**Uniwersytet Warmińsko-Mazurski w Olsztynie**

**Wydział Matematyki i Informatyki**

**Zespół TSR:**

**Damian Tomczak**

**Dominik Słodkowski**

**Bartosz Rybiński**

**Block-based programming language**

Praca zaliczeniowa napisana

napisana pod kierunkiem

mgr inż. Tomasza Żmijewskiego

Olsztyn 2022

1. **Opis firmy:**

TSR School jest franczyzodawcą o zasięgu ogólnokrajowym, który swoje placówki ma zlokalizowane w średnio zaludnionych polskich aglomeracjach powyżej 50 tyś mieszkańców. Poza tym nasza firma dostarcza narzędzia do nauczania zdalnego w języku angielskim.

Nasze przedsiębiorstwo jest akademią IT z prawie 10-letnim doświadczeniem proponującym nieszablonowane rozwiązania w zakładaniu innowacyjnego biznes w sektorze edukacyjnym. Korzystając z doświadczenia, „know-how”, materiałów edukacyjnych i platform do zarządzania.

Główna siedziba firmy zlokalizowana jest w Olsztynie na ulicy Słonecznej 54, 10-561 i otwarta jest w godzinach 8-16 (pomoc telefoniczna dla franczyzobiorców oraz uczniów jest otwarta w godzinach pracy placówki). Rekomendowane jest, aby studenci o pomoc kierowali się bezpośrednio do prowadzącego zajęcia. Preferowana metoda komunikacji ze szkołą odbywa się za pośrednictwem poczty elektronicznej pod adresem [contact@tsr.edu.pl](mailto:contact@tsr.edu.pl)

Dostarczamy studentom jak i prowadzącym zajęcia bezkonkurencyjne rozwiązania w dziedzinie rozwoju IT. Nasze sztandarowym narzędzie pokrywa zapotrzebowanie dla tych najmłodszych studentów. Narzędzie to jest aplikacją desktopową dla systemu linux, które naucza podopiecznych o programowaniu i umiejętności rozwiązywania problemów, natomiast prowadzącym zajęcia dostarcza możliwość monitorowania osiągnięć podopiecznych.

Studenci korzystający z stacjonarnego programu nauczania wyposażeni są po uruchomieniu sprzętu znajdującego się w placówce szkoły w zainstalowane oprogramowanie do nauki, oraz założone konto studenta przez prowadzącego zajęcia. To samo dotyczy się studentów korzystających z naszego zdalnego programu nauczania – w takim przypadku student powinien samodzielnie lub z pomocą rodzica wyposażyć się w oprogramowanie, które zbudowane powinno być gotowe do pobrania w serwisie github projektu znajdującego się pod adresem <https://github.com/damian-tomczak/tsr_15_00/releases>.

Po uruchomieniu programu studenta wyposażonego w swoje dane uwierzytelniana wita pulpit nawigacyjny programu (składającego się z pola login hasło oraz przycisku zaloguj), gdzie ma możliwość zalogowania się do platformy, w przypadku problemów z autoryzacją student winien skontaktować się z prowadzącym zajęcia. System dysponuje również możliwością logowania się prowadzącego zajęcia na tych samych zasadach co ucznia.

Po zalogowaniu użytkownikowi prezentowany jest widok składający się z rankingu, który prezentuje postęp nauki studenta wraz z porównaniem do innych studentów (ranking „po stronie” jest ograniczony do listy numerowanej z danymi podopiecznych). Widok pulpitu nawigacyjnego studenta po zalogowaniu składa się również z zadań do wykonania w postaci wertykalnej listy, której zawartość pobierana jest z bazy dany znajdującej się na serwerze, a do prowadzącego zajęcia należy obowiązek przypisywania zadań dla studenta z poziomu bazy danych albo jego wersji pulpitu nawigacyjnego po zalogowaniu. Prowadzący zajęcia podczas wyboru zadań ma możliwość sugerowania się rekomendowaną listą zadań dostarczanych przez główną siedziby firmy, skorzystać z puli potencjalnych zadań, bądź dodać zadanie samodzielnie. Prowadzący zajęcia po zalogowaniu dysponuje również możliwością dodania studenta na podstawie dostarczonych mu przez główną siedzibę danych.

Podczas działania programu, użytkownik ma możliwość dostosowania działania programu do własnych preferencji za pomocą ustawień dostępnych na pasku zadań.

Nauka z programem rozpoczyna się po wybraniu zadania a zadaniem ucznia jest osiągnięcie określonego celu przy pewnych zadanych warunkach początkowych. Interfejs użytkownika podczas realizowania zadania podzielony jest na trzy główne części:

1. Wertykalną lista bloków dostępnych dla studenta
2. Konsola informującą o wyniku pracy schematu
3. Główną część graficznego interfejsu użytkownika zajmuje tablica na której następuje tworzenie przebiegu pracy schematu

O sposobie działania programu decydują bloki, które realizują:

* podstawowe operacje warunkowe
* Funkcja koniunkcji
* Funkcja alternatywy
* Funkcja negacji
* podstawowe operacje arytmetyczne
  + operacja dodawania
  + operacja odejmowania
  + operacja mnożenia
  + operacja dzielenia
  + operacja reszty z dzielenia
* tworzenie zmiennych
* pętle warunkowe

Wynik działania bloków jest prezentowany studentowi w postaci tekstu na konsoli po dokonaniu wciśnięcia przycisku Startu.

**Słownik Pojęć**

Użytkownik = Osoba korzystająca z systemu zapisana w bazie danych, posiada login i hasło.

Student = Użytkownik zapisany w rankingu posiadający możliwość budowania schematów oraz wybierania i rozwiązywania przydzielonych zadań.

Prowadzący zajęcia = Użytkownik posiadający możliwość dodania studenta oraz przydzielania zadań dla studentów, mający dostęp do schematów studentów.

Blok = Podstawowa część schematu, mogąca realizować tworzenie i zmianę zmiennych, operacje warunkowe i arytmetyczne oraz pętle warunkowe, mogący wchodzić z interakcję z innymi blokami.

Schemat = Algorytm stworzony za pomocą bloków i relacji między nimi, którego wynik wypisywany jest na konsole.

Ranking = Lista prezentująca postęp nauki studenta z porównaniem do innych studentów, na podstawie ukończenia zadań dla nich przydzielonych.

Zadanie = Przydzielony przez prowadzącego zajęcia problem algorytmiczny, który jest rozwiązywany budując schemat.

Konsola = Standardowe wyjście tekstowe w dostępnym systemie, na którym wyświetlane są wyniki działania schematu.

* Organizacja – TSR/TSR School zwany również franczyzodawca zajmujący się organizacją usług w dziedzinie edukacji w branży IT
* Otoczenie organizacji – firmy rekrutujące specjalistów IT, inne szkoły programowania
* Intersariusze organizacji - dzieci i nastolatkowie, human reasources teams
* Dziedzina problemowa – nowoczesne technologie
* Intersariusze dziedziny problemowej - firmy technologiczne
* Pracownicy biznesowi – headquarter firmy, firmy rekrutujące specjalistów IT

Zakres organizacji, dziedziny problemowej, aktorzy

Firma TSR School

Organizacja = cała firma

Struktura firmy = prowadzący zajęcia, placówka dydaktyczna, główna siedziba firmy

Dziedzina problemowa : prowadzenie zajęć, nauka programowania i problem solving

Aktorzy biznesowi: student, nowy student

Pracownicy biznesowi: prowadzący zajęcia

PU:

- ~~Utwórz studenta~~ -> Administrowanie systemem

- ~~Przydziel zadanie~~ -> Administrowanie systemem

- ~~Autoryzuj użytkownika~~ -> Autoryzowanie

- ~~Wybierz zadanie~~ -> Nauka w programie

- ~~Buduj program~~ -> Nauka w programie

- ~~Kompiluj program~~ -> Nauka w programie

- ~~Oceń zadanie~~ -> Ranking

- ~~Dostosuj program~~ -> Nauka w programie

Aktorzy biznesowi: student, ~~nowy student~~

Pracownicy biznesowi: prowadzący zajęcia

PU:

- Administrowanie systemem

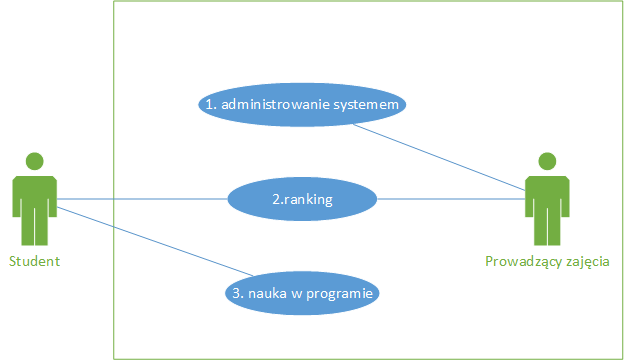
- Ranking

- Nauka w programie

Aktorzy biznesowi: student

Pracownicy biznesowi: prowadzący zajęcia

**Biznesowy Diagram Przypadków Użycia**



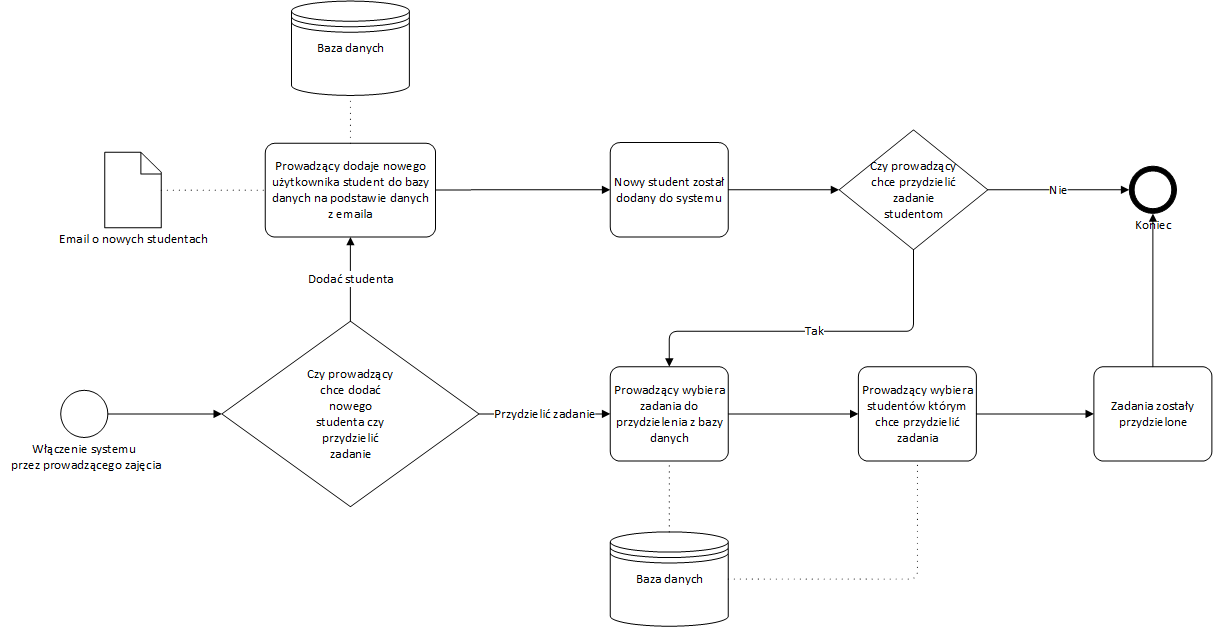
**Aktorzy**

* Student
* Prowadzący zajęcia

**Przypadki użycia**

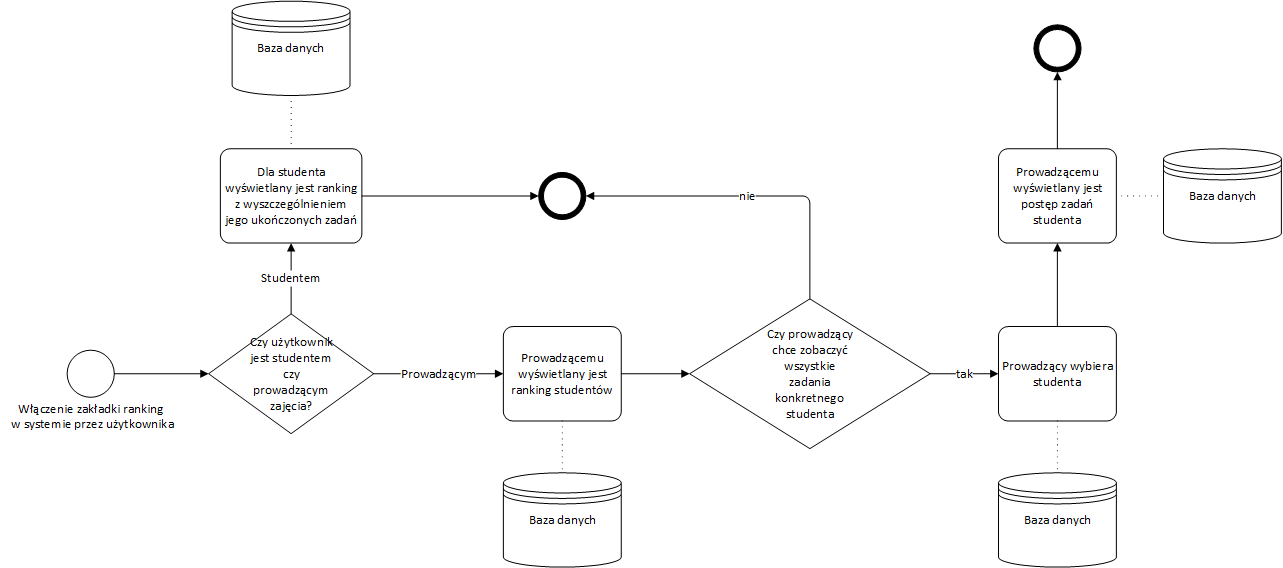
1. Administrowanie systemem

Administrowanie systemem lokalnie realizuje prowadzący zajęcia i do jego zadań należy tworzenie kont studentów, oraz przydzielanie zadań.



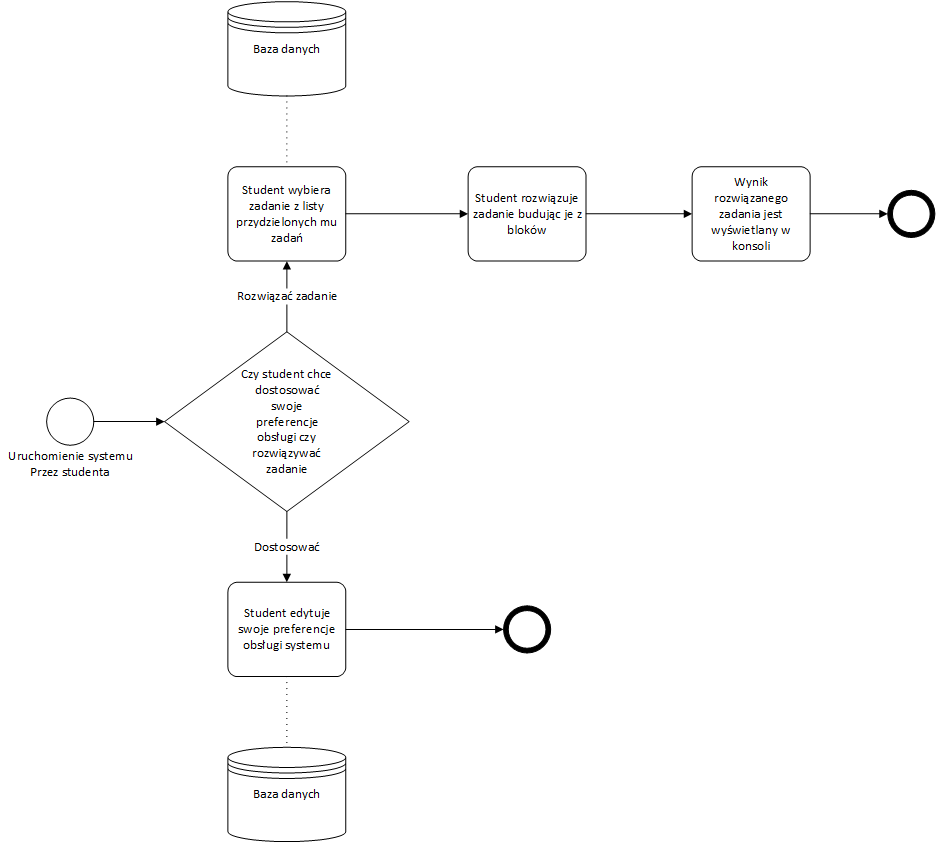
1. Ranking

Student ma możliwość monitorowania swoich osiągów oraz motywuje się dysponując możliwością porównaniu swoich wyników z rówieśnikami z grupy. Rola prowadzącego zajęcia w rankingu sprowadza się do analizy pracy jego studentów.



1. Nauka w programie

Student nabywa nowych umiejętności w programie, a jego obsługa sprowadza się do: dostosowaniu preferencji obsługi, wybieraniu przydzielanych zadań oraz tworzeniu skryptu za pośrednictwem bloków i sprawdzaniu i analizy wyników.



1)

Przykład:

Prowadzący zajęcia mgr inż. Jarosław Kaczyński otrzymał wiadomość e-mail zawierającą dane jego przyszłej grupy studentów składającej się z Brajana Kowalskiego, Mileny Samsel oraz Karola Tuska. Następnie Jarosław wymyśla dla studentów dane uwierzytelniania i wprowadza do systemu.

Jarosław Kaczyński wybiera studentów i dodaje dla nich zadanie dostępnego pod adresem https//www.gov.pl/tasks/task1.json

Specyfikacja:  
- prowadzący zajęcia dodaje studenta

- prowadzący zajęcia przydziela zadania dla studentów

2)

Przykład:

Prowadzący zajęcia po tygodniu analizuje wyniki pracy studentów za pomocą rankingu wygenerowane przez system i postanawia przedyskutować postęp prac studenta Karola Tuska.

Student Brajan Kowalski po zalogowaniu do systemu widzi wygenerowany przez system ranking i widzi, że w nim przoduje.

Specyfikacja:

- prowadzący zajęcia analizuje wyniki pracy studentów

- studenci porównują swoją pracę z innymi studentami

- system generuje ranking

3)

Przykład:

Student Milena Samsel po zalogowaniu, wybiera dostępne dla niej zadanie o nazwie „dzielenie z resztą”a następnie rozwija bazowy schemat składających się z dwóch bloków typu zmienna o wartościach 2 i 3 o odpowiednie bloki, aby osiągnąć wymieniony cel – „osiągnij resztę z dzielenia 5”. Przyciskiem Start student prosi system o wykonanie schematu, następnie system prezentuje na konsoli wynik działania schematu.

Specyfikacja:

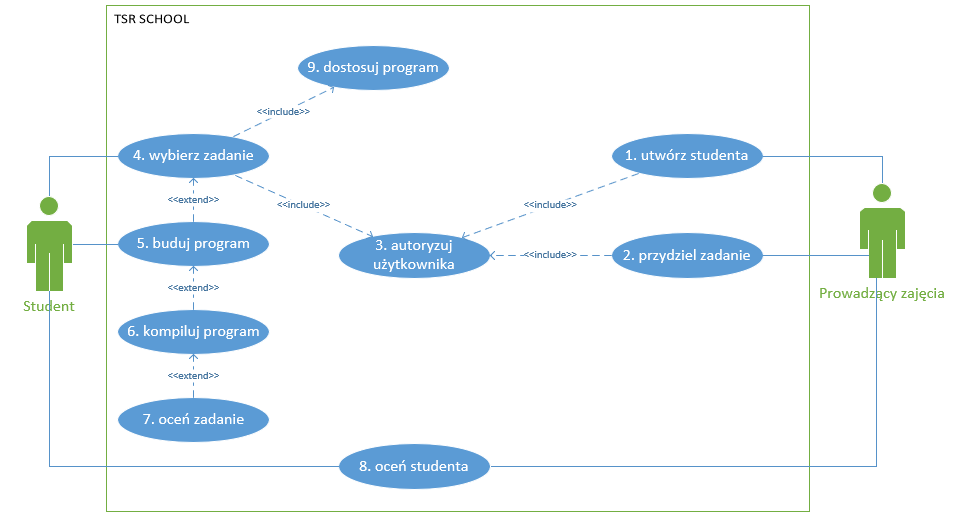
- student wybiera zadanie

- student rozwija schemat

- system wykonuje schemat

- system prezentuje wynik działania schematu

**Systemowy diagram przypadków użycia**



**Aktorzy**

* Student
* Prowadzący zajęcia

**Przypadki Użycia**

1. Utwórz studenta

Prowadzący zajęcia dostaje emaila z danymi studenta (lub listy studentów) który został przypisany do jego grupy. Na ich podstawie tworzy nowe konto dla studenta w systemie.

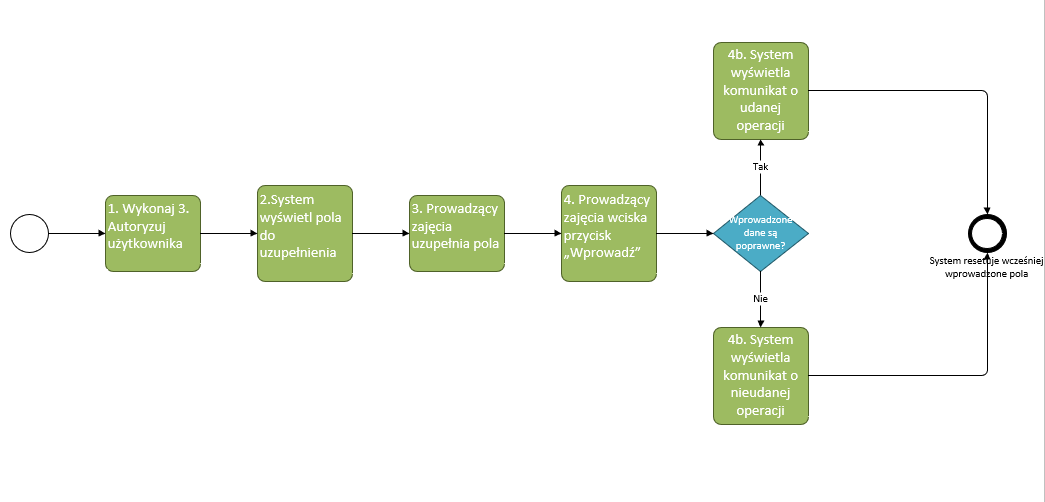
* + - 1. Wykonaj 3. Autoryzuj użytkownika
      2. System wyświetl pola do uzupełnienia
      3. Prowadzący zajęcia uzupełnia pola
      4. Prowadzący zajęcia wciska przycisk „Wprowadź”

4a. System wyświetla komunikat o udanej operacji

4b. System wyświetla komunikat o nieudanej operacji

* + - 1. System resetuje wcześniej wprowadzone pola

Prowadzący zajęcia mgr inż. Jarosław Kaczyński otrzymał wiadomość e-mail zawierającą dane jego przyszłej grupy studentów składającej się z Brajana Kowalskiego, Mileny Samsel oraz Karola Tuska. Prowadzący zajęcia loguje się do systemu, a następnie wymyśla i wprowadza dane studenta Brajana Kowalskiego (brajan.kowalski1 password123) w pola login i hasło, a następnie wciska przycisk „dodaj”. Czynność powtarza dla studenta Mileny Samsel (milena.samsel1 password1234)i Karola Tuska (karol.tusk1 password1235).



1. Przydziel Zadania

Prowadzący zajęcia tworzy nowe zadanie dla studentów w systemie z przykładowymi niekompletnymi rozwiązaniami, po czym umieszcza je w bazie danych i przypisuje wybrane z bazy danych zadania wybranym studentom.

* + - 1. Wykonaj 3. Autoryzuj użytkownika
      2. System wyświetla listę studentów i pole tekstowe

2a. System wyświetla komunikat o braku studentów

Zakończ

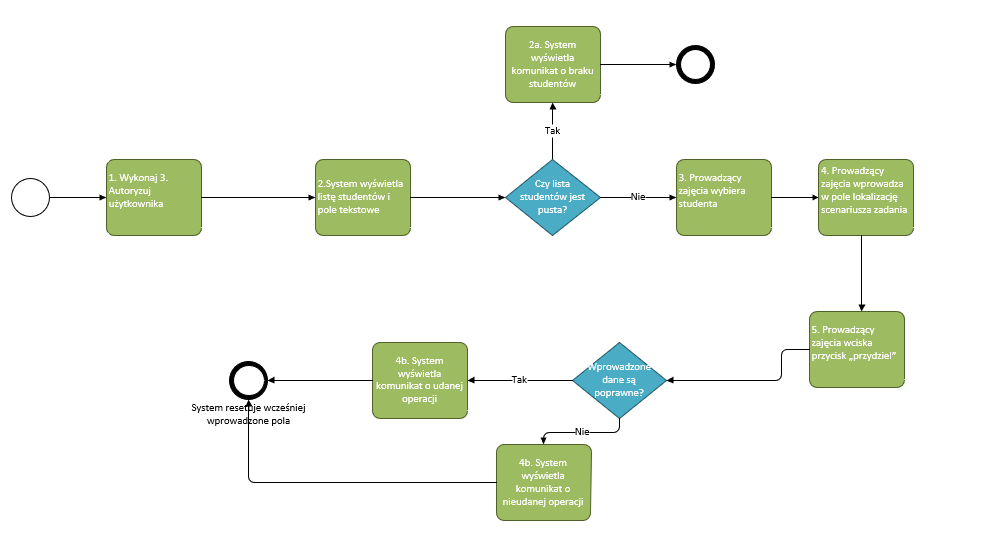
* + - 1. Prowadzący zajęcia wybiera studenta
      2. Prowadzący zajęcia wprowadza w pole lokalizację scenariusza zadania
      3. Prowadzący zajęcia wciska przycisk „przydziel”

5a. System wyświetla komunikat o udanej operacji

5b. System wyświetla komunikat o nieudanej operacji

* + - 1. System resetuje zaznaczenie i zawartość pola

Prowadzący zajęcia po zalogowaniu wybiera studenta Brajana Kowalskiego, Milenę Samsel i Karola Tuska z listy dostępnych studentów oraz wprowadza w pole „adres lokalizacji scenariusza” zadanie (https://www.gov.pl/tasks/task1.json) i wciska przycisk „przydziel”, następnie powiela wybranie studentów, ale w polu adresu lokalizacji scenariusza zadania wprowadza adres innego zadania (https://www.gov.pl/tasks/task2.json).



1. Autoryzuj użytkownika

Student/Prowadzący zajęcia wpisuje login I hasło w aplikacji. Jeżeli wprowadził prawidłowe dane przechodzi do przeglądu zadań. Jeśli wpisał złe informacje proszony jest o ponowne wpisanie hasła i loginu. Wprowadzone dane są później wykorzystywane w celu weryfikacji dostępu do danych.

1. Wybierz Zadanie

Student po zalogowaniu widzi ekran z możliwymi do wyboru zadaniami przypisanymi mu przez prowadzącego zajęcia. Wybiera konkretne zadanie i przechodzi do budowania programu.

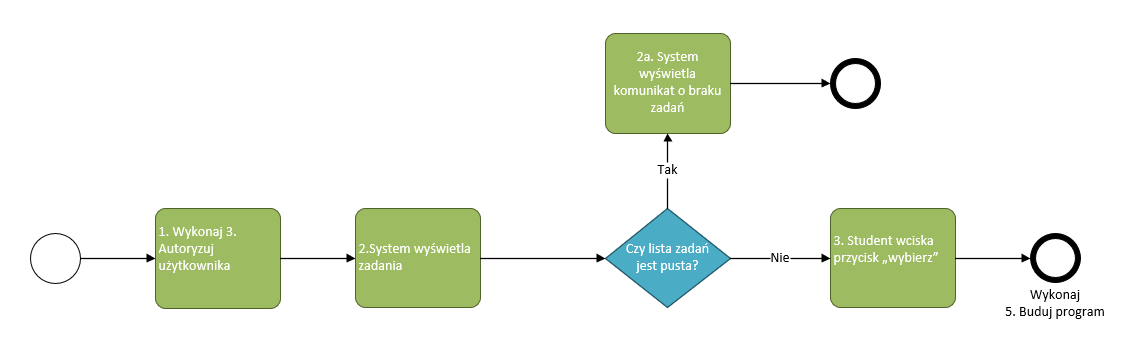
* + - 1. Wykonaj 3. Autoryzuj użytkownika
      2. System wyświetla listę zadań dostępnych dla studenta

2a. System wyświetla komunikat o braku dostępnych zadań

Zakończ

* + - 1. Student wciska przycisk „wybierz”
      2. Wykonaj 5. Buduj Program

Studentowi Karolowi Tuskowi po zalogowaniu system prezentuje wertykalną listę przypisanych do jego zadań w postaci identyfikator oraz nazwa (#1 Dzielenie z resztą, #2 Dodawanie, #3 Równanie kwadratowe). Karol zaznacza zadanie o identyfikatorze #2 i wybiera wciska przycisk „wybierz”.



1. Buduj Program

Student za pomocą bloków(zestaw instrukcji bądź instrukcja do wykonania przez procesor) które są układane przez niego w wybranej kolejności tworzy algorytm po czym może wcisnąć przycisk Start.

* + - 1. System wyświetla planszę budowania
      2. Student wyświetla listę bloków

2a Student postanawia opuścić zadanie

Wykonaj 4. Wybierz zadanie

* + - 1. Student wybiera blok z listy
      2. System umieszcza blok na planszy
      3. Student modyfikuje blok

4a. Student modyfikuje wartość bloku

4b. Student modyfikuje wyjście/a bloku

4c. Student modyfikuje wejście/a bloku

* + - 1. Student podejmuje decyzję

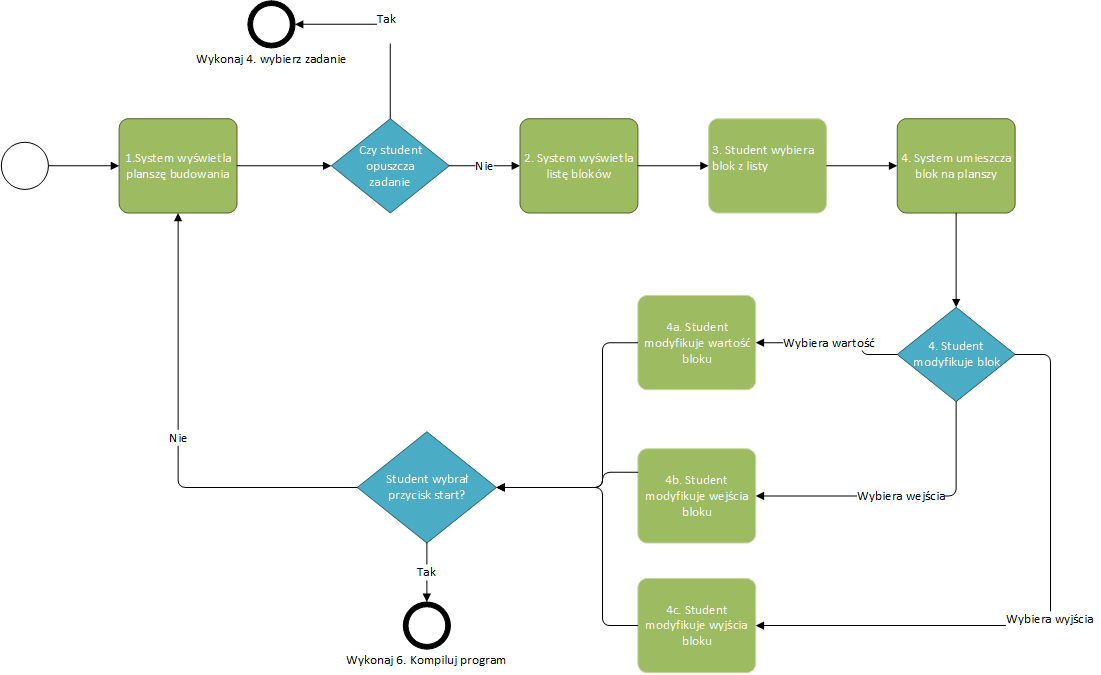
5a. Student dalej buduje program

Wróć do 5.1. System wyświetla planszę budowania

5b. Student wciska przycisk „start”

Wykonaj 6. Kompiluj program

Student Karol Tusk po zalogowaniu i wybraniu zadania „#2 Dodawanie” wyświetlana jest przez system plansza do budowania. Użytkownik zapoznaje się z istniejącymi blokami (blok typu zmienna int z wartością 1, blok typu zmienna int z wartością 2 oraz blok output) oraz celem działania programu „uzyskaj wynik na wyjściu 3”. karol.tusk1 następnie otwiera menu wyboru bloków i z kategorii działania arytmetyczne wybiera addition. Kolejną czynnością studenta jest połączenie wejść bloku addition do zmiennych typu int wartością 2 oraz 1, wyjście bloku addition łączy z blokiem output. Aby sprawdzić rezultat student wciska przycisk „start”.



1. Kompiluj program

Po wciśnięciu przycisku start program blokowy jest parse’owany do pythona i jest interpretowany przez interpreter. Jeżeli program jest napisany poprawnie jego wynik jest wyświetlany studentowi na konsoli, jeżeli nie zwracany jest błąd.

* + - 1. System parsuje schemat blokowy na kod pythona
      2. System wykonuje kod pythona

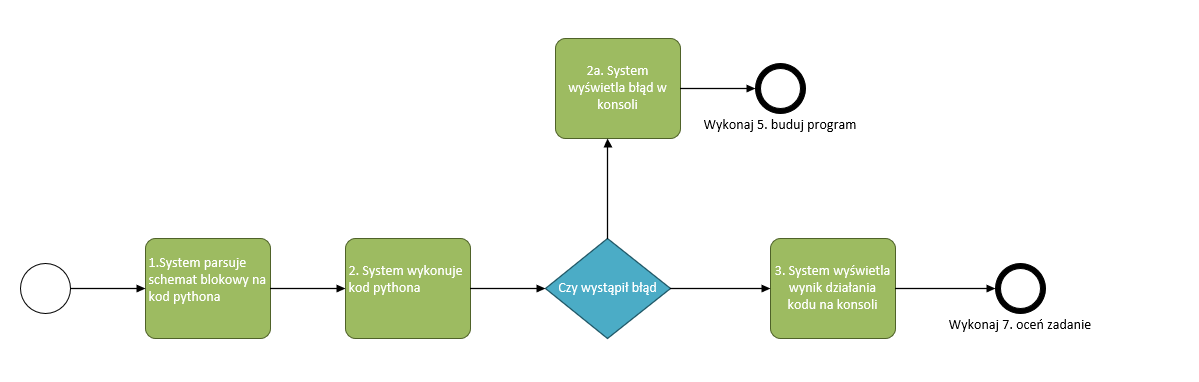
2a. System zwrócił błąd podczas interpretowania kodu

2a.1. System wyświetla błąd w konsoli

2a.2. Wykonaj 5. Buduj program

* + - 1. System wyświetla wynik działania kodu na konsoli
      2. Wykonaj 7. Oceń zadanie

System otrzymuje listę bloków w aplikacji wraz z ich stanem i relacjami miedzy nimi. Bloki to zmienna typu int z wartością 2, zmienna typu int z wartością 1, operacja arytmetyczna addition, blok output. Wejścia bloku addition zawierają relacje z blokami zmiennych z wartościami 1 oraz 2, natomiast wyjście zawiera relacje z outputem. System parsuje schemat na kod pythona. Wynik jego działania jest prawidłowy i wyświetla komunikat 3. Komunikat z wartością 3 jest prezentowany na konsoli użytkownika i przekazywany do oceny przez system.



1. Oceń zadanie

System po wykonaniu schematu porównuje rezultat z oczekiwanym rezultatem potem wyświetla stosowny komunikat w zależności od wyniku porównania, a uzyskanie prawidłowego zapisuje do bazy.

* + - 1. System odbiera wynik
      2. System sprawdza wynik

2a. Jeśli wynik prawidłowy

2a.1. System aktualizuje wykonanie zadanie

2a.2 System wyświetla komunikat o prawidłowym wykonaniu na konsoli

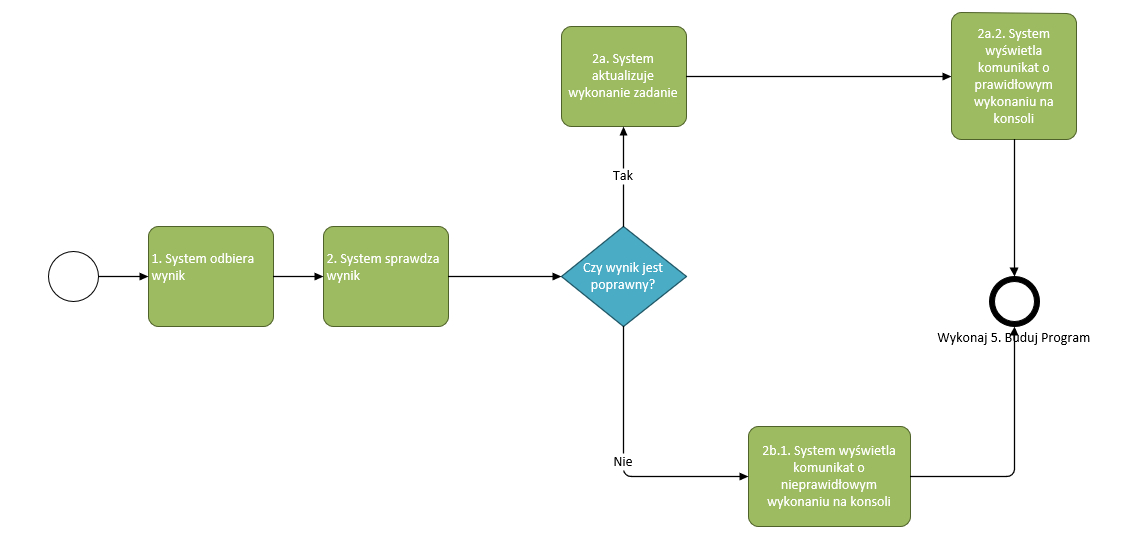
2a.3 Wykonaj 5. Buduj program

2b. Jeśli wynik nieprawidłowy

2b.1 System wyświetla komunikat o nieprawidłowym wykonaniu na konsoli

2b.2 Wykonaj 5. Buduj program

System odbiera wynik działania schematu studenta Karola Tuska w postaci stringu „3”, jego wartość jest porównywana z oczekiwaną wartością przypisaną do zadania „3”. Wynik to prawda, dlatego na konsoli studenta pojawia się komunikat o prawidłowym wykonaniu zadania, a wykonanie zadania rejestrowane jest w bazie danych.

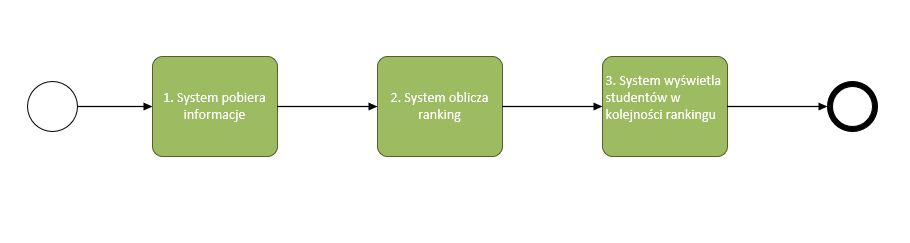


1. Oceń studenta

Prowadzona jest automatyczna lista studentów w oparciu o ilość wykonanych zadań do której mają dostęp studenci i prowadzący zajęcia.

* + - 1. System pobiera informacje
      2. System oblicza ranking
      3. System wyświetla studentów w kolejności rankingu

Student Karol zalogował się do systemu co stanowi wezwanie do wyświetlenia rankingu. System wyświetla listę trzech studentów z Brajana Kowalskiego, Mileny Samsel oraz Karola Tuska w kolejności ilości wykonanych zadań. 1. brajan.kowalski 2 zadania 2. milena.samsel 1 zadanie 3. karol.tusk 0 zadań.

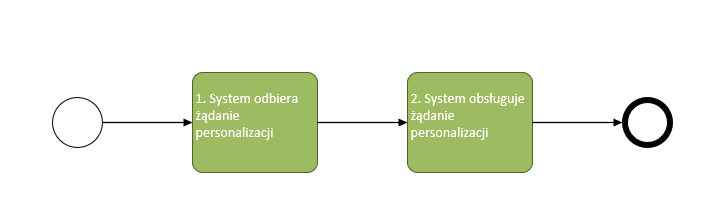


1. Dostosuj program

Użytkownik i prowadzący ma możliwość podstawowej możliwości dostosowania programu do własnych preferencji za pośrednictwem paska zadań (m.in. rozdzielczość).

* + - 1. System odbiera żądanie personalizacji
      2. System obsługuje żądanie personalizacji

Użytkownik Jarosław Kaczyński naciska przycisk „inne” na pasku zadań, system rozwija opcje ukryte spod tej opcji, które składają się z informacji o programie i pomocy. Następnie użytkownik naciska opcję „pomoc”. System prezentuje mu pomoc w obsłudze programu. Użytkownik zapoznaje się i zamyka okno.



**Dokumenty występujące w ramach dziedziny problemowej**

**Graphical user interface, text, application

Description automatically generatedformularz\_rejestracji** = rodzaj\_kursu + { dane\_uczestnika } + { dane\_opiekuna } + skad\_wiesz + uwagi

**rodzaj\_kursu** = [on-site | remote]

**dane\_uczestnika** = email + rok\_urodzenia + imie\_i\_nazwisko

**dane\_opiekuna** = email + nr\_telefonu + imie\_nazwisko + kod\_pocztowy

Graphical user interface, text, application

Description automatically generated

**dyplom** = imie\_nazwisko\_uczestnika + prowadzacy\_zajecia + data

Text

Description automatically generated

**zgoda\_przetwarzania\_danych\_osobowych** = data + podpis



**faktura\_za\_serwer** = data\_wystawienia + data\_wystawienia + nr\_faktury + nazwa\_sprzedawcy + adres\_sprzedawcy + nip\_sprzedawcy + vat\_ue + nabywca + adres\_nabywcy + płatnosc + zamowienie + pelna\_cena + uwagi

**platnosc** = [ karta | przelew | przelewy24 ]

**zamówienie** = { usluga }

**usluga** = nazwa\_serwera + ilosc + cena\_netto + vat + warto\_brutto