**PROI Projekt nr 3**

Zestaw klas reprezentujących logikę programu rozwiązującego zadania obliczeniowe z prostymi układami elektronicznymi

Czobot Stanisław I1

**Uściślenia projektu:**

* Projekt operuje na układach stałoprądowych.
* Projekt liczy rezystancję zastępczy, pojemność zastępczą, oraz natężenie prądu płynącego przez układ.
* Ponieważ w układach stałoprądowych kondensator stanowi rozwarcie, zatem do układu podczas mierzenia rezystancji zastępczej nie należy dodawać odnóg zawierających tego typu element.
* Możliwe są połączenia równoległe posiadające tylko dwie odnogi.
* Jeśli odnoga połączenia równoległego ma więcej niż jeden element to powinna być przedstawiona jako oddzielny obiekt Scheme i dodany jako pojedynczy element. Dzięki temu możliwe jest zastosowanie jest rekurencyjnej funkcji do mierzenia parametrów układu. Jednak takie rozwiązanie niesie za sobą pewne ograniczenia co do rozbudowy schematu, ponieważ w pewnym momencie może nam zabraknąć miejsca na stosie lub program będzie długo wykonywał obliczenia, oczywiście w przypadku prostych układów(testowane z 10 elementami i 3 rozgałęzieniami równoległymi) nie zauważyłem żadnych problemów.
* Występujące źródło napięciowe powinno być pierwszym dodanym do schematu elementem(lub można korzystać z przeciążonego konstruktora Scheme, który dodaje automatycznie źródło).

**Klasy wchodzące w skład projektu:**

1. Element, klasy pochodne:
   1. Resistor
   2. Capacitor
   3. VoltageSource
   4. Scheme
2. Node
3. Tests

**Opis poszczególnych klas:**

1. **Element** – wirtualna klasa, umożliwiająca użycie polimorfizmu podczas przechowywania elementów schematu układu. Zawiera pole enum eleType z informacją jakiego typu jest to element, jej klasy pochodne to:
   1. **Resistor** – klasa reprezentująca rezystor, zawiera wartość oporu(typu double), którą można pozyskać jako wynik metody double getValue().
   2. **Capacitor** – klasa reprezentująca kondensator, zawiera wartość pojemności(typu double), ), którą można pozyskać jako wynik metody double getValue().
   3. **VoltageSource** – klasa reprezentująca źródło napięcia, zawiera wartość wytwarzanego napięcia(typu double), którą można pozyskać jako wynik metody double getValue().
   4. **Scheme** – klasa stanowiąca główny człon projektu, zawiera ona wektor węzłów oraz wektor wskaźników na elementy. Jest ona klasą pochodną klasy element, aby można było jako element jednego schematu wkładać inny schemat, co jest konieczne np. w przypadku połączeń równoległych, gdy w jednej z jego odnóg jest więcej niż 1 element. Zawiera ona funkcje takie jak np. double calculateResistance(), które zwracają wartości odpowiednich parametrów układu. Aby dodać element do schematu trzeba posłużyć się odpowiednimi metodami, spośród: addNode, addElement, addScheme. W przypadku tej klasy metoda double getValue(); zwraca wartość -1, ponieważ musiała ona zostać nadpisana nad wirtualną metodę klasy Element.
2. **Node** – klasa reprezentująca węzeł układu elektronicznego. Zawiera jedno wejście oraz jedno lub dwa wyjścia. Każdo z wejść/wyjść zawiera numer z wektora elementów schematu, do którego prowadzi. Według własnych uściśleń węzeł może mieć tylko dwa wyjścia, zatem możliwe są tylko połączenia równoległe z dwiema odnogami.
3. **Tests** – klasa posiadająca funkcje do testowania niektórych z użyteczności klas wchodzących w skład projektu.

**Zabiegi do dalszej rozbudowy projektu:**

Najprostszym sposobem rozbudowy byłoby dodanie klas reprezentujących inne elementy elektroniczne, np. cewki, czy źródło prądowe, oraz metody do liczenia odpowiadających im wartości.

Dzięki przechowywaniu w klasie Scheme wektora elementów i węzłów stworzenie interfejsu graficznego nie powinno być problemem, ponieważ wystarczy je wszystkie narysować i połączyć według informacji zapisanych w węzłach.

Dodanie możliwości połączeń równoległych z więcej niż dwiema odnogami wymagałoby rozszerzenia klasy Node, a także zwiększenia poziomu rekurencji w metodach liczących parametry tychże połączeń.

Rozszerzenie projektu do liczenia także układów zmiennoprądowych wymagałoby zmian w podklasach Element, zwłaszcza zamiany reprezentacji wartości z typu double na typ odpowiadający wskazom zmiennoprądowym.