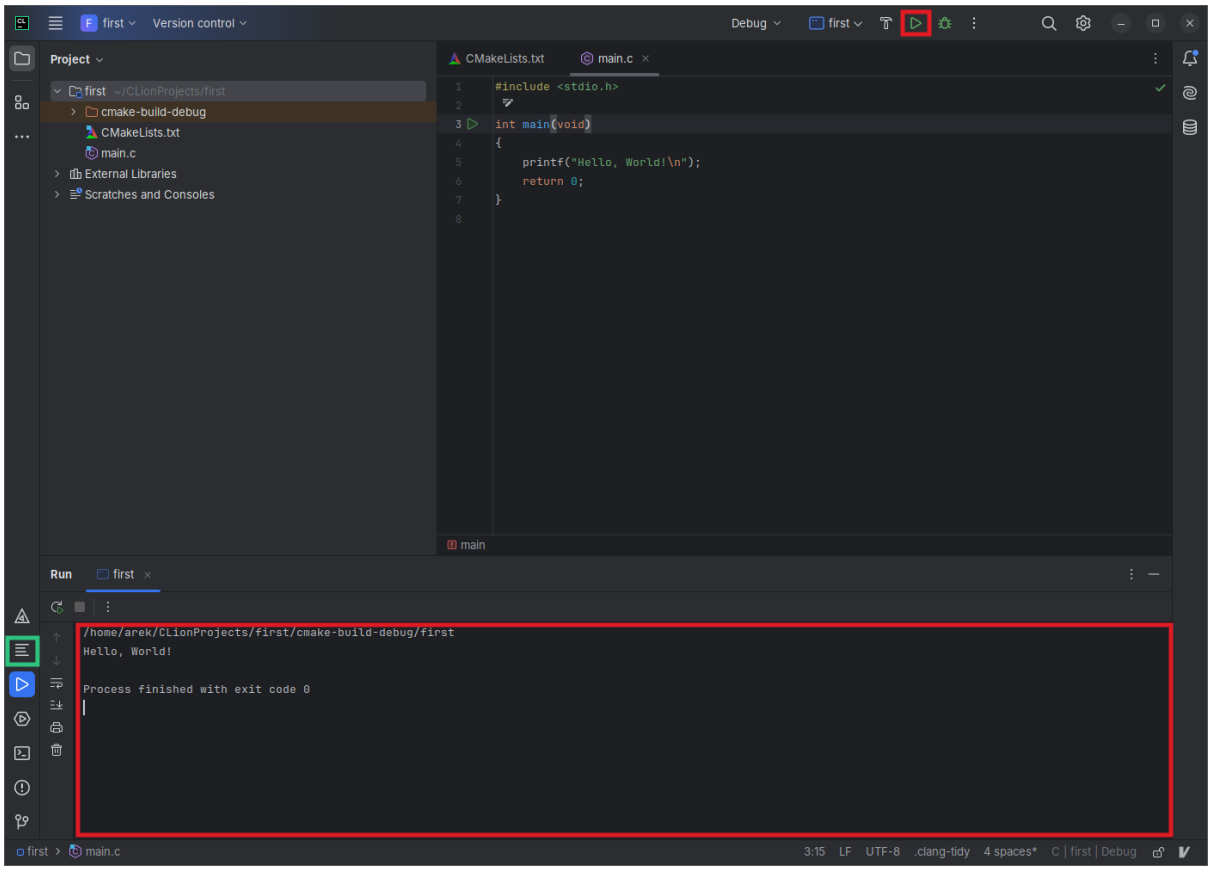
[1]Numery wierszy nie są częścią kodu źródłowego programu w języku C. Są one jedynie wyświetlane przez środowisko CLion, aby ułatwić pracę programistom.

Po najechaniu na niego wskaźnikiem myszy można zobaczyć komunikat podobny do tego, jaki wygenerował kompilator.

Opis sygnalizacji błędów składniowych i ostrzeżeń przez kompilator i środowisko CLion można podsumować następująco:

Warunkiem koniecznym, aby program działał prawidłowo, jest usunięcie z jego kodu źródłowego wszystkich defektów. Nie wolno także nigdy ignorować zgłaszanych przez kompilator ostrzeżeń!

Po przywróceniu kodu źródłowego do postaci z Rysunku 25 można program skompilować, a następnie uruchomić (Rysunek 29), klikając zieloną strzałkę znajdującą się nad oknem edycji kodu źródłowego lub używając kombinacji klawiszy `Shift + F10`.



Rysunek 29: Efekt uruchomienia programu

Wynik działania programu zostanie przedstawiony w dolnym oknie.

W tym przypadku jest to napis „Hello, World!”.

Jeśli programista lub programistka spróbuje uruchomić program, który wcześniej nie był kompilowany, środowisko wykryje brak pliku wykonywalnego i samo uruchomi kompilator przed wykonaniem programu.

Jest to wygodne rozwiązanie, skracające czas potrzebny do uruchomienia programu, ale ma ono także wady.

Korzystając z niego, programista lub programistka może „przegapić” informacje o ostrzeżeniach.

Dlatego warto w takim przypadku obejrzeć zawartość okna komunikatów kompilatora, do którego można uzyskać dostęp po kliknięciu przycisku zaznaczonego na prawej krawędzi Rysunku 29 na zielono.

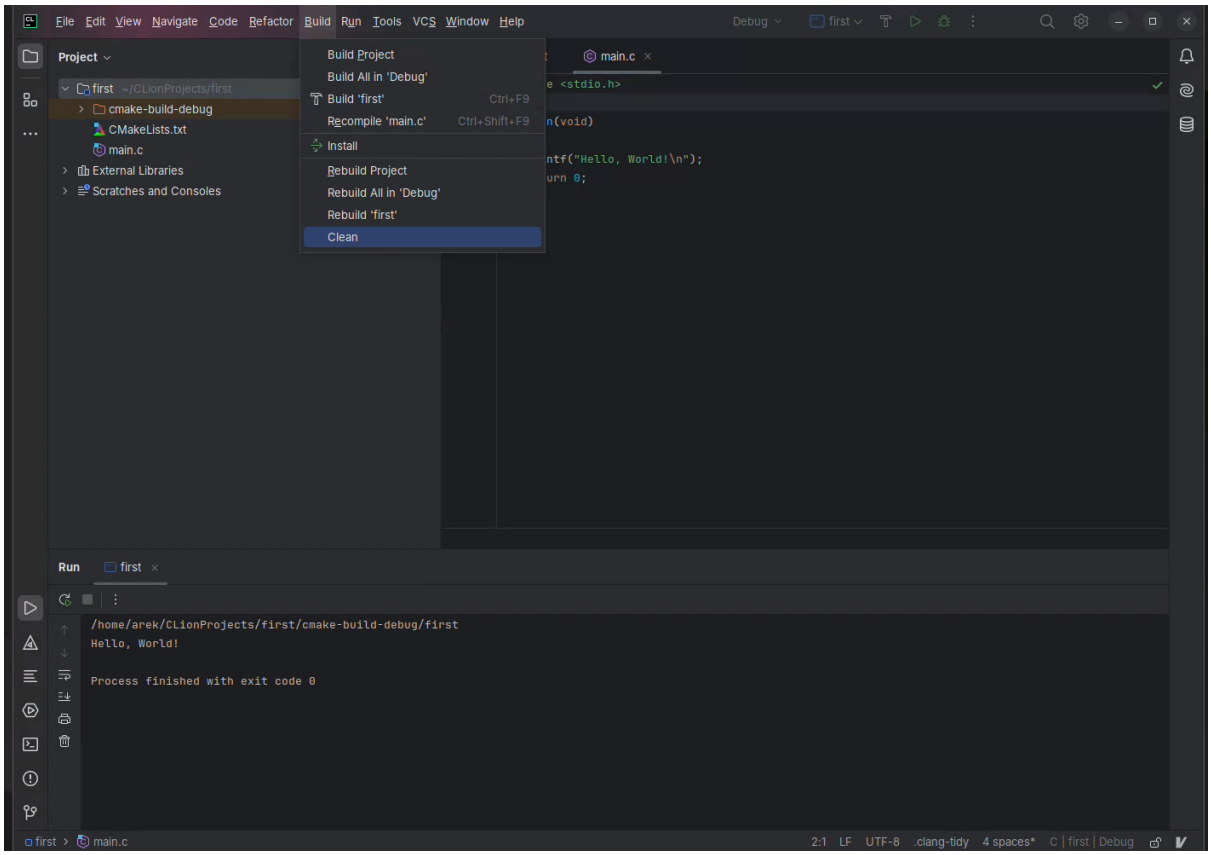
Po skompilowaniu i uruchomieniu programu w pamięci masowej, tj. na dysku twardym lub SSD, pozostaje plik wykonywalny.

Ma to tę zaletę, że jego ponowne uruchomienie nie wymaga kompilacji kodu źródłowego, o ile nie uległ on zmianie.

Z drugiej strony nie zawsze musi być to pożądane.

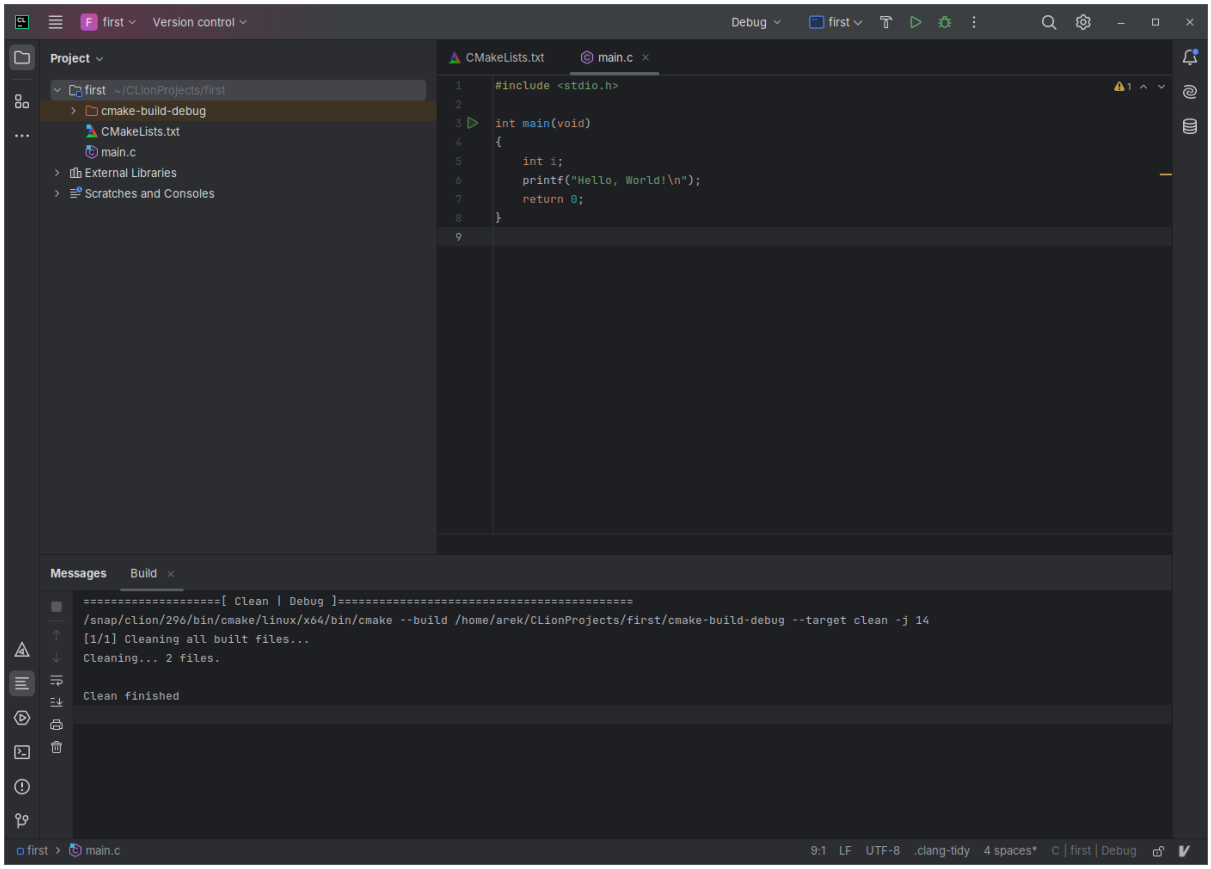
Dlatego środowisko CLion udostępnia możliwość usunięcia tego pliku i innych plików, które powstają w wyniku kompilacji.

Wystarczy w tym celu wybrać z menu głównego opcję `Build` → *Clean* (Rysunek 30).



Rysunek 30: Opcja usuwania efektów kompilacji

Po wykonaniu tej operacji w dolnym oknie środowiska pojawi się odpowiedni komunikat (Rysunek 31).



Rysunek 31: Komunikat o zakończeniu usuwania efektów kompilacji

7. Debugowanie programu

Przypomnijmy jeszcze raz: aby program działał poprawnie, należy poprawić wszystkie jego fragmenty, w których znajdują się defekty powodujące, że kompilator generuje komunikaty o błędach składniowych i ostrzeżeniach.

Jest to warunek konieczny, jednak niewystarczający.

Oprócz błędów składniowych w programach mogą występować także *błędy logiczne*.

Oznacza to, że nawet program napisany w sposób jednoznaczny i zgodnie z zasadami gramatyki języka programowania może działać nieprawidłowo.

Co gorsze, błędy logiczne jest o wiele trudniej znaleźć niż błędy składniowe.

Tych pierwszych kompilator programistom po prostu nie wskaże.

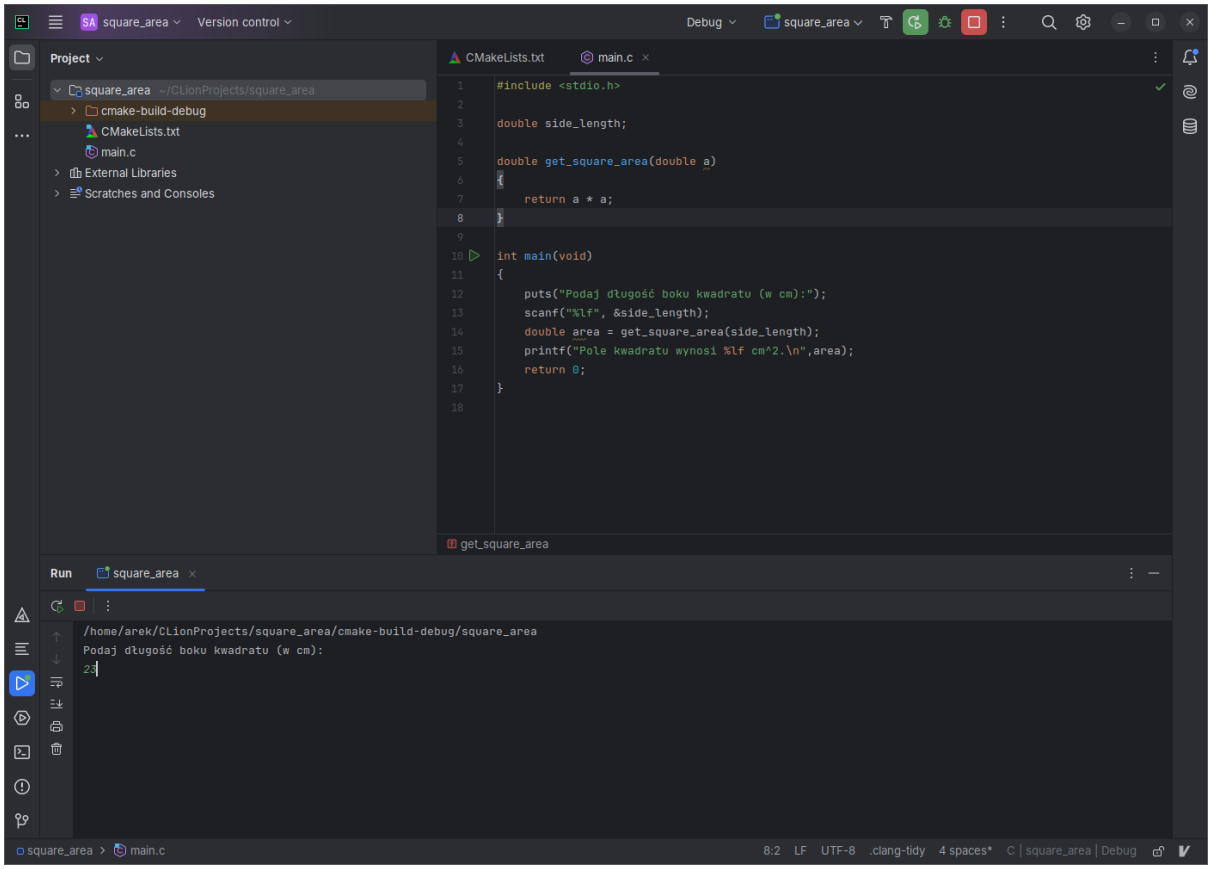
Na szczęście środowiska do programowania, takie jak CLion, udostępniają narzędzia, które pomagają w ich lokalizacji.

Jednym z najbardziej użytecznych jest *debugger*, czyli specjalne narzędzie, które pozwala wykonać program *krokowo*, tj. wiersz po wierszu, i badać jego stan.

Można w ten sposób obserwować, jak zmienia się stan zmiennych i w jakiej kolejności wykonywane są poszczególne fragmenty programu.

Działanie debuggera zostanie zaprezentowane na przykładzie programu, który liczy pole kwadratu o podanej przez użytkownika długości boku.

Jego kod źródłowy został zamieszczony na Rysunku 32.



Rysunek 32: Kod źródłowy analizowanego programu

Program ten jest trochę bardziej zaawansowany niż ten opisany w rozdziale 6.

Składa się on z dwóch funkcji: `get_square_area()` i `main()`.

W funkcji `main()` wywoływana jest funkcja `puts()`, będąca częścią biblioteki języka C, która wypisuje na ekranie komunikat proszący użytkownika o wprowadzenie liczby.

W kolejnym wierszu uruchamiana jest inna funkcja, o nazwie `scanf()`, która pobiera wartość wprowadzoną przy pomocy klawiatury i umieszcza ją w zmiennej globalnej `side_length`.

Następnie program wywołuje funkcję `get_square_area()`.

Jej działanie polega na pomnożeniu przez siebie wartości jej parametru o nazwie `a` i zwróceniu wyniku.

Pod ten parametr, w miejscu uruchomienia funkcji, podstawiana jest wartość zmiennej `side_length`.

Wynik działania funkcji jest zapisywany do zmiennej lokalnej `area`.

Następnie wartość `area`, wraz z odpowiednim komunikatem, jest wypisywana na ekranie.

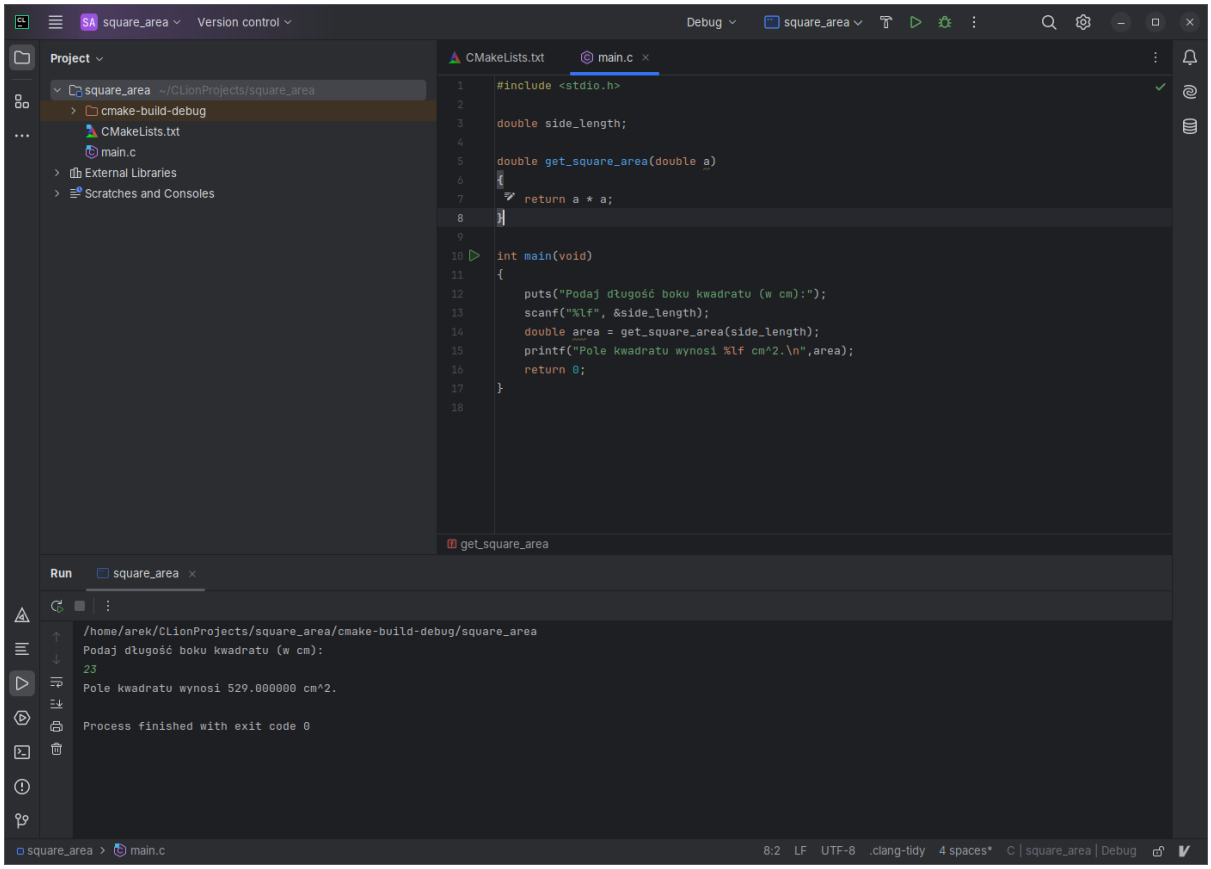
Po tym program kończy działanie.

Rysunek 32 przedstawia program w trakcie „zwykłego”, czyli niekrokowego wykonania.

Dokładniej jest to moment, w którym funkcja `scanf()` oczekuje na wprowadzenie przez użytkownika liczby i zakończenie tej operacji naciśnięciem klawisza `Enter`.

Wprowadzana liczba jest widoczna w dolnym oknie środowiska.

W tym samym oknie widoczny będzie także wynik obliczeń wykonywanych przez program (Rysunek 33).



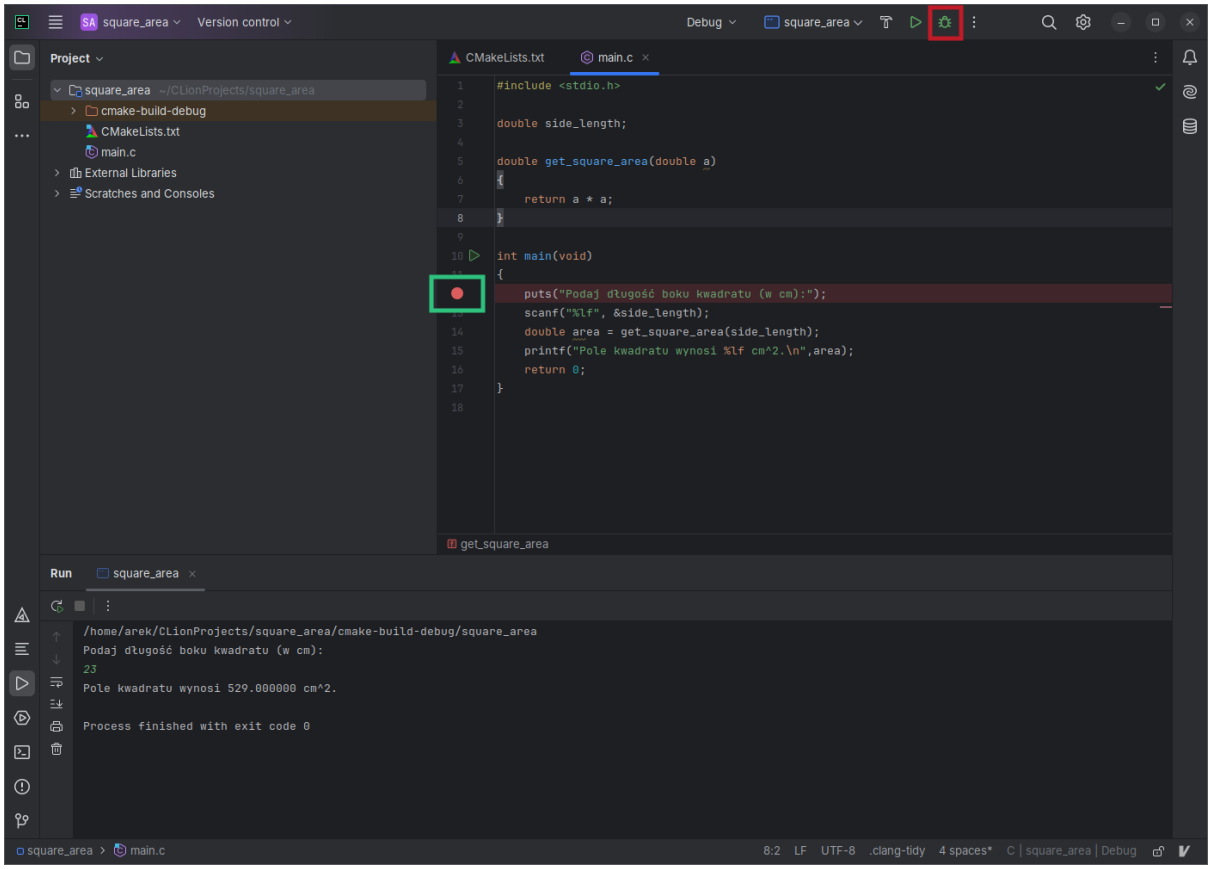
Rysunek 33: Wynik działania programu

Zanim zostanie uruchomiony tryb krokowy wykonania programu, należy w kodzie źródłowym wskazać miejsce, od którego to wykonanie ma się rozpocząć.

To miejsce nazywa się *pułapką* (ang. *breakpoint*).

W przypadku środowiska CLion wystarczy kliknąć numer wiersza, gdzie ta pułapka ma zostać założona (Rysunek 34).

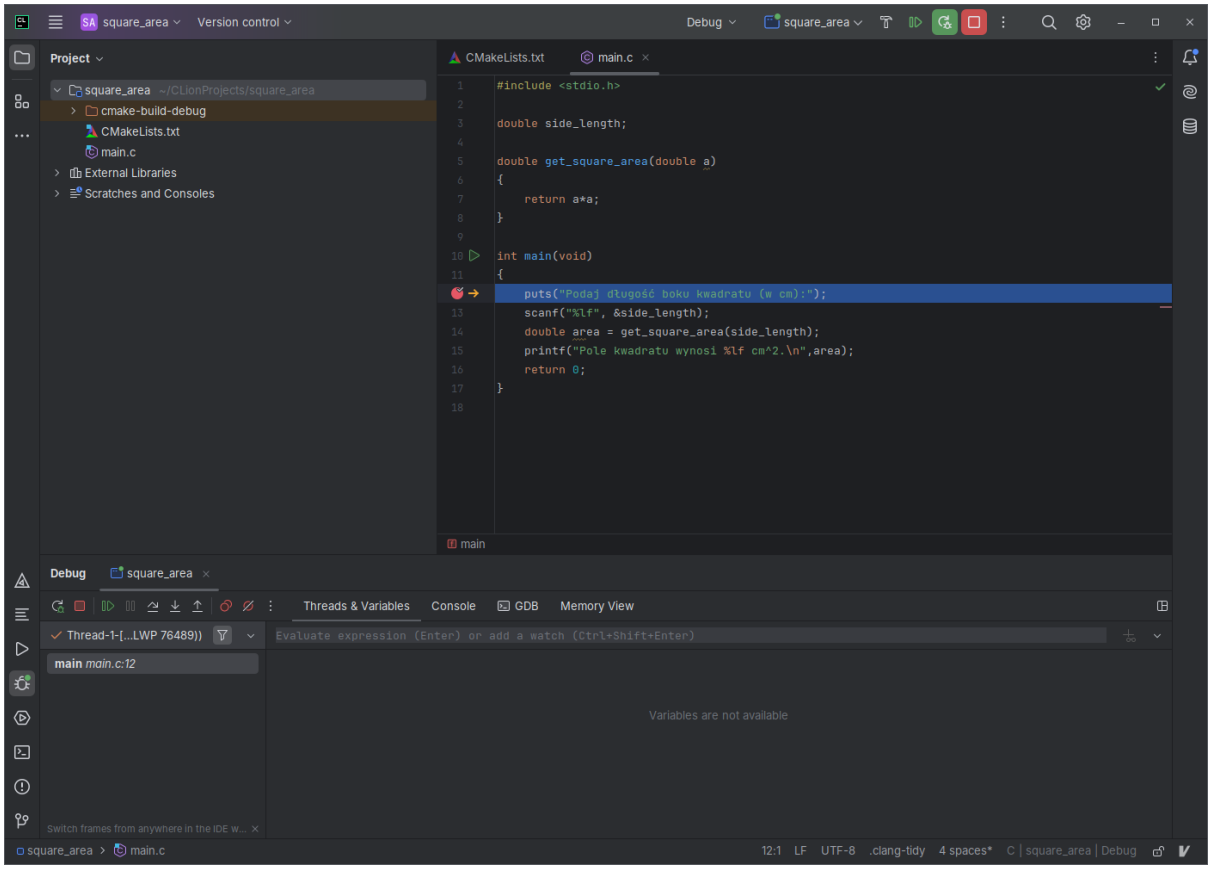
W tym miejscu pojawi się czerwona kropka symbolizująca pułapkę.



Rysunek 34: Zakładanie pułapki

Krokowy tryb pracy w środowisku CLion można włączyć, klikając zielony przycisk z symbolem owada[2] (Rysunek 35) lub używając kombinacji klawiszy `Shift + F9`.

[2] Angielskie słowo *bug* oznacza pluskwę, ale w żargonie programistów jest używane na określenie błędu logicznego w programie. Z kolei wyraz *debugging* oznacza dosłownie „odpluskwianie”, czyli usuwanie błędów logicznych.



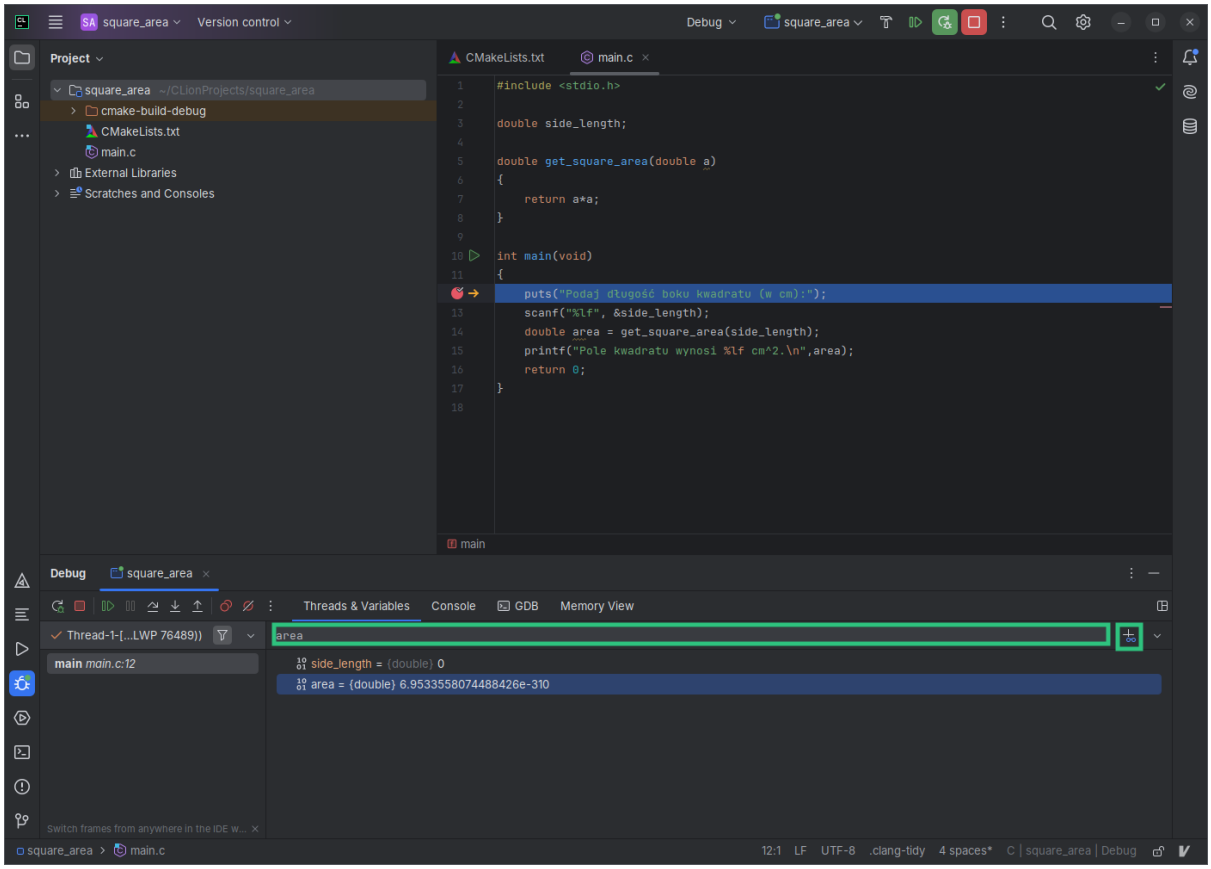
Rysunek 35: Rozpoczęcie wykonania krokowego

Jak można się przekonać, patrząc na Rysunek 35, wykonanie programu zatrzyma się dokładnie w miejscu ustawienia pułapki.

Teraz w dolnym oknie można dodać nazwy zmiennych lub całe wyrażenia, których wartości powinny być obserwowane w trakcie wykonania krokowego.

Przykładowo mogą to być `side_length` i `area`.

Nazwę każdej z tych zmiennych trzeba wprowadzić osobno w polu tekstowym znajdującym się u góry okienka i zatwierdzić znajdującym się po prawej stronie przyciskiem z plusem (Rysunek 36).



Rysunek 36: Wskazanie zmiennych do obserwacji

Warto zwrócić uwagę, że zmienna globalna (`side_length`) ma wartość początkową równą 0,0, a wartość zmiennej lokalnej (`area`) jest przypadkowa.

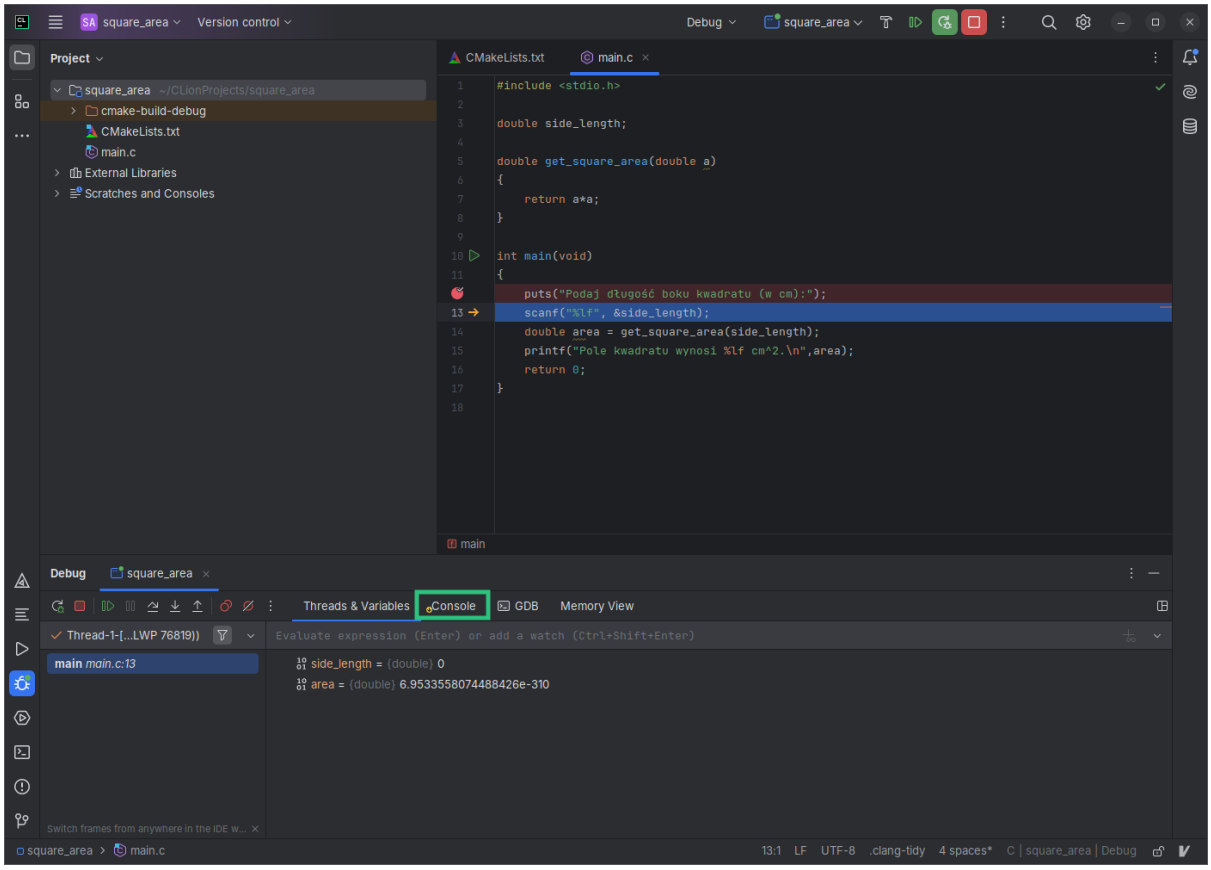
Ponadto nie ma konieczności ręcznego dodawania zmiennych lokalnych do listy obserwowanych.

CLion zrobi to automatycznie w chwili rozpoczęcia wykonania funkcji, w której te zmienne zostały zadeklarowane.

Aby wykonać dalej program, ale w trybie krokowym, należy nacisnąć klawisz `F8`.

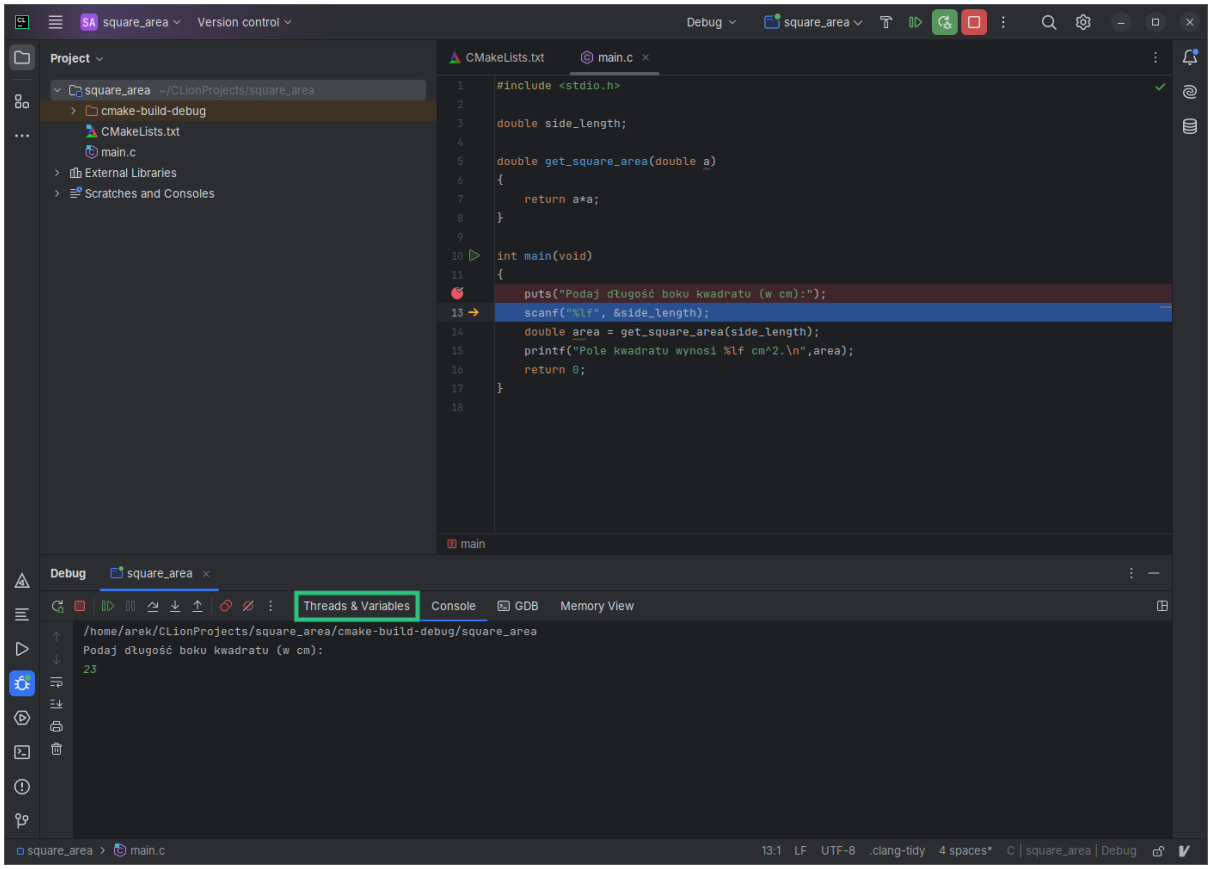
Debugger wykona kolejny wiersz programu.

Jest to wywołanie funkcji `scanf()`, odczytującej wartość z klawiatury, dlatego należy przełączyć się na zakładkę „*Console*” i wprowadzić odpowiednią liczbę (Rysunek 37).



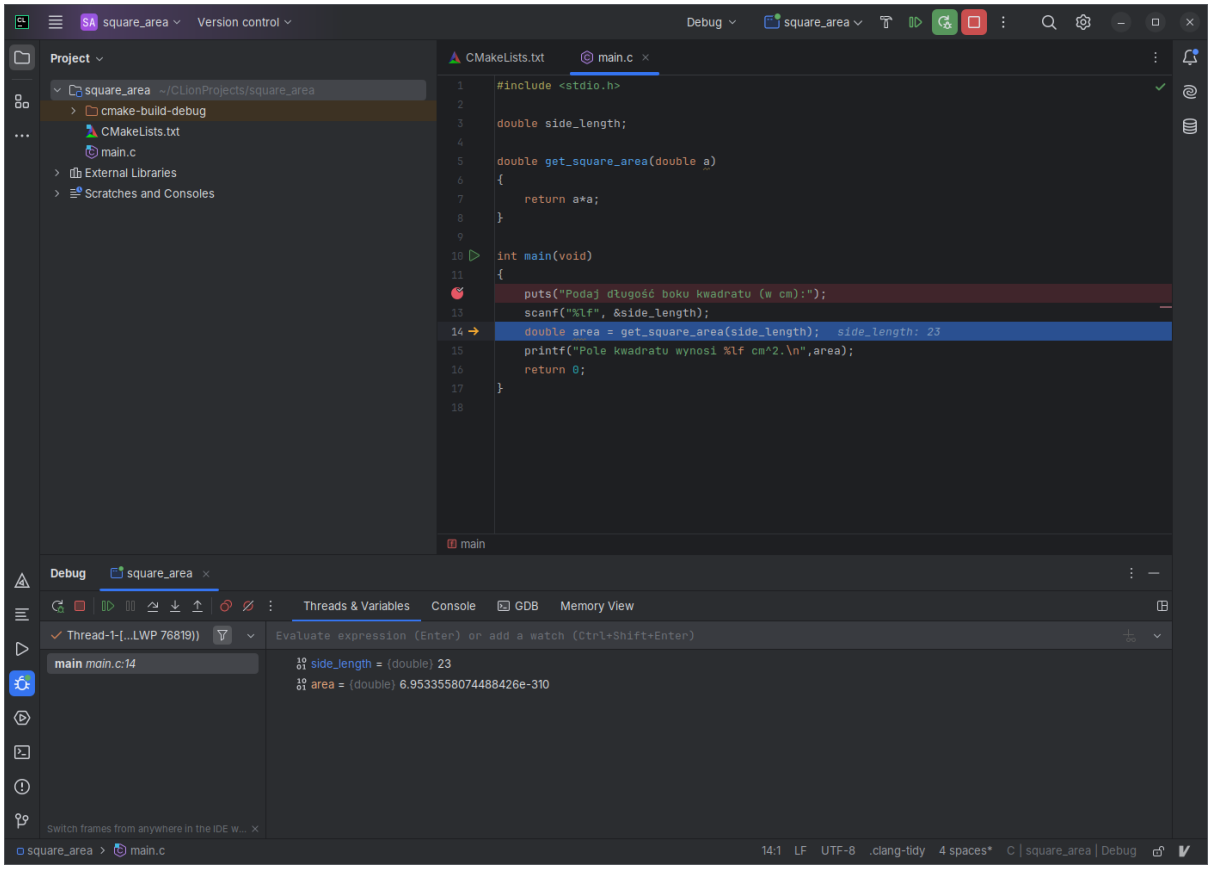
Rysunek 37: Przełączenie na konsolę w trybie krokowym

Następnie należy z powrotem przełączyć się na zakładkę „*Threads & Variables*” (Rysunek 38).



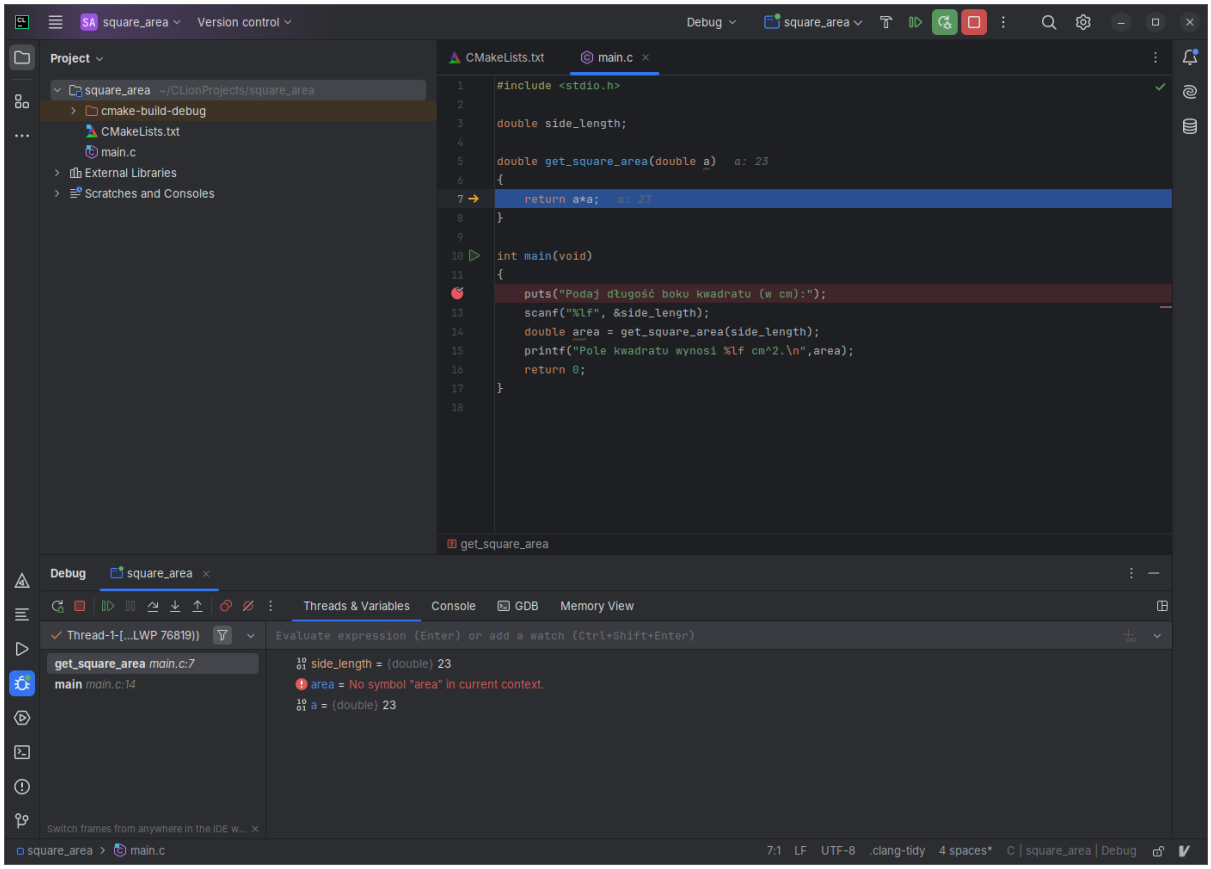
Rysunek 38: Wprowadzenie liczby w trybie krokowym

Gdy program zatrzyma się na wierszu, gdzie wywoływana jest funkcja `get_square_area()`, należy nacisnąć klawisz `F7` (Rysunek 39).



Rysunek 39: Miejsce wywołania funkcji `get_square_area()`

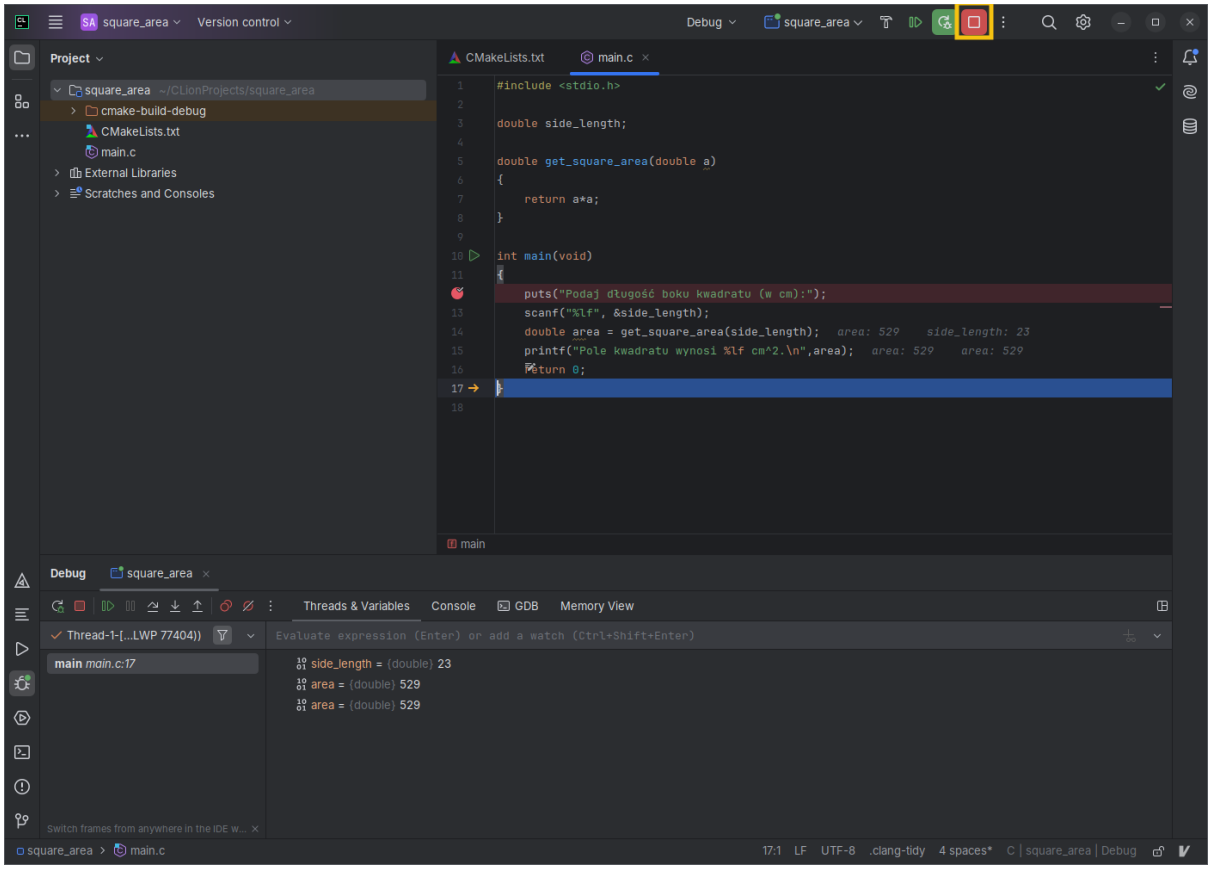
W przeciwieństwie do `F8`, klawisz `F7` pozwala obserwować wykonanie wnętrza funkcji (Rysunek 40). Klawisz `F8` spowodowałby wykonanie tej funkcji w jednym kroku.



Rysunek 40: Wykonanie wnętrza funkcji `get_square_area()`

Dalej można już posługiwać się klawiszem `F8`, wykonując program aż do momentu, gdy wykonana zostanie instrukcja `return 0;` w funkcji `main()`.

Osoby korzystające z CLion w systemie Linux mogą zakończyć krokowy tryb pracy programu, klikając czerwony przycisk znajdujący się nad oknem edytora (Rysunek 41).



Rysunek 41: Zakończenie krokowego wykonania programu

Jeśli środowisko działa w systemie Windows, operację tę można wykonać po prostu, naciskając klawisz `F8`.

W przypadku tego systemu operacyjnego komunikaty programu, na przykład wyświetlane funkcją `printf()`, mogą pokazać się na ekranie dopiero po zakończeniu działania programu.

Program wydaje się działać poprawnie, ale proszę wykonać zadanie 5.

8. Przykładowy program

Listing 1 zawiera kod źródłowy programu, w którym zdefiniowano funkcję o nazwie `calculate_area()`, obliczającą pole koła.

Funkcja ta ma jeden parametr wejściowy typu `double`, przez który przekazywana jest długość promienia, a zwraca wartość pola.

Typ wartości zwracanej i parametru określa, że obie te liczby będą liczbami rzeczywistymi[3].

Funkcja ta jest wywoływana w głównej funkcji programu o nazwie `main()`.

W tej funkcji program pyta użytkownika o długość promienia, a następnie przy pomocy funkcji `scanf()` zapisuje ją w zmiennej `radius`.

Jeśli długość promienia jest nieujemna, wartość tej zmiennej jest przekazywana do funkcji `calculate_area()`, która zwraca obliczoną wielkość pola do wywołania funkcji `printf()`.

Ta ostatnia wypisuje na ekranie komunikat, wielkość promienia koła oraz jego pole.

Jeśli promień miałby „ujemną długość”, program wypisze komunikat o błędzie.

Te dwa przypadki są rozróżniane dzięki zastosowaniu instrukcji warunkowej `if`.

[3]Dokładniej są to liczby zmiennoprzecinkowe, które stanowią dosyć dobrą reprezentację liczb rzeczywistych w pamięci komputera.

Uwaga! W kodzie programu celowo wprowadzono dwa błędy. Jeden z nich jest błędem składni i ujawnia się podczas kompilacji, a drugi jest błędem logicznym i ujawnia się w trakcie wykonania programu.

Listing 1: Kod źródłowy programu obliczającego pole koła o zadanym promieniu.

```c
#include <stdio.h>

#define PI 3.1415

double calculate_area(double r)
{
    return PI*r*r;
}

double radius;

int main(void)
{
    puts("Podaj długość promienia koła");

    scanf("%lf", &radius);

    // Program sprawdza, czy promień ma wartość
    // większą lub równą zero.
    if(radius <= 0.0)
        printf(
            "Pole koła o promieniu %f cm wynosi %f cm^2.\n",
            radius,
            calculate_area(radius)
        )
    else
        puts("Podana długość promienia jest nieprawidłowa.");

    return 0;
}
```

9. Zadania

1. Zapoznaj się z treścią tej instrukcji. Naucz się podstawowej obsługi środowiska CLion.
2. Popraw błąd składniowy w programie z listingu 1.
3. Znajdź i usuń błąd logiczny, który objawia się w trakcie wykonania programu. Użyj do tego techniki debugowania.
4. Sprawdź, jakie inne opcje ułatwiające programowanie oferuje środowisko CLion. Postaraj się nauczyć niektórych z nich.
5. Używając techniki debugowania sprawdź, czy program obliczający pole kwadratu zadziała prawidłowo, tzn. zgodnie z regułami matematyki, jeśli użytkownik wprowadzi ujemną długość boku.

   Jeśli nie, spróbuj ten problem naprawić, wzorując się na programie obliczającym pole koła.