

Specyfikacja Wymagań Użytkownika

Kalkulator w Trybie Programisty

1. Wprowadzenie

1.1 Cel dokumentu

Niniejszy dokument określa wymagania funkcjonalne i нефункционаłne dla kalkulatora w trybie programisty. Opisuje sposób, w jaki system ma się zachowywać względem użytkownika podczas wykonywania operacji na liczbach w różnych systemach liczbowych.

1.2 Zakres aplikacji

Kalkulator programisty umożliwia wykonywanie operacji arytmetycznych, logicznych i bitowych na liczbach w czterech systemach liczbowych z możliwością podglądu wartości w różnych szerokościach słowa maszynowego.

1.3 Definicje i skróty

- **Hex** - system szesnastkowy (0-9, A-F)
- **Dec** - system dziesiętny (0-9)
- **Oct** - system ósemkowy (0-7)
- **Bin** - system binarny (0-1)
- **Qword** - 64 bity
- **Dword** - 32 bity
- **Word** - 16 bitów
- **Byte** - 8 bitów
- **GUI** - Graphical User Interface (graficzny interfejs użytkownika)

2. Aktorzy Systemu

2.1 Użytkownik (Programista)

Opis: Osoba korzystająca z kalkulatora do wykonywania operacji na liczbach w różnych systemach liczbowych.

Cele:

- Wykonywanie szybkich obliczeń w systemach binarnym, ósemkowym, dziesiętnym i szesnastkowym
- Przeliczanie wartości między systemami liczbowymi
- Wykonywanie operacji bitowych i logicznych
- Podgląd reprezentacji liczby w różnych szerokościach słowa

3. Wymagania Funkcjonalne

3.1 Systemy Liczbowe

3.1.1 Wybór systemu liczbowego

ID: WF-01

Priorytet: Wysoki

Wymaganie: Użytkownik musi mieć możliwość wyboru jednego z czterech systemów liczbowych:

- **Hex** (szesnastkowy) - cyfry 0-9 oraz A-F
- **Dec** (dziesiętny) - cyfry 0-9
- **Oct** (ósemkowy) - cyfry 0-7
- **Bin** (binarny) - cyfry 0-1

Zachowanie systemu:

- Po wybraniu systemu liczbowego, kalkulator automatycznie aktywuje tylko te klawisze, które są dopuszczalne w danym systemie
- W trybie Hex dostępne są wszystkie klawisze 0-9 oraz A-F
- W trybie Dec dostępne są tylko klawisze 0-9
- W trybie Oct dostępne są tylko klawisze 0-7
- W trybie Bin dostępne są tylko klawisze 0 i 1
- Niedostępne klawisze powinny być wizualnie oznaczone (np. wyszarzone)

3.1.2 Automatyczne przeliczanie

ID: WF-02

Priorytet: Wysoki

Wymaganie: System musi automatycznie przeliczać aktualnie wyświetlaną liczbę na wszystkie pozostałe systemy liczbowe.

Zachowanie systemu:

- Po wprowadzeniu lub zmianie liczby w jednym systemie, wartość jest natychmiast przeliczana i wyświetlana w pozostałych trzech systemach
- Wszystkie reprezentacje tej samej liczby są widoczne równocześnie
- Zmiana aktywnego systemu nie zmienia wartości liczby, tylko sposób jej wprowadzania

3.2 Podgląd Szerokości Słowa

ID: WF-03

Priorytet: Średni

Wymaganie: Użytkownik musi mieć możliwość podglądu aktualnej liczby w różnych szerokościach słowa maszynowego:

- **Qword** - 64 bity
- **Dword** - 32 bity
- **Word** - 16 bitów
- **Byte** - 8 bitów

Zachowanie systemu:

- W każdej sekcji szerokości słowa wyświetlana jest ta sama liczba obcięta do odpowiedniej liczby bitów
 - Obcięcie następuje od najmniej znaczących bitów
 - Wartości są aktualizowane na bieżąco podczas wprowadzania liczby lub wykonywania operacji
-

3.3 Operacje Arytmetyczne

3.3.1 Operacje binarne

ID: WF-04

Priorytet: Wysoki

Wymaganie: Użytkownik musi mieć możliwość wykonywania podstawowych operacji arytmetycznych:

- **Dodawanie (+)** - dodaje dwie liczby
- **Odejmowanie (-)** - odejmuje drugą liczbę od pierwszej
- **Mnożenie (×)** - mnoży dwie liczby
- **Dzielenie (÷)** - dzieli pierwszą liczbę przez drugą

Zachowanie systemu:

- Użytkownik wprowadza pierwszą liczbę, wybiera operację, wprowadza drugą liczbę i naciska "="
- Wynik jest wyświetlany natychmiast po naciśnięciu "="

3.3.2 Operacje unarne

ID: WF-05

Priorytet: Średni

Wymaganie: System musi obsługiwać operacje unarne:

- **Zmiana znaku (+/-)** - zmienia znak liczby
- **Pierwiastek (√)** - oblicza pierwiastek kwadratowy
- **Odwrotność (1/x)** - oblicza odwrotność liczby (1 podzielone przez liczbę)

Zachowanie systemu:

- Operacja jest wykonywana natychmiast po naciśnięciu odpowiedniego klawisza
 - Wynik zastępuje aktualnie wyświetlaną liczbę
-

3.4 Funkcje Kasowania

3.4.1 Wyczyszczenie wszystkiego (C)

ID: WF-06

Priorytet: Wysoki

Zachowanie systemu:

- Usuwa całą wprowadzoną liczbę
- Kasuje wszystkie zaplanowane operacje
- Resetuje kalkulator do stanu początkowego
- Nie czyści pamięci

3.4.2 Wyczyszczenie bieżącego wpisu (CE)

ID: WF-07

Priorytet: Wysoki

Zachowanie systemu:

- Usuwa tylko ostatni wprowadzony element
- Zachowuje poprzednie operacje i liczby
- Pozwala na ponowne wprowadzenie liczby bez utraty kontekstu obliczeń

3.4.3 Backspace (←)

ID: WF-08

Priorytet: Średni

Zachowanie systemu:

- Usuwa ostatnią wprowadzoną cyfrę
- Pozwala na korektę pomyłek podczas wprowadzania liczby
- Jeśli nie ma cyfr do usunięcia, wyświetla 0

3.5 Operacje Pamięci

ID: WF-09

Priorytet: Średni

Wymaganie: Użytkownik musi mieć możliwość przechowywania i manipulowania liczbą w pamięci kalkulatora:

- **MC (Memory Clear)** - czyści zawartość pamięci
- **MR (Memory Recall)** - wyświetla zawartość pamięci
- **MS (Memory Store)** - zapisuje bieżącą liczbę do pamięci (zastępuje poprzednią wartość)
- **M+ (Memory Add)** - dodaje bieżącą liczbę do wartości w pamięci
- **M- (Memory Subtract)** - odejmuje bieżącą liczbę od wartości w pamięci

Zachowanie systemu:

- Pamięć przechowuje jedną liczbę
- Wskaźnik pamięci (np. "M") jest widoczny, gdy pamięć zawiera niezerową wartość
- Operacje M+ i M- aktualizują wartość w pamięci bez wyświetlania jej na ekranie

3.6 Operacje Logiczne

3.6.1 Operacje binarne logiczne

ID: WF-10

Priorytet: Wysoki

Wymaganie: Użytkownik musi mieć możliwość wykonywania operacji logicznych na bitach liczby:

- **And** - logiczne „i” (wynikowy bit = 1 tylko gdy oba porównywane bity są równe 1)
- **Or** - logiczne „lub” (wynikowy bit = 1 gdy przynajmniej jeden z porównywanych bitów to 1)
- **Xor** - logiczne „różne” (wynikowy bit = 1 tylko gdy bity są różne: 0-1 lub 1-0)

Zachowanie systemu:

- Operacje działają analogicznie do operacji arytmetycznych
- Użytkownik wprowadza pierwszą liczbę, wybiera operację logiczną, wprowadza drugą liczbę i naciska "="
- Operacja jest wykonywana bit po bicie na reprezentacji binarnej liczb

3.6.2 Operacja unarna logiczna

ID: WF-11

Priorytet: Wysoki

Wymaganie: System musi obsługiwać negację bitową:

- **Not** - negacja bitowa (odwraca każdy bit liczby: $1 \rightarrow 0$, $0 \rightarrow 1$)

Zachowanie systemu:

- Operacja wykonywana jest natychmiast po naciśnięciu klawisza
- Negacja dotyczy wszystkich bitów w aktualnie wybranej szerokości słowa

3.7 Operacje Bitowe (Przesunięcia)

3.7.1 Przesunięcia logiczne

ID: WF-12

Priorytet: Średni

Wymaganie: Użytkownik musi mieć możliwość wykonywania operacji przesunięć bitowych:

- **Lsh (Left Shift)** - przesunięcie bitów w lewo
- **Rsh (Right Shift)** - przesunięcie bitów w prawo

Zachowanie systemu:

- Wszystkie bity przesuwają się w odpowiednim kierunku
- W Lsh: z prawej strony dopisywane są zera
- W Rsh: z lewej strony dopisywane są zera
- Użytkownik wprowadza liczbę, wybiera operację przesunięcia, wprowadza liczbę pozycji do przesunięcia i naciska "="

3.7.2 Rotacje

ID: WF-13

Priorytet: Średni

Wymaganie: System musi obsługiwać rotacje bitowe:

- **RoL (Rotate Left)** - rotacja bitów w lewo
- **RoR (Rotate Right)** - rotacja bitów w prawo

Zachowanie systemu:

- Bity przesuwają się w odpowiednim kierunku
 - W RoL: bit wypchnięty po lewej stronie wraca na koniec (najmniej znaczący bit)
 - W RoR: bit wypchnięty po prawej stronie wraca na początek (najbardziej znaczący bit)
 - Liczba bitów w liczbie pozostaje stała
-

3.8 Obsługa Nawiasów

ID: WF-14

Priorytet: Niski

Wymaganie: Użytkownik musi mieć możliwość stosowania nawiasów do budowania złożonych wyrażeń matematycznych.

Zachowanie systemu:

- Kalkulator obsługuje nawiasy otwierające "(" i zamykające ")"
 - Nawiasy pozwalają na określenie kolejności wykonywania operacji
 - System oblicza wyrażenie zgodnie z priorytetem operatorów i nawiasów
 - Niezamknięte nawiasy powinny być sygnalizowane użytkownikowi
 - Liczba otwartych nawiasów może być wyświetlana na ekranie
-

4. Przypadki Użycia (Use Cases)

UC-01: Przeliczanie liczby między systemami liczbowymi

Aktor: Użytkownik

Warunek początkowy: Kalkulator jest uruchomiony i wyświetla wartość 0

Główny scenariusz:

1. Użytkownik wybiera system liczbowy Dec
2. Użytkownik wprowadza liczbę "255"
3. System wyświetla liczbę w systemie Dec: "255"
4. System automatycznie przelicza i wyświetla wartość w systemie Hex: "FF"
5. System automatycznie przelicza i wyświetla wartość w systemie Oct: "377"
6. System automatycznie przelicza i wyświetla wartość w systemie Bin: "11111111"
7. Użytkownik zmienia system na Hex

8. System umożliwia wprowadzanie cyfr 0-9 i A-F

Warunek końcowy: Liczba jest wyświetlona we wszystkich czterech systemach liczbowych

Rozszerzenia:

- 2a. Użytkownik wprowadza niedozwoloną cyfrę dla danego systemu
 - 2a1. System ignoruje naciśnięcie niedostępnego klawisza
 - 2a2. Powrót do kroku 2
-

UC-02: Wykonywanie podstawowej operacji arytmetycznej

Aktor: Użytkownik

Warunek początkowy: Kalkulator jest uruchomiony

Główny scenariusz:

1. Użytkownik wprowadza pierwszą liczbę "10" w systemie Dec
2. System wyświetla "10" na ekranie głównym
3. Użytkownik naciska przycisk operacji "+"
4. System zapamiętuje liczbę i operację
5. Użytkownik wprowadza drugą liczbę "5"
6. System wyświetla "5" na ekranie głównym
7. Użytkownik naciska przycisk "="
8. System oblicza wynik: $10 + 5 = 15$
9. System wyświetla wynik "15" we wszystkich systemach liczbowych

Warunek końcowy: Wynik operacji jest wyświetlony na ekranie

Rozszerzenia:

- 7a. Użytkownik wykonuje dzielenie przez zero
 - 7a1. System wyświetla komunikat o błędzie "Nie można dzielić przez zero"
 - 7a2. Kalkulator pozostaje w stanie przed operacją
-

UC-03: Wykonywanie operacji logicznej AND

Aktor: Użytkownik

Warunek początkowy: Kalkulator jest uruchomiony w trybie Bin

Główny scenariusz:

1. Użytkownik wprowadza pierwszą liczbę "1010" (10 w Dec)
2. System wyświetla liczbę we wszystkich formatach
3. Użytkownik naciska przycisk operacji "And"
4. System zapamiętuje liczbę i operację
5. Użytkownik wprowadza drugą liczbę "1100" (12 w Dec)
6. System wyświetla drugą liczbę
7. Użytkownik naciska przycisk "="

8. System wykonuje operację AND bit po bicie: $1010 \text{ AND } 1100 = 1000$

9. System wyświetla wynik "1000" (8 w Dec) we wszystkich systemach

Warunek końcowy: Wynik operacji logicznej jest wyświetlony

UC-04: Przesunięcie bitowe w lewo (Left Shift)

Aktor: Użytkownik

Warunek początkowy: Kalkulator wyświetla liczbę "5" (Bin: 101)

Główny scenariusz:

1. Użytkownik ma liczbę "5" na ekranie (Bin: 101)
2. Użytkownik naciska przycisk "Lsh"
3. System zapamiętuje liczbę i operację
4. Użytkownik wprowadza liczbę pozycji przesunięcia "2"
5. System wyświetla "2"
6. Użytkownik naciska "="
7. System przesuwa bity w lewo o 2 pozycje: $101 \rightarrow 10100$
8. System wyświetla wynik "20" (Bin: 10100) we wszystkich formatach

Warunek końcowy: Liczba jest przesunięta bitowo w lewo

UC-05: Używanie pamięci kalkulatora

Aktor: Użytkownik

Warunek początkowy: Kalkulator jest uruchomiony, pamięć jest pusta

Główny scenariusz:

1. Użytkownik wprowadza liczbę "100"
2. System wyświetla "100"
3. Użytkownik naciska "MS" (Memory Store)
4. System zapisuje "100" do pamięci
5. System wyświetla wskaźnik "M" sygnalizujący wypełnioną pamięć
6. Użytkownik wprowadza nową liczbę "50"
7. Użytkownik naciska "M+" (Memory Add)
8. System dodaje 50 do wartości w pamięci ($100 + 50 = 150$)
9. Użytkownik naciska "MR" (Memory Recall)
10. System wyświetla wartość z pamięci "150"

Warunek końcowy: Wartość z pamięci jest wyświetlona na ekranie

Rozszerzenia:

- 9a. Użytkownik naciska "MC" (Memory Clear)
 - 9a1. System czyści pamięć
 - 9a2. Wskaźnik "M" znika
-

UC-06: Podgląd liczby w różnych szerokościach słowa

Aktor: Użytkownik

Warunek początkowy: Użytkownik wprowadził liczbę "300" w systemie Dec

Główny scenariusz:

- [illegible]

Warunek końcowy: Wszystkie reprezentacje szerokości słowa są widoczne

Uwagi: W reprezentacji Byte liczba 300 ($256 + 44$) zostaje obcięta do 44, ponieważ 8 bitów może pomieścić tylko wartości 0-255.

UC-07: Budowanie złożonego wyrażenia z nawiasami

Aktor: Użytkownik

Warunek początkowy: Kalkulator jest uruchomiony

Główny scenariusz:

1. Użytkownik naciska "("
2. System wyświetla wskaźnik otwartego nawiasu, np. "(1"
3. Użytkownik wprowadza "5"
4. Użytkownik naciska "+"
5. Użytkownik wprowadza "3"
6. Użytkownik naciska ")"
7. System zmniejsza licznik nawiasów
8. Użytkownik naciska "x"
9. Użytkownik wprowadza "2"
10. Użytkownik naciska "="
11. System oblicza: $(5 + 3) \times 2 = 16$
12. System wyświetla wynik "16"

Warunek końcowy: Złożone wyrażenie jest obliczone zgodnie z priorytetem operacji

Rozszerzenia:

- 10a. Użytkownik naciska "=" bez zamknięcia wszystkich nawiasów
 - 10a1. System wyświetla komunikat "Niezamknięte nawiasy"
 - 10a2. Obliczenie nie jest wykonywane

UC-08: Kasowanie wprowadzonych danych

Aktor: Użytkownik

Warunek początkowy: Użytkownik rozpoczął wprowadzanie wyrażenia "12 + 34"

Główny scenariusz (CE):

1. Użytkownik wprowadził "12"
2. Użytkownik nacisnął "+"
3. Użytkownik wprowadza "34"
4. Użytkownik naciska "CE" (Clear Entry)
5. System usuwa tylko "34"
6. Operacja "12 +" pozostaje w pamięci kalkulatora
7. Użytkownik może wprowadzić nową liczbę

Warunek końcowy: Tylko ostatni wpis jest usunięty

Alternatywny scenariusz (C):

1. Użytkownik wprowadził "12"
2. Użytkownik nacisnął "+"
3. Użytkownik wprowadza "34"
4. Użytkownik naciska "C" (Clear)
5. System usuwa całe wyrażenie "12 + 34"
6. Kalkulator wraca do stanu początkowego (wyświetla 0)

Alternatywny scenariusz (Backspace):

1. Użytkownik wprowadza "1234"
2. Użytkownik naciska "←" (Backspace)
3. System usuwa ostatnią cyfrę "4"
4. Na ekranie pozostaje "123"
5. Użytkownik może kontynuować wprowadzanie cyfr

UC-09: Negacja bitowa (NOT)

Aktor: Użytkownik

Warunek początkowy: Kalkulator wyświetla liczbę "5" w trybie Byte (8 bitów)

Główny scenariusz:

1. Użytkownik ma liczbę "5" na ekranie (Bin: 00000101)
2. System wyświetla w formacie Byte: 00000101
3. Użytkownik naciska "Not"
4. System wykonuje negację bitową: 00000101 → 11111010
5. System wyświetla wynik "250" (Bin: 11111010)
6. Wynik jest natychmiast widoczny we wszystkich systemach liczbowych

Warunek końcowy: Wszystkie bity liczby zostały zanegowane

Uwagi: Wynik negacji zależy od wybranej szerokości słowa (Byte, Word, Dword, Qword)

5. Wymagania Niefunkcjonalne

5.1 Użyteczność (Usability)

ID: WNF-01

Priorytet: Wysoki

- Interfejs musi być intuicyjny i zbliżony do standardowych kalkulatorów programisty (np. Windows Calculator w trybie Programmer)
- Klawisze niedostępne w danym systemie liczbowym muszą być wyraźnie oznaczone (wyszarzone lub dezaktywowane)
- Aktualna wartość musi być zawsze widoczna we wszystkich systemach liczbowych równocześnie
- Rozmiar przycisków musi być wystarczający dla wygodnego klikania
- Kontrast kolorów musi zapewniać czytelność

5.2 Responsywność (Performance)

ID: WNF-02

Priorytet: Wysoki

- Wszystkie przeliczenia między systemami liczbowymi muszą odbywać się natychmiast ($< 100\text{ms}$)
- Operacje arytmetyczne i logiczne muszą być wykonywane bez zauważalnego opóźnienia
- Aktualizacja podglądu szerokości słowa musi być natychmiastowa
- Interfejs użytkownika nie może "zawieszać się" podczas obliczeń

5.3 Precyzja (Reliability)

ID: WNF-03

Priorytet: Wysoki

- Kalkulator musi obsługiwać liczby całkowite w zakresie wybranej szerokości słowa (do 64 bitów)
- Operacje arytmetyczne muszą być dokładne w ramach ograniczeń szerokości słowa
- Przepięnienie zakresu (overflow) powinno być obsługiwane zgodnie ze standardami arytmetyki liczb całkowitych
- Wszystkie przeliczenia między systemami muszą być matematycznie poprawne

5.4 Obsługa Błędów

ID: WNF-04

Priorytet: Średni

- Dzielenie przez zero musi być sygnalizowane użytkownikowi komunikatem
- Nieprawidłowe operacje (np. pierwiastek z liczby ujemnej) muszą być odpowiednio obsługiwane
- Komunikaty o błędach muszą być zrozumiałe dla użytkownika (w języku polskim)
- Po wystąpieniu błędu użytkownik powinien móc kontynuować pracę

5.5 Kompatybilność

ID: WNF-05

Priorytet: Średni

- Aplikacja powinna działać na systemach Windows, Linux i macOS
 - Interfejs powinien być responsywny i działać na różnych rozdzielczościach ekranu
 - Aplikacja powinna wspierać obsługę klawiatury (skrótów klawiszowe)
-

6. Harmonogram Implementacji

Faza 1: Podstawowa funkcjonalność (Tydzień 1-2)

1. Interfejs użytkownika - układ kalkulatora
2. Wprowadzanie liczb w systemie dziesiętnym (Dec)
3. Wyświetlanie liczby na ekranie głównym
4. Podstawowe operacje arytmetyczne: dodawanie, odejmowanie, mnożenie, dzielenie
5. Funkcje kasowania: C, CE, Backspace

Kryteria akceptacji Fazy 1:

- Użytkownik może wprowadzać liczby dziesiętne
 - Podstawowe operacje arytmetyczne działają poprawnie
 - Funkcje kasowania działają zgodnie z oczekiwaniami
-

Faza 2: Systemy liczbowe (Tydzień 3-4)

6. Implementacja systemu binarnego (Bin)
7. Implementacja systemu ósemkowego (Oct)
8. Implementacja systemu szesnastkowego (Hex)
9. Automatyczne przeliczanie między systemami liczbowymi
10. Aktywacja/dezaktywacja klawiszy w zależności od wybranego systemu

Kryteria akceptacji Fazy 2:

- Wszystkie cztery systemy liczbowe są funkcjonalne
 - Przeliczanie między systemami działa automatycznie i poprawnie
 - Klawisze są odpowiednio aktywowane/dezaktywowane
-

Faza 3: Szerokość słowa (Tydzień 5)

11. Wyświetlanie wartości w formacie Byte (8 bitów)
12. Wyświetlanie wartości w formacie Word (16 bitów)
13. Wyświetlanie wartości w formacie Dword (32 bity)
14. Wyświetlanie wartości w formacie Qword (64 bity)
15. Automatyczna aktualizacja wszystkich formatów

Kryteria akceptacji Fazy 3:

- Wszystkie cztery szerokości słowa są wyświetlane równocześnie
 - Obcięcie wartości działa poprawnie
 - Aktualizacja jest natychmiastowa
-

Faza 4: Operacje zaawansowane (Tydzień 6)

- 16. Operacje unarne: zmiana znaku (+/-), pierwiastek ($\sqrt{}$), odwrotność ($1/x$)
- 17. Obsługa nawiasów
- 18. Obsługa priorytetów operacji

Kryteria akceptacji Fazy 4:

- Operacje unarne działają poprawnie
 - Nawiasy są prawidłowo obsługiwane
 - Priorytety operacji są zgodne ze standardami matematycznymi
-

Faza 5: Operacje pamięci (Tydzień 7)

- 19. Implementacja pamięci kalkulatora
- 20. Funkcje: MC, MR, MS
- 21. Funkcje: M+, M-
- 22. Wskaźnik stanu pamięci

Kryteria akceptacji Fazy 5:

- Wszystkie operacje pamięci działają poprawnie
 - Wskaźnik pamięci jest widoczny gdy pamięć jest wypełniona
-

Faza 6: Operacje logiczne (Tydzień 8)

- 23. Operacje binarne logiczne: And, Or, Xor
- 24. Operacja unarna: Not

Kryteria akceptacji Fazy 6:

- Operacje logiczne działają bit po bicie
 - Negacja bitowa uwzględnia wybraną szerokość słowa
-

Faza 7: Operacje bitowe (Tydzień 9)

- 25. Przesunięcia logiczne: Lsh, Rsh
- 26. Rotacje: RoL, RoR

Kryteria akceptacji Fazy 7:

- Przesunięcia bitowe działają poprawnie
 - Rotacje zachowują wszystkie bity
-

Faza 8: Testy i optymalizacja (Tydzień 10-11)

- 27. Testy jednostkowe wszystkich operacji
- 28. Testy integracyjne złożonych wyrażeń

- 29. Obsługa błędów i walidacja danych wejściowych
- 30. Optymalizacja wydajności i responsywności interfejsu
- 31. Dokumentacja użytkownika

Kryteria akceptacji Fazy 8:

- Wszystkie testy przechodzą pomyślnie
 - Obsługa błędów działa zgodnie z wymaganiami
 - Dokumentacja jest kompletna
-