Laboratorium nr 8: Szablon dla układu równań liniowych

1 Cel ćwiczenia

Konstrukcje szablonów klas i funkcji. diagramu czynności oraz diagramu klas za pomocą języka UML.

2 Program zajęć

- *Ocena realizacji zadania z poprzedniego laboratorium* – ocenie podlega poprawność programu, kompilacja (kompilacja musi przebiegać bez ostrzeżeń), styl pisania programu oraz dokumentacja zrealizowana w systemie `doxygen`.
- *Realizacja wstępnej fazy prac nad nowym zadaniem* – w ramach wstępnej realizacji zadania należy stworzyć diagram diagram *czynności* oraz zaktualizować diagram *klas* (należy uwzględnić, że klasy `UkladRownanLiniowych`, *Macierz* oraz *Wektor* są w tym zadaniu szablonami). Diagram czynności powinien opisywać procedurę wyznaczania pierwiastków równania. Diagram można wykonać wykorzystując jeden z programów: `bouml`, `dia` lub `umbrello`. Wymienione programy dostępne są na panamincie. Na podstawie stworzonych diagramów należy wygenerować lub napisać definicje klas w języku C++.

  Następnie należy przerobić na szablon klasę `Wektor` i doprowadzić do poprawnej kompilacji i konsolidacji programu.
- *Ocena realizacji wstępnej fazy zadania*

3 Opis zadania programowego

Podstawą tego zadania jest program zrealizowany w ramach wcześniejszego zadania, który pozwala rozwiązać układ równań. Należy przerobić klasy `UkladRownanLinowych`, `Macierz` oraz `Wektor` (ewentualnie inne pomocnicze klasy, jeśli zostały użyte i zależą od typu danych elementów układu równań) na szablony klas. Działanie programu nie może różnić się od działania programu w zadaniu poprzednim. Należy też odpowiednio zaktualizować dokumentację. Przed przystąpieniem do przeróbki wcześniejszego zadania należy stworzyć odpowiednie diagramy: czynności oraz zaktualizować wcześniejszy diagram klas. Diagram czynności powinien opisywać procedurę wyznaczania pierwiastków równania.

Oprócz opisanych przeróbek należy rozszerzyć funkcjonalność programu o możliwość wywoływania programu z nazwą pliku. Program wówczas powinien czytać dane z pliku zamiast z wejścia standardowego.

3.1 Opis działania programu

Program ma działać identycznie jak w zadaniu poprzednim. Jedyną różnicą jest nowy sposób wywołania pliku. W wersji poprzedniej program czytał wszystko z wejścia standardowego. Można więc go było wywołać na kilka sposobów:

1. Bezpośrednie wywołanie programu i wprowadzanie danych z klawiatury:

```
./mnozenie
```

2. Przekierowanie na wejście standardowe programu zawartości pliku. Program czyta wówczas dane chociaż sam o tym *nie wie*.

```
./mnozenie < rownanie_liniowe.dane
```

3. Wykorzystanie mechanizmów potokowych. Skierowanie wyjścia standardowego innego program na wejście standardowe naszego programu.

```
cat rownanie_liniowe.dane | ./mnozenie
```

W tym zadaniu wprowadzona jest możliwość czytania z pliku, gdy pierwszym argumentem programu jest nazwa pliku. Pozwala to na wywołanie programu w formie:

4. Wywołanie programu z nazwą pliku danych:

```
./mnozenie rownanie_liniowe.dane
```

3.2 Wymagania co do konstrukcji programu

Obowiązują co do konstrukcji modułowej, uporządkowanej struktury katalogów, przeciążeń, dokumentacji itd., które były sformułowane na potrzeby wcześniejszych zadań.

3.3 Wersja uproszczona – oceniana nie więcej niż na 4,0

W wersji uproszczonej można ograniczyć się do stworzenia szablonu tylko dla klasy `Wektor` i `Macierz`.

3.4 Wersja bardzo uproszczona – oceniana nie więcej niż na 3,0

Oprócz wcześniej podanego uproszczenia można zrezygnować z wywołania programu z nazwą pliku.

3.5 Rozszerzenia nieobowiązkowe

Należy rozszerzyć plik `Makefile` i utworzyć odpowiednie moduły (o ile będzie to konieczne), aby możliwa była kompilacja programu dla typu `float`. Należy zaobserwować różnice wielkości błędów.