Studio projektowe

Projekt kompilatora online

Autorzy:
Piotr Karaś
Patrycja Kopacz

Prowadzący
dr hab. Konrad Kułakowski

Projekt kompilatora online jest już w ostatniej fazie rozwoju. Według stworzonego wcześniej diagramu Gantta pozostało wykonanie testów sprawdzających (10.12.2018) oraz przygotowanie dokumentacji (17.12.2018).
Przez cały okres trwania projektu trzymaliśmy się wyznaczonych ram czasowych. Dla przypomnienia zadania, które mieliśmy wykonać podczas implementacji naszego projektu były następujące:

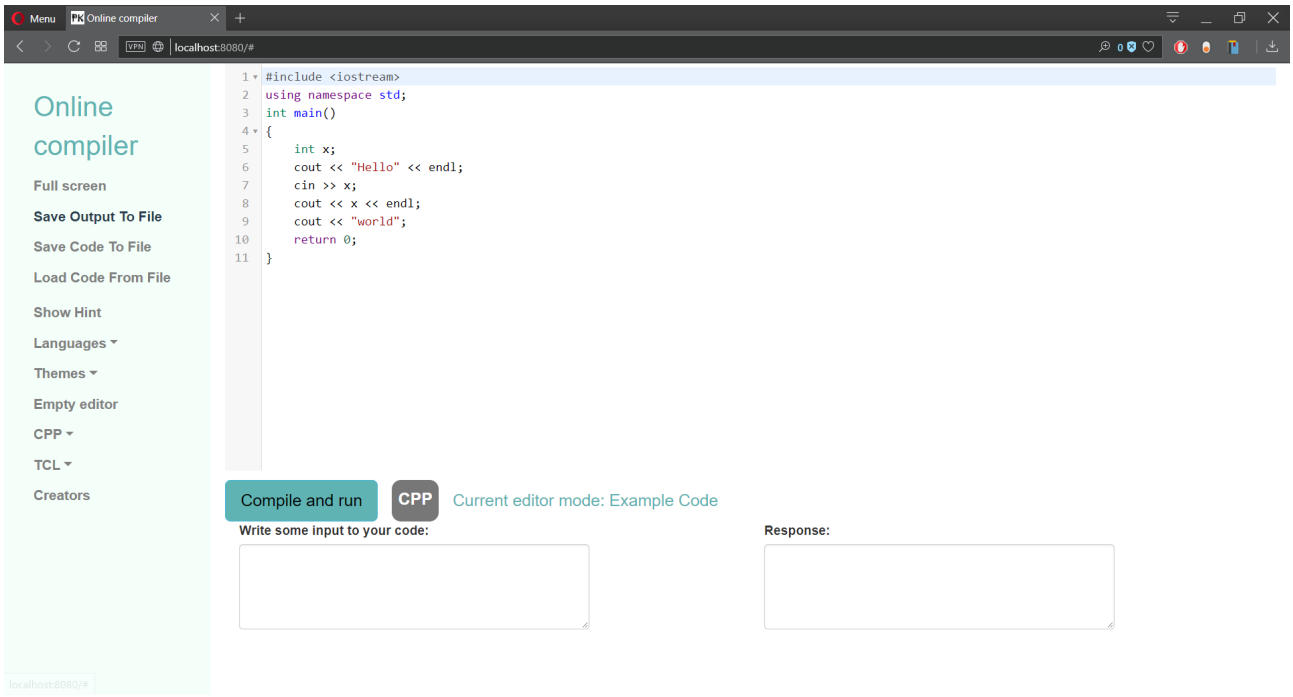
- Stworzenie prostego serwera REST-owego
- Podstawowy wygląd strony
- Komunikacja między frontendem a serwerem
- Kompilacja i uruchomienie przesłanego kodu
- Prosty edytor (podświetlanie składni)
- Zapewnienie wielowątkowości
- Wyświetlanie zadań
- Stworzenie przykładowych zadań wraz z opisem
- Poprawki wyglądu
- Zapewnienie wsparcia dla różnych języków

Wszystkie podpunkty zostały zrealizowane w wyznaczonych wcześniej terminach.

Kilka uwag:
Projekt sam w sobie nie jest jednak jeszcze przygotowany do skomercjalizowania ponieważ nie zapewnia on całkowitego bezpieczeństwa. Przede wszystkim użytkownik może wykonywać dowolny kod – jest ograniczony jedynie przez dostępne biblioteki i jeden plik – nie może on stworzyć całego projektu z wieloma plikami. Ze względu na to może on wywołać niepożądane zmiany na naszym serwerze. Aby projekt mógł być wykorzystywany najlepiej byłoby zwirtualizować system na którym on działa – aby w razie złośliwości lub niedopatrzenia użytkowników zniszczeniu uległ nic nie znaczący system, bez zmian w fizycznym środowisku. Do tego celu można by było użyć choćby Dockera, VirtualBoxa czy też inne narzędzie.
Po drugie należałoby dodatkowo uruchamiać nasz serwer, a przede wszystkim programy napisane przez użytkowników w bezpiecznych środowiskach – sandboxach (piaskownice), które zapewniałyby, że nie wykonają się żadne akcje, które mogłyby modyfikować pliki na dysku, uruchamiać dodatkowe zadania i wszelkie inne destrukcyjne czynności.

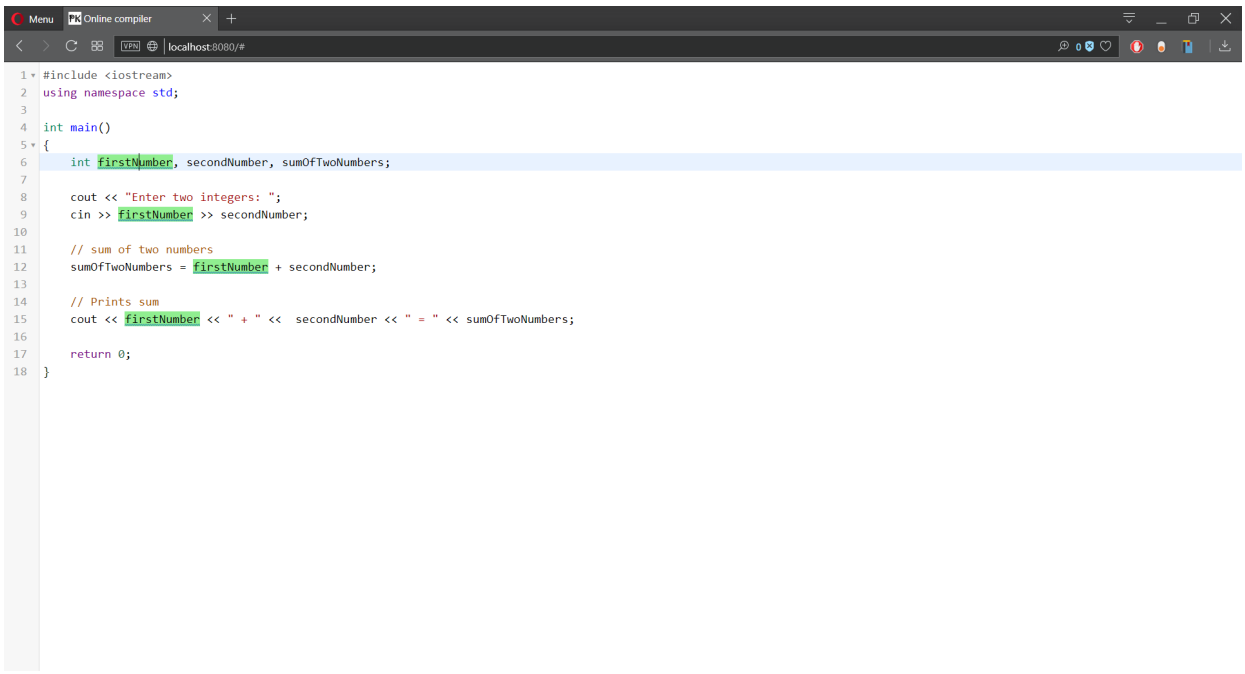
Działanie systemu

Po uruchomieniu serwera i wejścia przez przeglądarkę na udostępnianą stronę internetową pojawia się następujący widok:

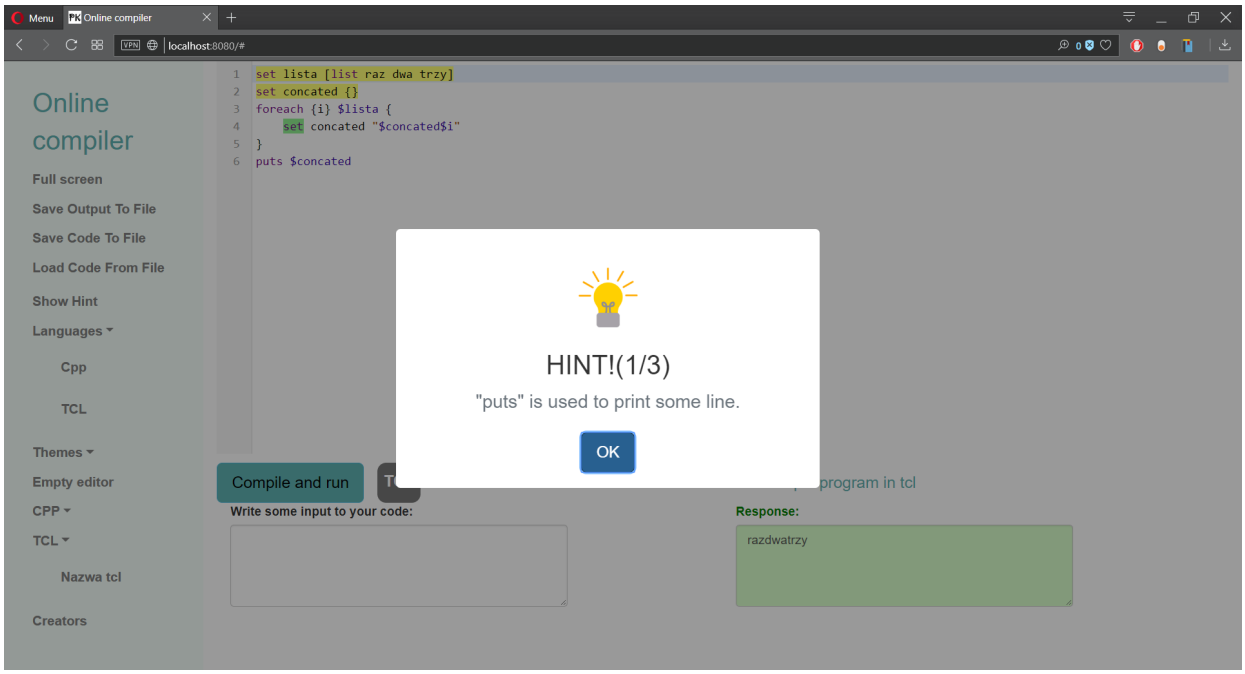


Omówienie kontrolek:

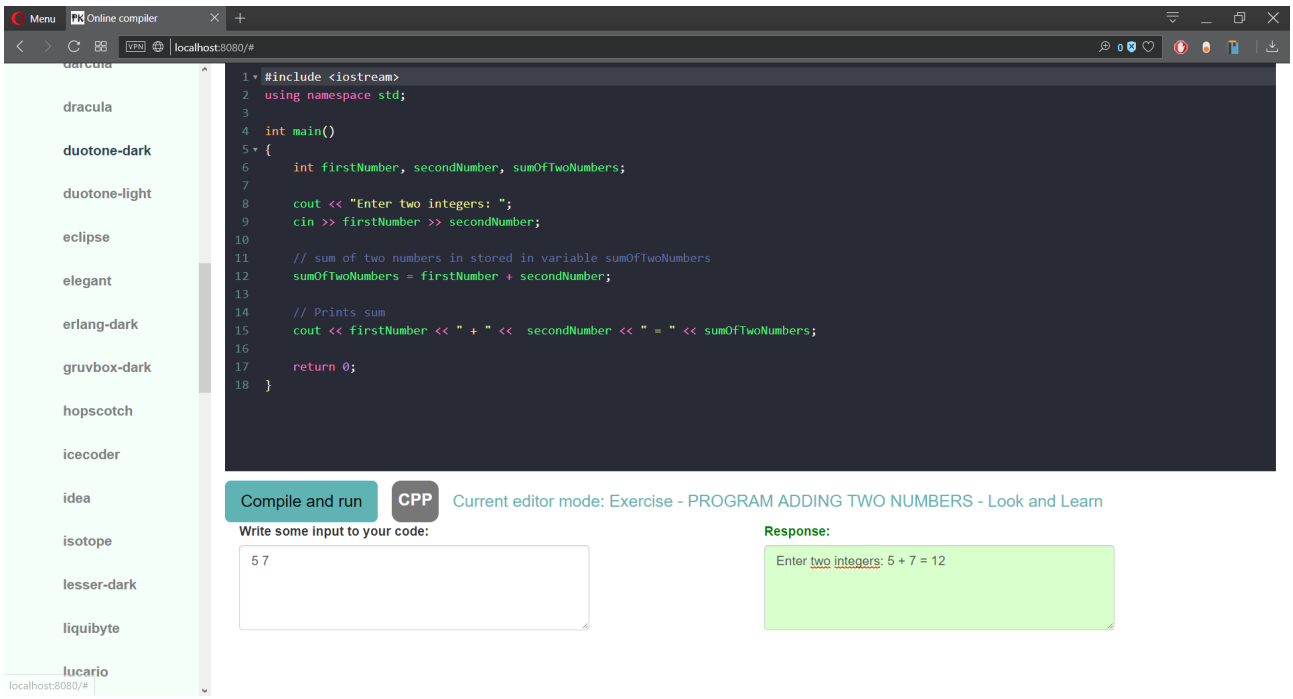
- Full screen – tak jak zostało założone użytkownik powinien mieć udostępnione *wygodne* narzędzie do pisania kodu, dlatego ma on możliwość wejścia w tryb pełnoekranowy po kliknięciu na ten przycisk. Aby z niego wyjść należy wcisnąć Esc.



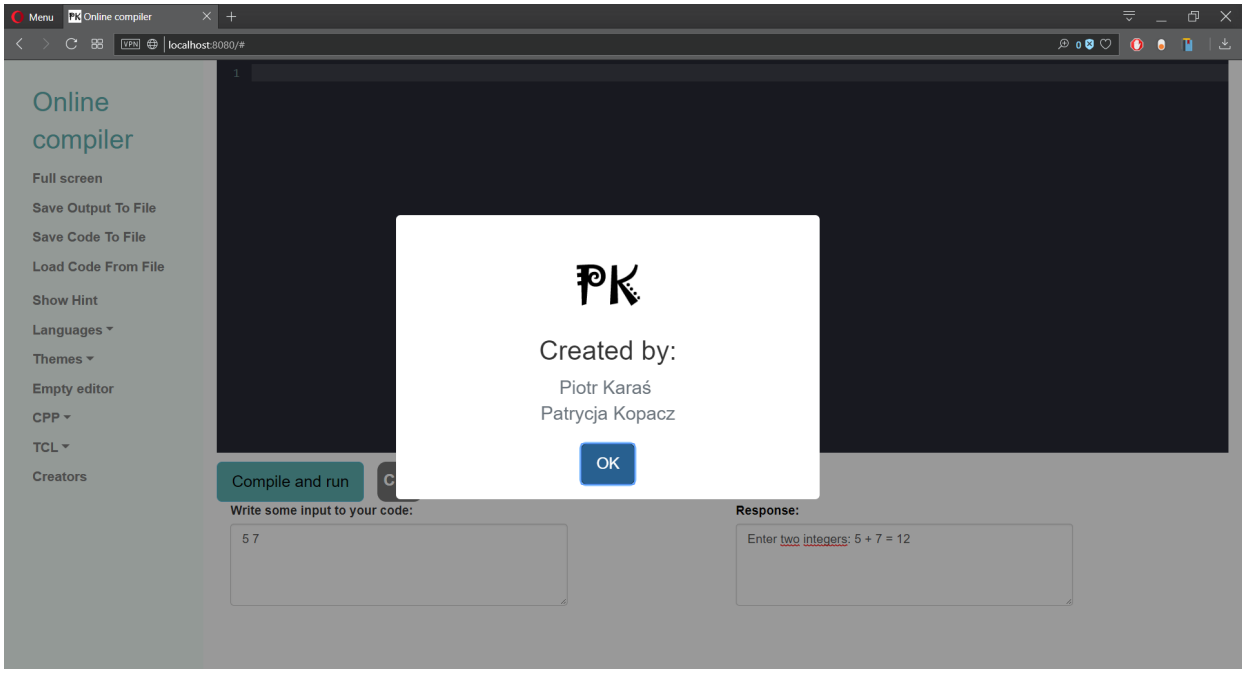
- Save output to file – użytkownik może chcieć zapamiętać wynik swojego programu – po kliknięciu w ten przycisk może go wygodnie pobrać i zapisać do pliku na swoim dysku.
- Save code to file – podobnie rzecz ma się dla całego kodu. Po skończonej pracy, aby nie stracić napisanego kodu można go pobrać i zapisać na swoim dysku.
- Load code from file – aby oszczędzić czasu przy ponownym ładowaniu kodu można go pobrać z pliku na dysku. Wstawi się on wtedy w miejsce aktualnego kodu. Po takiej operacji trzeba pamiętać aby odpowiednio ustawić język.
- Show hint – przy próbie napisania kawałka kodu w nowym języku czasem łatwo utknąć. Do pomocy przychodzą wtedy wskazówki. Po kliknięciu na ten przycisk pokazują się one w ładnej formie i pomagają użytkownikowi ruszyć dalej.



- Languages – można tutaj wybrać jeden z jak na razie dwóch dosyć różniących się języków programowania: C++ oraz Tcl. Po dokonanym wyborze zostanie odpowiednio podświetlona składnia, a przy kompilacji zostanie przekazany jakiego kompilatora/interpretera trzeba użyć.
- Themes – daje użytkownikowi możliwość ustawienia tła i sposobu podkreślania słów kluczowych. Wielu programistów i nie tylko woli używać ciemnego tła i jasnej czcionki do codziennego użytku, ze względu na mniejsze zmęczenie wzroku.



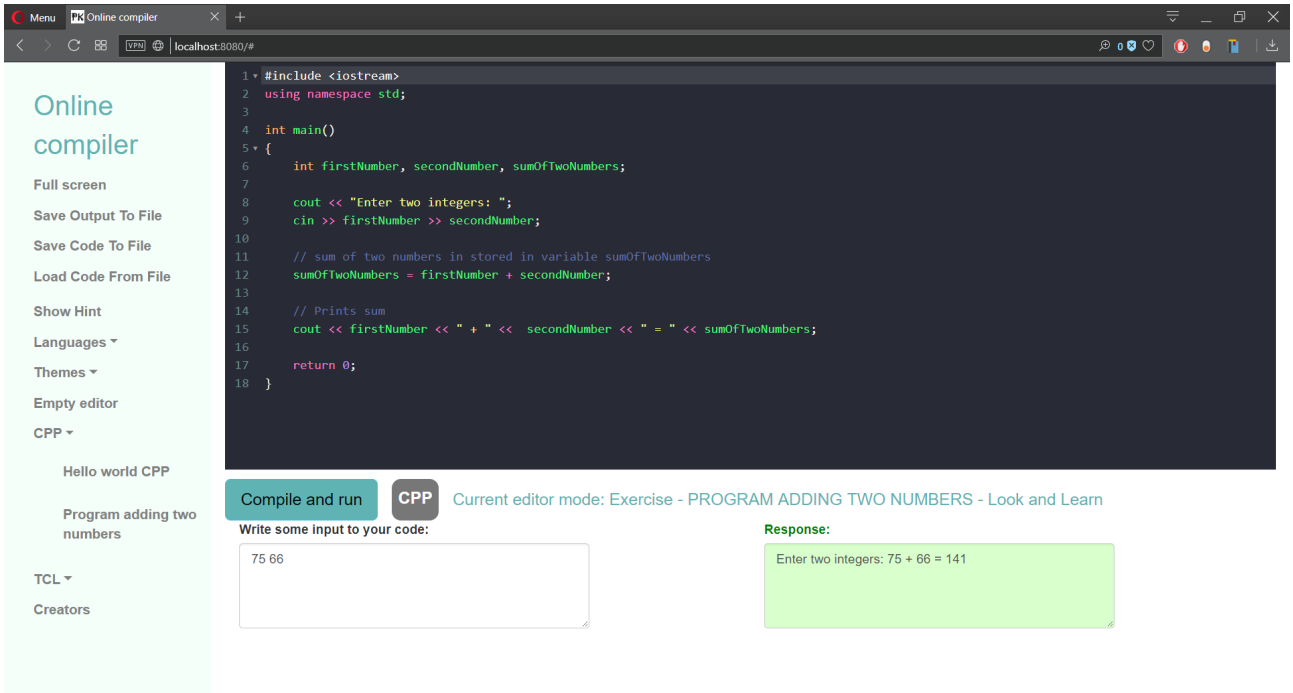
- Empty editor – Jeśli użytkownik nie chce wykonywać jednego z zadań i po prostu chce napisać swój program kliknięcie w ten przycisk spowoduje wyczyszczenie edytora i od tego czasu przy kompilacji nie będzie sprawdzane czy program został napisany poprawnie.
- CPP
- TCL
  Po kliknięciu w te pola pojawią się listy zadań które są możliwe do wykonania. Po wybraniu konkretnego zadania ładuje się ono do edytora, język zmieniany jest na właściwy, a przy kompilacji będzie następowało sprawdzenie czy wartość zwrócona jest taka, jaka być powinna.
- Creators – Pokazuje przez kogo została stworzona aplikacja.



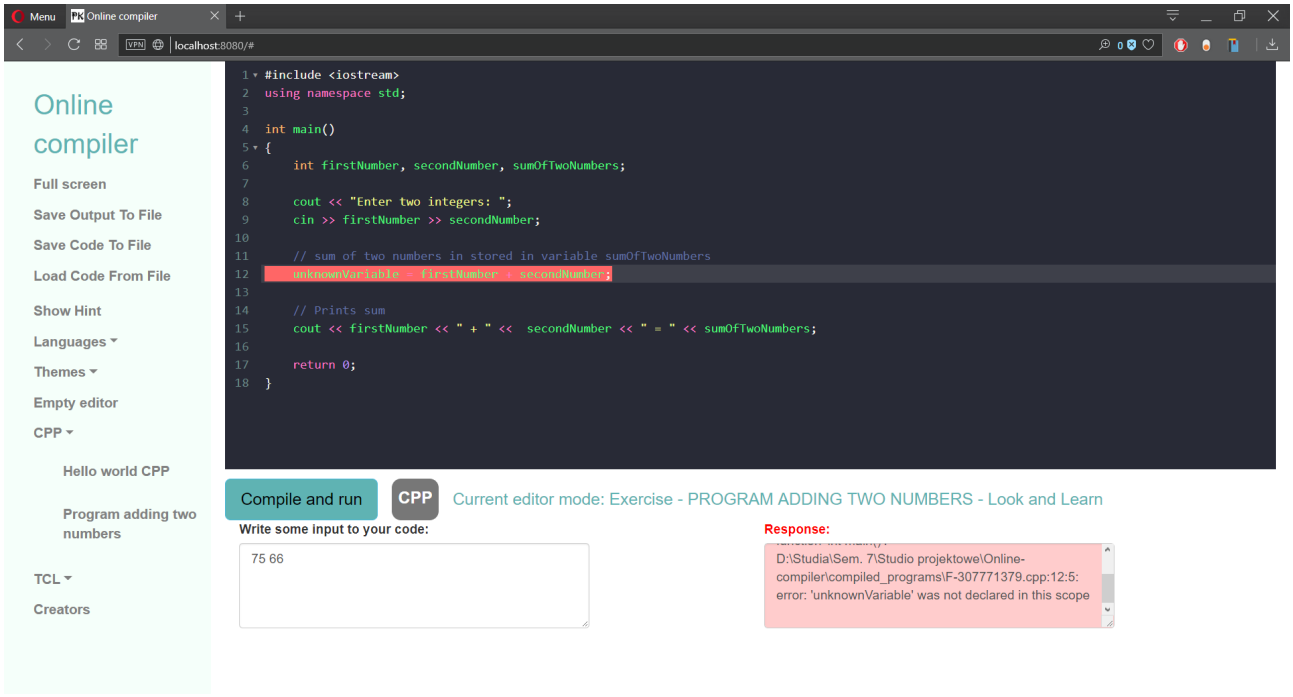
Oprócz tego pod edytorem znajduje się najważniejszy przycisk: Compile and run.
Po jego naciśnięciu napisany kod zostaje przesłany na serwer, gdzie w zależności od języka zostaje skompilowany w odpowiedni sposób, a później uruchomiony lub też po prostu uruchomiony jeśli nie wymaga procesu kompilacji (jak Tcl).
Po tym procesie zostaje wyświetlony rezultat programu lub też błąd jeśli nie udało się skompilować lub wystąpił on podczas uruchomienia.
Jeśli wszystko poszło dobrze pole Response podświetla się na zielono, jeśli nie to na czerwono.
Obok przycisku znajduje się pole informujące o aktualnie wybranym języku programowania, zadanie które jest wykonywane oraz opis tego zadania.

Pod nimi znajdują się pola odpowiednio wejścia i wyjścia z programu. Dzięki temu użytkownik może zapewnić własny input, a później odczytać output lub komunikat błędu.

Przykład poprawnego skompilowania:



Przykład błędu popełnionego w kodzie programu. W takiej sytuacji podświetlana jest linijka w której wystąpił błąd:



Oprócz tego jest możliwość wyłączenia możliwości edycji niektórych linijek. Przykładowo jeśli w zadaniu trzeba tylko uzupełnić ciało metody – wszystko co ją otacza może być nieaktywne – użytkownik nie może tych linii wtedy zmieniać. W tym przykładzie zaznaczone na żółto linie są nieaktywne.