**Sprawozdanie do projektu:**

Kalkulator

**Autorzy projektu:**

Tymoteusz Urbaniak

Kamil Jadczuk vel Bartoszuk

Dariusz Korszun

**Informatyka VI semestr**

**Koszalin 2015**

1. **Protokół założycielski**

W dniu 13 marca 2015 roku, w Koszalinie zebrała się grupa studentów uczelni wyższej Politechniki Koszalińskiej w składzie:

W czasie trwania zebrania w/w grupa ustaliła co następuje:

# Postanowienia ogólne

§1

1. Grupa powołana została w celu stworzenia projektu programistycznego wraz z dokumentacją w ramach przedmiotu „Zespołowy projekt ”.

2. Grupa przyjmuje nazwę Zespół projektowy i jest ona nazwą reprezentacyjną grupy.

3. Kierownikiem projektu zostanie wybrana jedna osoba z grona Zespołu projektowego, a wynik wyboru kierownika opisuje załącznik nr 1.

§2

1. Siedzibą grupy jest Wydział Elektroniki i Informatyki Politechniki Koszalińskiej, zwanej dalej PK, mieszczący się w Koszalinie przy ulicy Śniadeckich 2.

§3

1. Grupa podejmuje się zaprojektowania i realizacją projektu według założeń i tematu przedstawionych w załączniku nr 2.

2. Terminy otwarcia, zamknięcia oraz terminy poszczególnych etapów realizacji projektu zawarte będą w załączniku nr 3.

3. O podziale obowiązków dotyczących realizacji projektu opisanego załącznikiem nr 1 decyduje kierownik projektu i zawarte one będą w załączniku nr 4.

# Prawa, obowiązki i kary członków Zespołu projektowego

§4

1. Członkowie są zobowiązani do przestrzegania zasad i reguł ustalonych przez Zespół projektowy.

2. Każdy członek Zespołu projektowego zobowiązuje się do działania na korzyść grupy, wykonywania prac zmierzających do osiągnięcia założonego celu oraz współpracy z pozostałymi członkami grupy.

3. Każdy członek Zespołu projektowego ma prawo do wolnego zgłaszania pomysłów i projektów w celu poprawy efektywności pracy i funkcjonowania grupy.

§5

1. Uchylanie się członka Zespołu projektowego od obowiązków lub działanie na niekorzyść Zespołu projektowego skutkuje poddaniem pod głosowanie przez członków nad jego ukaraniem w postaci upomnienia.

2. Trzykrotne ustne upomnienie skutkuje propozycją obniżenia oceny o 0,5 punktu.

# Prawa, obowiązki i kara kierownika Zespołu projektowego

§6

1. Kierownik grupy upoważniony jest do reprezentowania interesów Zespołu projektowego wobec partnerów zewnętrznych, władz uczelnianych oraz wykładowców PK.

2. Umowy zawarte przez kierownika grupy z partnerami zewnętrznymi, władzami uczelnianymi lub wykładowcami PK są dla grupy wiążące.

3. Kierownik grupy zobowiązany jest do działania na korzyść Zespołu projektowego.

4. Kierownik grupy zobowiązany jest do realizowania uchwał i postanowień ustanowionych przez Zespół projektowy.

5. Każda decyzja wydana w formie załącznika musi być podpisana przez kierownika grupy.

§7

1. Jakiekolwiek działanie na niekorzyść Zespołu projektowego lub rażące naruszenie zasad Zespołu projektowego może być powodem odwołania kierownika grupy.

# Postanowienia końcowe

§6

Protokół założycielski sporządzono w czterech jednobrzmiących egzemplarzach potwierdzonych podpisami wszystkich członków Zespołu projektowego.

|  |  |  |
| --- | --- | --- |
| Tymoteusz Urbaniak | Kamil Jadczuk vel Bartoszuk | Dariusz Korszun |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |

Załącznik nr 2

Na temat realizowanego projektu zgłoszono propozycje:

|  |  |  |
| --- | --- | --- |
| Tymoteusz Urbaniak | Kamil Jadczuk vel Bartoszuk | Dariusz Korszun |

Załącznik nr 3

Termin otwarcia projektu „….” rozpoczyna się w dniu 13 marzec 2015 roku, a termin zakończenia upływa w dniu ………… 2015 roku.

Terminy poszczególnych etapów realizacji będą zawarte w aneksach do tego załącznika, numerowanych kolejno począwszy od numeru 1 i decyzje o ich ustanowieniu pozostawiamy Kierownikowi projektu.

|  |  |  |
| --- | --- | --- |
| Tymoteusz Urbaniak | Kamil Jadczuk vel Bartoszuk | Dariusz Korszun |

Załącznik nr 4

Ustalono podział obowiązków związanych z realizacją projektu:

- Tymoteusz Urbaniak

- Kamil Jadczuk vel Bartoszuk

- Dariusz Korszun

|  |  |  |
| --- | --- | --- |
| Tymoteusz Urbaniak | Kamil Jadczuk vel Bartoszuk | Dariusz Korszun |

1. **Cel projektu**

Celem Naszym jest stworzenie aplikacji kalkulatora z możliwością dodawania nowych funkcji (przycisków).

**Funkcjonalne**

* Intuicyjny interfejs
* Przejrzystość
* Duża dokładność
* Możliwość zapamiętywania wyniku
* Możliwość wykonywania działań arytmetycznych
* Możliwość wykonywania obliczeń budowlanych:
* Obliczanie ilości farby, gładzi, kafelek, paneli, drewna potrzebnych do remontu
* Kasowanie danych
* Funkcje trygonometryczne
* Duże przyciski
* Przejrzysty wyświetlacz
* Szybkie uruchamianie
* Zmiana funkcji przycisków

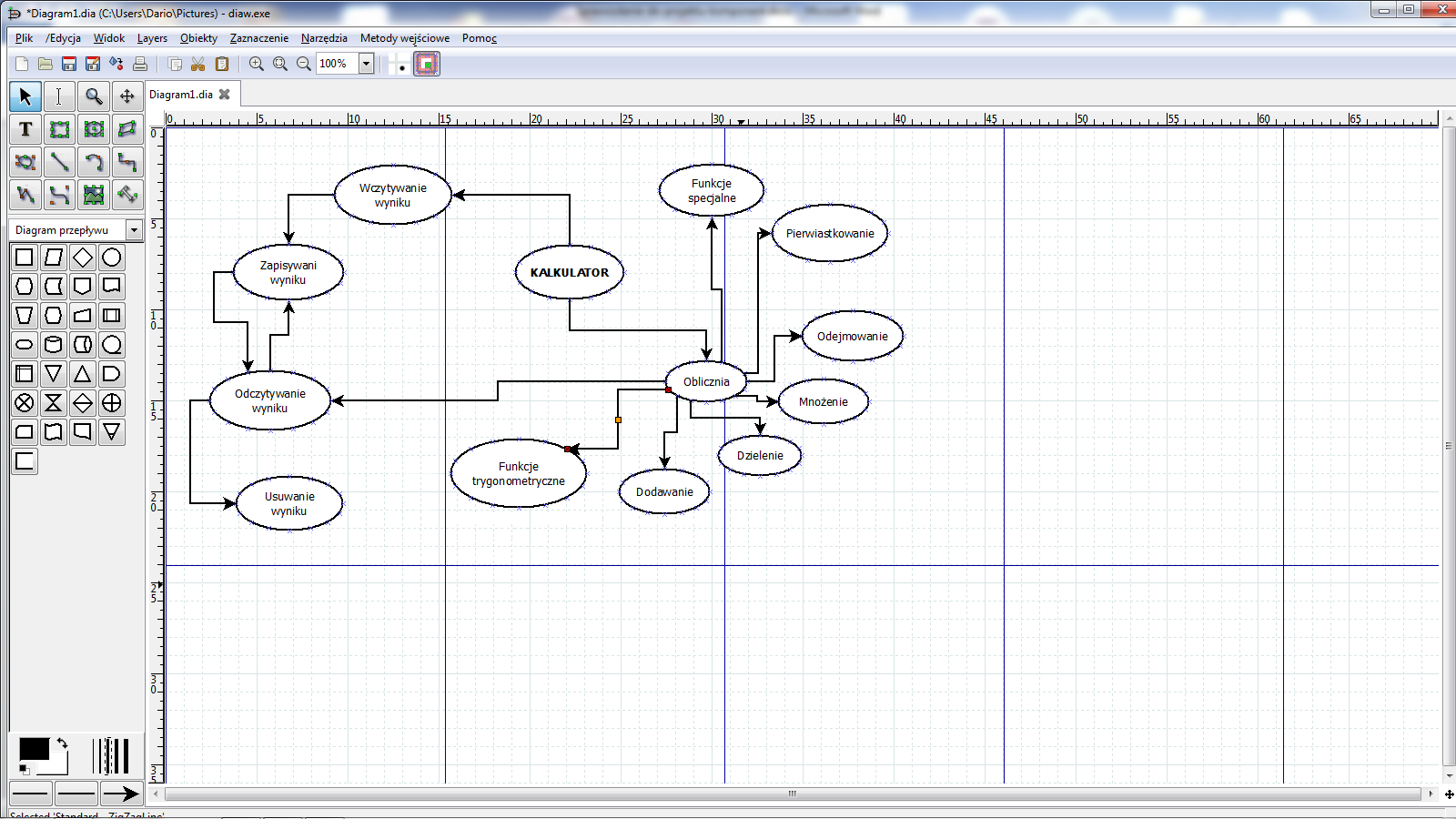
**Poza funkcjonalne**

* zmiana koloru przycisków
* zmiana czcionek na przyciskach
* zmiana wielkości przycisków
* zmiana układu przycisków
* zmiana koloru czcionki przycisków
* zmiana klawisza uruchomiającego przycisk
* zmiana wyglądu wyświetlacza
* zmiana układu wyświetlacza
* zmiana czcionki wyświetlacza

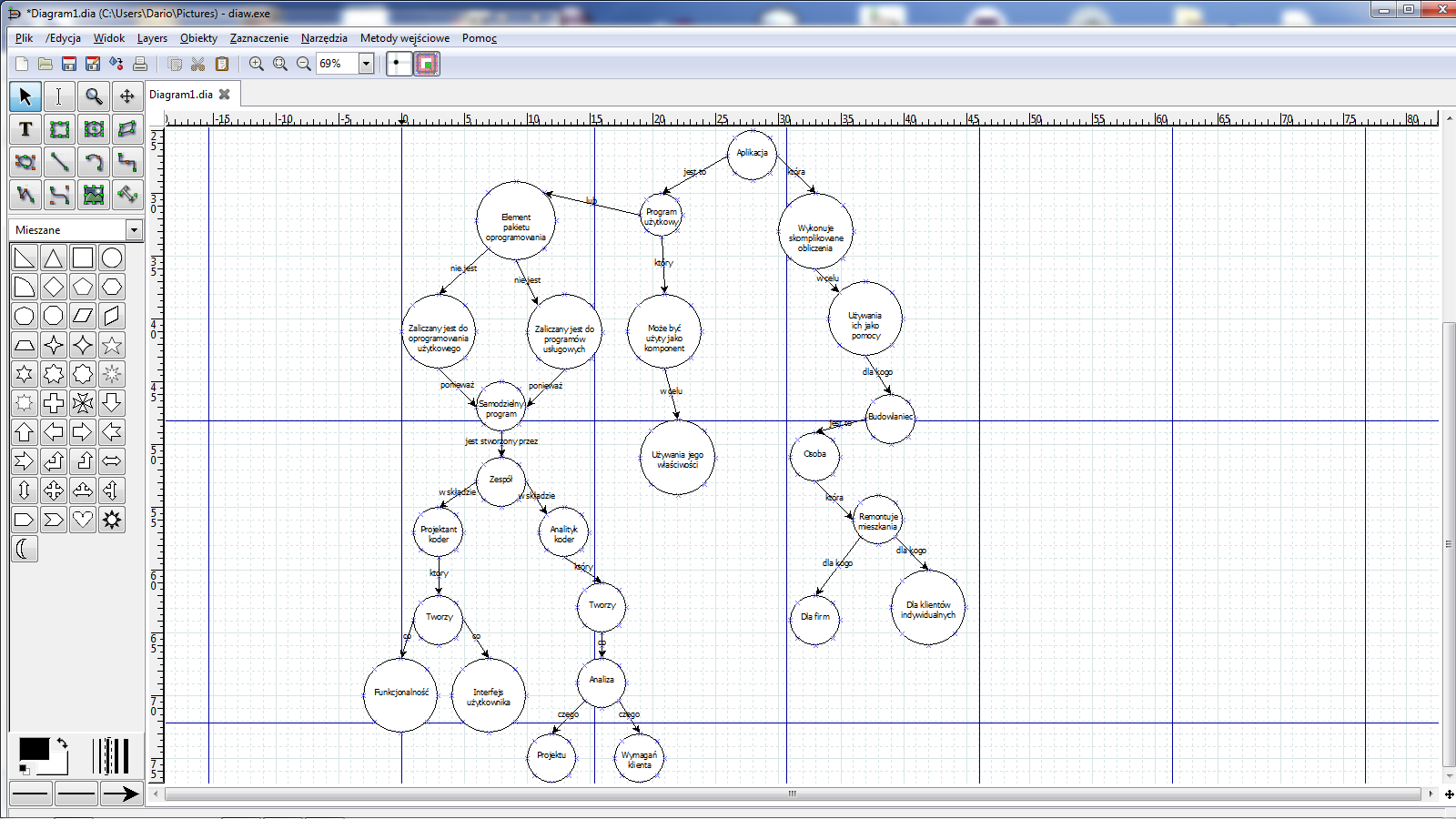
**Dziedzinowe**

* Procesor: zgodny z wymaganiami systemu operacyjnego
* Pamięć RAM: zgodna z wymaganiami systemu operacyjnego
* Twardy dysk: 50MB wolnej przestrzeni dyskowej
* Drukarka: laserowa lub atramentowa
* System operacyjny: Windows XP Service Pack 2, Windows Vista, Windows 7

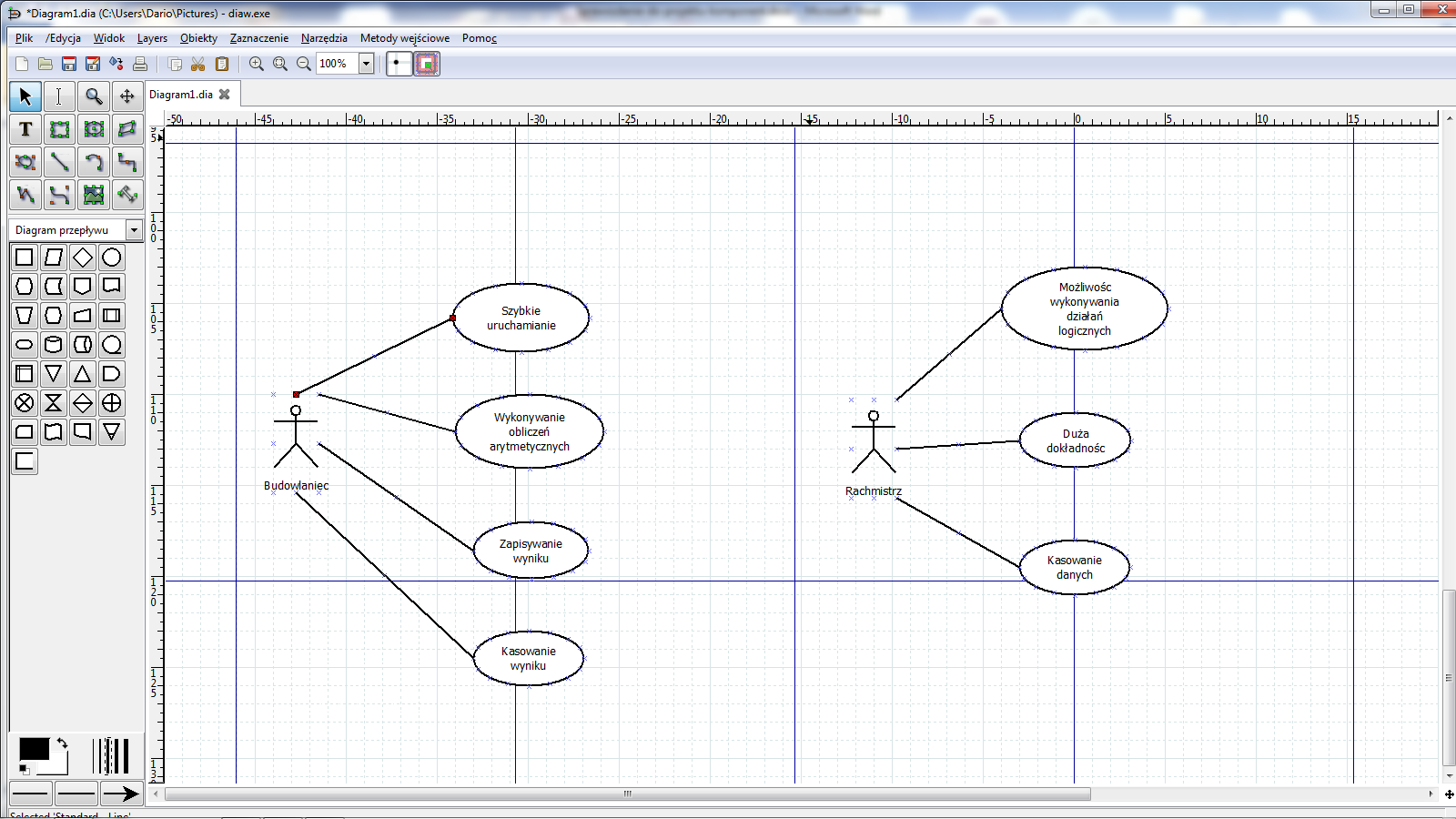
1. **Mapa myśli**



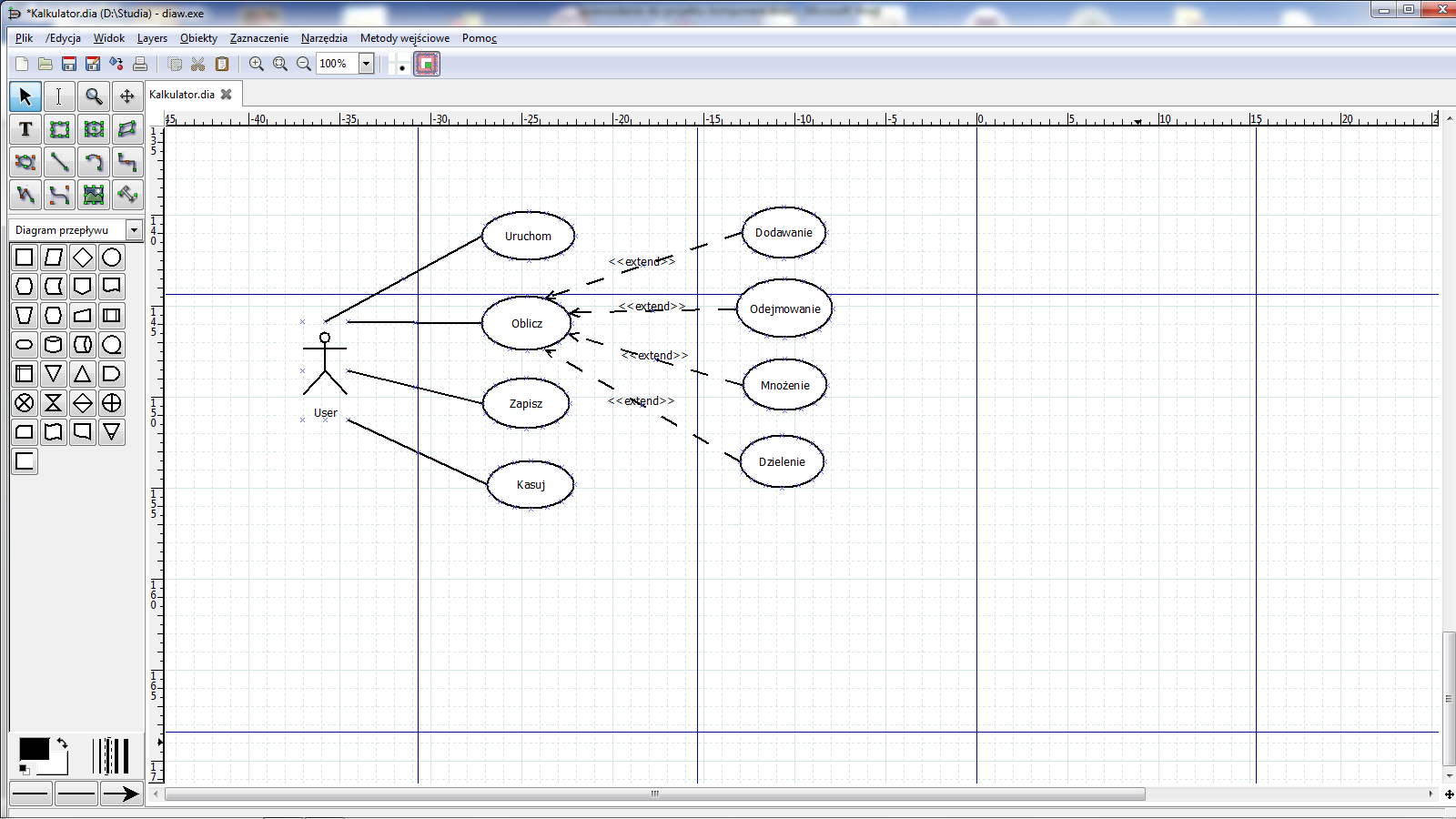
1. **Mapa konceptualna**



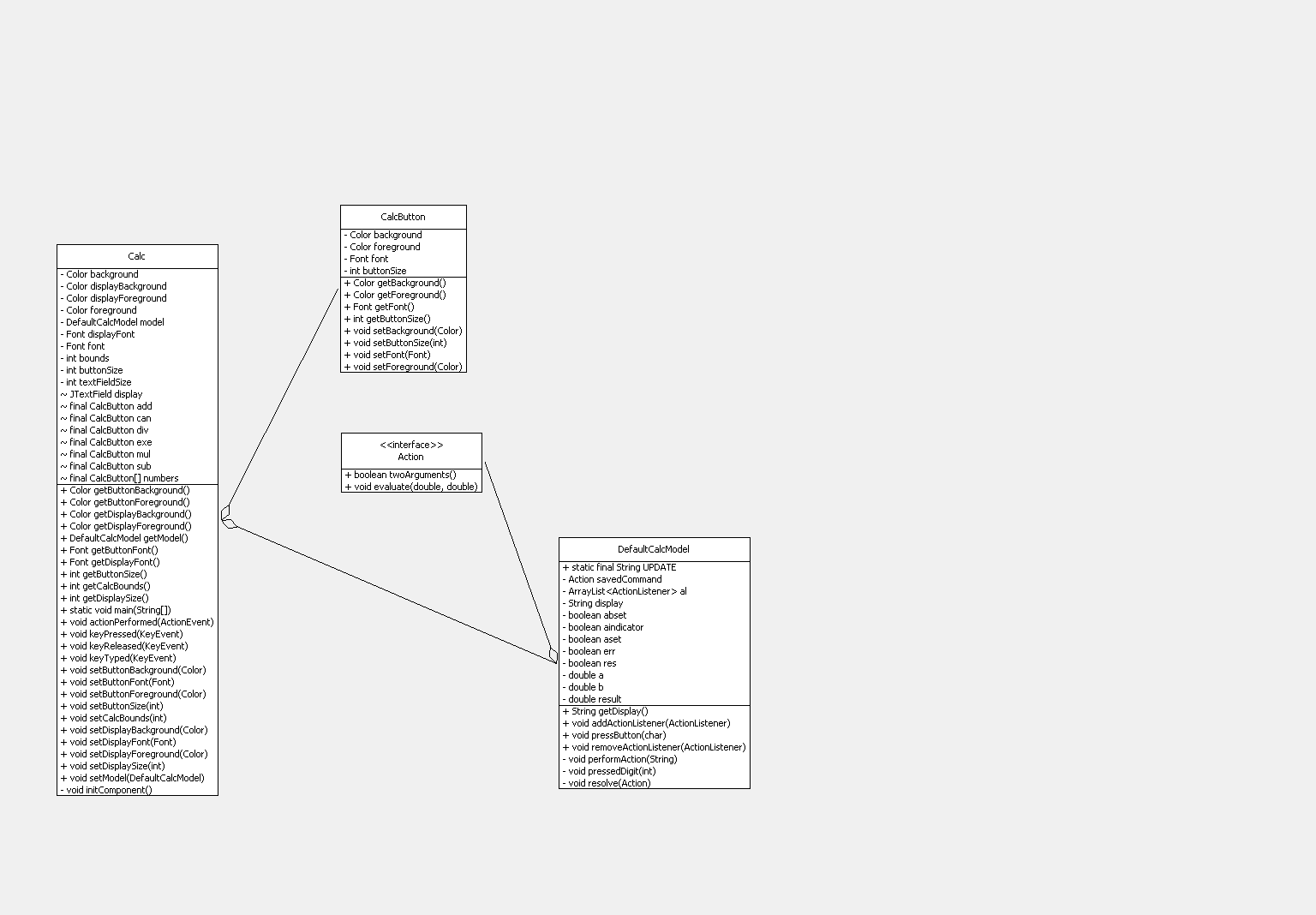
1. **Identyfikacja aktorów**



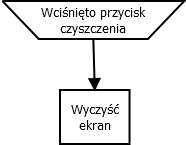
1. **Diagram przypadków użycia**



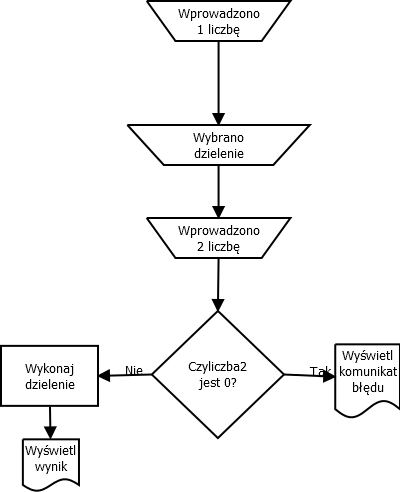
1. **Diagram klas**

****

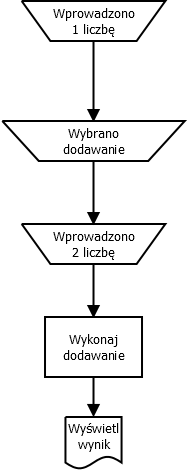
1. **Diagramy proceduralne**
2. **Czyszczenie**

****

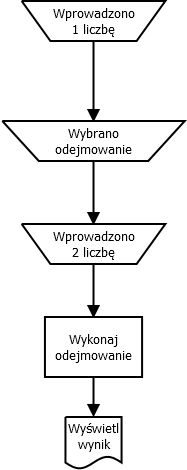
1. **Dzielenie**

****

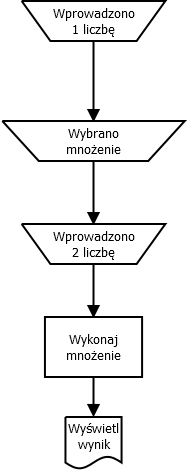
1. **Dodawanie**

****

1. **Odejmowanie**

****

1. **Mnożenie**

****

1. **Specyfikacja przypadków użycia**

|  |  |
| --- | --- |
| Funkcja | Dodawanie |
| Nazwa | Add |
| Klasa | DefaultCalcModel.java |
| Opis funkcji | Funkcja dodaje liczby wprowadzone przez użytkownika i wyświetla sumę w oknie wyniku. |
| Autorzy | KJ, TU |
| Dane wejściowe | Double |
| Dane wyjściowe | Double |

|  |  |
| --- | --- |
| Funkcja | Odejmowanie |
| Nazwa | Sub |
| Klasa | DefaultCalcModel.java |
| Opis funkcji | Funkcja odejmuje liczby wprowadzone przez użytkownika i wyświetla różnicę w oknie wyniku. |
| Autorzy | KJ, TU |
| Dane wejściowe | Double |
| Dane wyjściowe | Double |

|  |  |
| --- | --- |
| Funkcja | Dzielenie |
| Nazwa | Div |
| Klasa | DefaultCalcModel.java |
| Opis funkcji | Funkcja dzieli liczby wprowadzone przez użytkownika i wyświetla iloraz w oknie wyniku. |
| Autorzy | KJ, TU |
| Dane wejściowe | Double |
| Dane wyjściowe | Double |

|  |  |
| --- | --- |
| Funkcja | Mnożenie |
| Nazwa | Mul |
| Klasa | DefaultCalcModel.java |
| Opis funkcji | Funkcja mnoży liczby wprowadzone przez użytkownika i wyświetla iloczyn w oknie wyniku. |
| Autorzy | KJ, TU |
| Dane wejściowe | Double |
| Dane wyjściowe | Double |

1. **Zarys interfejsu użytkownika GUI**

****

1. **Fragmenty kodu**

**import java.awt.event.ActionEvent;**

**import java.awt.event.ActionListener;**

**import java.util.ArrayList;**

**public class DefaultCalcModel {**

**private double a,b,result;**

**public static final String UPDATE = "UPDATE";**

**private boolean abset=false;**

**private boolean aindicator=false;**

**private boolean aset =false;**

**private boolean res = false;**

**private boolean err = false;**

**private String display="";**

**private Action savedCommand;**

**private ArrayList<ActionListener>al = new ArrayList<ActionListener> ();**

**public void addActionListener (ActionListener alist) {**

**al.add(alist);**

**}**

**private void performAction (String action) {**

**for (int i=0; i<al.size(); i++)**

**al.get(i).actionPerformed(new ActionEvent (this, ActionEvent.ACTION\_PERFORMED, action));**

**}**

**public String getDisplay() {**

**// TODO Auto-generated method stub**

**return display;**

**}**

**public void pressButton(char command) {**

**if (command >= '0' && command <= '9') pressedDigit(command - '0');**

**if (command == '+') resolve (new Action(){**

**@Override**

**public void evaluate(double a, double b) {**

**result = a+b;**

**System.out.println(result);**

**}**

**@Override**

**public boolean twoArguments() {**

**return true;**

**}**

**});**

**if (command == '-') resolve (new Action(){**

**@Override**

**public void evaluate(double a, double b) {**

**result = a-b;**

**}**

**@Override**

**public boolean twoArguments() {**

**// TODO Auto-generated method stub**

**return true;**

**}**

**});**

**if (command == '/') resolve (new Action(){**

**@Override**

**public void evaluate(double a, double b) {**

**if (b==0) {**

**display = "ERR DIV BY 0";**

**err=true;**

**}**

**else result = a/b;**

**}**

**@Override**

**public boolean twoArguments() {**

**// TODO Auto-generated method stub**

**return true;**

**}**

**});**

**if (command == '\*') resolve (new Action(){**

**@Override**

**public void evaluate(double a, double b) {**

**result = a\*b;**

**}**

**@Override**

**public boolean twoArguments() {**

**// TODO Auto-generated method stub**

**return true;**

**}**

**});**

**if (command == '=') resolve (new Action(){**

**@Override**

**public void evaluate(double a, double b) {**

**if (savedCommand!=null)savedCommand.evaluate(a, b);**

**savedCommand = null;**

**res = true;**

**}**

**@Override**

**public boolean twoArguments() {**

**// TODO Auto-generated method stub**

**return false;**

**}**

**});**

**if (command == 'C') resolve (new Action(){**

**@Override**

**public void evaluate(double a, double b) {**

**if (!display.equals("")) {**

**display="";**

**}**

**else if (aset) {**

**display = ""+a;**

**aset = false;**

**}**

**else {**

**display = "";**

**}**

**}**

**@Override**

**public boolean twoArguments() {**

**// TODO Auto-generated method stub**

**return false;**

**}**

**});**

**}**

**private void resolve(Action action) {**

**if (display.equals("")) return;**

**if (aset) {**

**if (!action.twoArguments()) {**

**b = Double.parseDouble(display);**

**aindicator = true;**

**action.evaluate(a, b);**

**if (!err) display = ""+result;**

**else err=false;**

**abset=true;**

**aset=false;**

**}**

**}**

**if (!aset) {**

**if (!abset) {**

**a = Double.parseDouble(display);**

**aindicator = true;**

**aset = true;**

**if (action.twoArguments()) savedCommand = action;**

**else action.evaluate(a, b);**

**}**

**else abset = false;**

**}**

**performAction(UPDATE);**

**System.out.println ("display: |" + display + "| a: " + a + " b: " + b +**

**" ASET: " + aset + " savedAction: " +savedCommand);**

**}**

**private void pressedDigit(int i) {**

**if (res) {**

**display = "";**

**res = false;**

**}**

**if (aindicator) {**

**display = "";**

**aindicator = false;**

**}**

**if (display.equals("0.0)")) display = "";**

**display+=i;**

**System.out.println ("display: |" + display + "| a: " + a + " b: " + b +**

**" ASET: " + aset + " savedAction: " +savedCommand);**

**performAction(UPDATE);**

**}**

**public void removeActionListener(ActionListener a) {**

**al.remove(a);**

**}**

**}**