Zaawansowane C++: Zadanie I
Wstępny opis programu

Kamila Korzec

6 kwietnia 2018

1 Wybrany temat

Obliczenia symboliczne

Napisać program analizujący składnię wprowadzanej funkcji jednej zmiennej i konwertujący ją do ONP. Program ma mieć możliwość zapisu i odczytu funkcji, policzenia wartości funkcji w zadanym przedziale, oraz narysowania wykresów

2 Specyfikacja programu

Podstawowe funkcjonalności programu

Analiza składni i konwersja

Interfejs graficzny zawiera dwa pola tekstowe w których użytkownik może wpisać wzór funkcji w jednej z dwóch notacji. Program sprawdza, które pole zostało wypełnione przez użytkownika, analizuje jego treść i tworzy klasę zawierającą tokeny jednoznacznie określające wzór funkcji oraz metody zwracające ten wzór w obu notacjach. Program automatycznie wypełnia drugie (puste) pole tekstowe skonwertowanym wzorem. Jeśli użytkownik wypełni oba pola tekstowe, program wyświetla odpowiedni błąd w GUI.

Policzenie wartości funkcji

Po analizie wzoru, na podstawie zbioru tokenów, program jest w stanie obliczyć wartość funkcji dla dowolnego punktu lub obsłużyć stosowny wyjątek, jeśli wartość funkcji w punkcie nie istnieje. Wartości funkcji obliczane są w określonym przez użytkownika zakresie ze stałym, odpowiednio małym krokiem. Zestawienie punktów i wartości są wyświetlone w GUI z określoną precyzją. Punkty, w których wartość nie istnieje, są oznaczone jako "N/A". Punkty, w których wartość osiąga nieskończoność są wyświetlone jako "±Infty"

Zapis funkcji

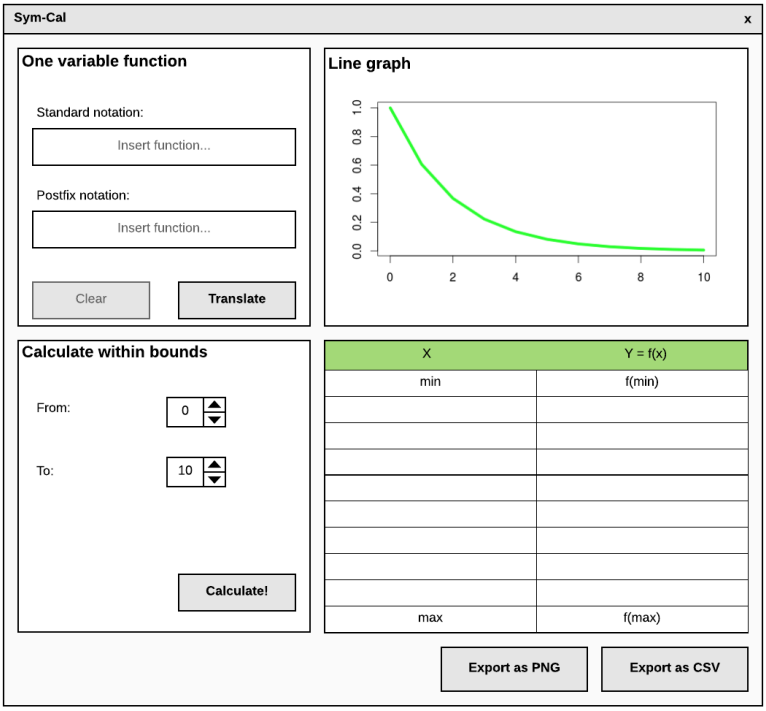
Użytkownik może zapisać wzór funkcji w obu notacjach oraz wyeksportować obliczone wartości do pliku tekstowego/CSV. Dane w pliku sformatowane są w następujący sposób:

```
Traditional notation: a*x+b
Postfix notation: a x * b +
X, f(X)
x, y
```

Wykres

Po obliczeniu punktów z określonego przedziału w interfejsie graficznym rysowany jest wykres. Przedziały i skala są automatycznie dobierane na podstawie przedziału określonego przez użytkownika i obliczonych wartości. Wykres może zostać wyeksportowany do pliku graficznego.

Projekt interfejsu graficznego



Przykładowe scenariusze użytkownika

Scenariusz 1

Użytkownik:

1. wpisuje poprawny wzór funkcji jednej zmiennej i naciska przycisk ”Translate”
2. odczytuje wzór w ONP
3. określa zakres w jakim funkcja powinna być obliczona i naciska przycisk ”Calculate”
4. odczytuje przybliżone wartości funkcji z wykresu
5. odczytuje dokładne wartości funkcji z tabeli
6. eksportuje wartości do pliku CSV

Scenariusz 2

Użytkownik:

1. wpisuje niepoprawny wzór funkcji jednej zmiennej (użyte niepoprawne symbole, funkcja wielu zmiennych) i naciska przycisk ”Translate”
2. odczytuje komunikat błędu: ”Invalid function” w miejscu, w którym powinien pojawić się przekonwertowany wzór
3. czyści oba pola tekstowe przyciskiem ”Clear”
4. wykonuje kroki opisane w scenariuszu 1

Scenariusz 3

Użytkownik:

1. wpisuje poprawny wzór funkcji jednej zmiennej w ONP i naciska przycisk "Translate"
2. odczytuje wzór w tradycyjnej notacji
3. określa zakres w jakim funkcja powinna być obliczona i naciska przycisk "Calculate"
4. odczytuje kształt funkcji z wykresu
5. zapisuje wykres do pliku graficznego

3 Wstępny projekt wykonania programu

Projekt zakłada wykorzystanie biblioteki Qt do zaimplementowania interfejsu graficznego oraz Tokenizera (implementacja własna lub biblioteka zewnętrzna) umożliwiającego przekształcenie wyrażenia podanego przez użytkownika do odpowiedniej klasy.

Opis głównych klas w programie

Poniżej opisano planowane cztery główne klasy. W przypadku dwóch klas określono podstawowe metody i właściwości klasy. Implementacja dwóch pozostałych klas zależy od wykorzystania bibliotek zewnętrznych. Projekt i funkcjonalności klas mogą się zmienić w zależności od potrzeb.

OneVarFunction

```
OneVarFunction {
  static OneVarFunction fromSuffixNotation(string input) {};
  static OneVarFunction fromStandardNotation(string input) {};
  string   toSuffixNotation() {};
  string   toStandardNotation() {};
  Point[] calculateValues(int from, int to) {};

  Token[] representation;
}
```

Point

```
Point {
  float x;
  float y;
}
```

Token

Interfeja klasy Token jest jeszcze nieznany - w zależności od użytego Tokenizera użyta klasa może pochodzić z biblioteki zewnętrznej albo z biblioteki standardowej. Implementacja klasy powininna umożliwiać jednoznaczną interpretacje zawartości tokena.

Graph

Interfejs klasy graph jest jeszcze nieznany - klasa prawdopodobnie będzie zaimplementowana z wykorzystaniem biblioteki Qt. Klasa powinna wspierać rysowanie wykresu na podstawie zadanych punktów.

UI - Klasy biblioteki Qt

Interfejs graficzny programu będzie zaimplementowany z wykorzystaniem biblioteki Qt - w projekcie zostaną użyte klasy umożliwiające obsługę podstawowych inputów (tekst i liczba) oraz klasy umożliwiające wizualizację danych.

4 Repozytorium

Link do repozytorium projektu: https://github.com/kamilakorzec/symcal

5 Zagadnienia do rozważenia

1. Jaki tokenizer?
2. Rozpoznawanie zmiennej - x czy dowolna litera?
3. Rozpoznawanie i obsługa stałych (e, π)
4. Rozpoznawanie niestandardowych wyrażeń (potęgi, pierwiastki)
5. Wyświetlanie błędów - osobne okienko, stałe miejsce w UI czy zależne od kontekstu?