My Project

Generated by Doxygen 1.9.2

Chapter 1

Laboratorium 1 - ułamki

1.0.1 Tresc zadan dla Panstwa:

Zadanie 0: absolutnie obowiazkowe, chociaz bez punktow

1. Pierwsza rzecza jest poprawa bledow kompilacji, czyli wpisanie poprawnych Panstwa danych w pliku: main.↩ cpp
2. Oddanne zadanie musi sie bezwzglednie kompilowac na systemie Linux:
   - Jesli sie nie skompiluje to jest to 0 punktow za zadanie!
   - Oczywiscie w razie problemow z kompilacja prosze się zgłaszać/pisać.
   - Dobrze, jesli nie byloby warningow kompilacji, ale za to nie obnizam punktow.
   - Aby się upewnić, że się kompiluje można skorzystać z `narzędzia online judge` (VPN AGH konieczny). Aby wyslac zadanie nalezy wybrac konkurs (*C++ lab1*), problem (*lab1*), oraz jezyk programowania (*c++*), prosze zalaczyc obydwa pliki fraction.h, fraction.cpp, jak na obrazku: @ image html domjudge.png @ image latex domjudge.png
3. Oddane zadanie nie powinno crashować na żadnym teście, jeśli crashuje proszę zrobic implementacje -fake, ktora nie dopusci do crasha nawet jesli test bedzie failowal, ale za to testy nie beda sie crashowaly. W przypadku crasha biorę pod uwagę tylko tyle testów, ile przejdzie do czasu crasha!
4. Mam program antyplagiatowy, dlatego prosze pracowac samodzielnie!
   - Osoby ktore udostepniaja swoje rozwiazania rowniez beda mialy kare!
   - Na ukaranie prowadzący ma czas 2 tygodnie po terminie oddania, czyli nawet jak ktoś otrzyma punkty wcześniej ma pewność, że za oszustwa/łatwowierność dosięgnie go niewidzialna ręka sprawiedliwości.
5. Zadanie z zalozenia bedzie sprawdzane automatycznie, przez testy jednostkowe dostepne w pliku: fraction↩ Tests.cpp,
6. ∗Dobrze jakby nie bylo warningow kompilacji (flagi: `-Wall -Wextra -pedantic -Werror`, a dla hardcorów jeszcze: `-Weffc++`)
7. Punkty beda odejmowane za wycieki pamieci (jest podpiety `valgrind`)
8. Niewykluczone jest sprawdzanie reczne - zaleznie od prowadzacego dana grupe.

Tresc do implementacji - szukaj w plikach ∗.h

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

File Documentation

3.1 fraction.h File Reference

Zaimplementuj podaną na zajęciach klasę reprezentującą ułamek:

```
#include <iosfwd>
#include <string>
```

Macros

- #define **IMPLEMENTED_classFractionExists** 0
- #define **IMPLEMENTED_hasNumeratorAndDenominator** 0
- #define **IMPLEMENTED_hasDefaultConstructor** 0
- #define **IMPLEMENTED_hasConstructorWhichInitialiseFields** 0
- #define **IMPLEMENTED_hasGettersAndSetters** 0
- #define **IMPLEMENTED_hasPrintFunction** 0
- #define **IMPLEMENTED_counterOfDestructedFractionsImplemented** 0
- #define **IMPLEMENTED_readWriteImplemented** 0
- #define **IMPLEMENTED_fractionNameSettableFromConstructor** 0
- #define **IMPLEMENTED_fractionConstStaticFieldsImplemented** 0

3.1.1 Detailed Description

Zaimplementuj podaną na zajęciach klasę reprezentującą ułamek:

1. Nazwa klasy Fraction), po której zdefiniowaniu zmień: w makrze `IMPLEMENTED_classFraction↩Exists 0` -> `IMPLEMENTED_classFractionExists 1`
2. Do klasy dodaj pola protected: `numerator` (licznik) i `denominator` (mianownik) następnie ustaw jedynkę obok `IMPLEMENTED_hasNumeratorAndDenominator`
3. Do klasy dodaj konstruktor bezparametrowy (może być też z wartościami domyślnymi), który ustawi wartości licznika na 0 i mianownika na 1 następnie ustaw jedynkę obok `IMPLEMENTED_hasDefault↩Constructor`

4. Do klasy dodaj konstruktor z parametrami (może być modyfikacja powyższego), który ustawi licznik i mianownik na podstawie parametrów, a) wartością domyślną dla mianownika ma być 1 następnie ustaw jedynkę obok `IMPLEMENTED_hasConstructorWhichInitialiseFields`

5. Do klasy dodaj zestaw metod dostępowych/modyfikujących obiekty klasy -tzw. gettery i settery, które umożliwią modyfikacje i pobranie wartości licznika i mianownika następnie ustaw jedynkę obok `IMPLEMENTED_hasGettersAndSetters`

6. Do klasy dodaj metodę print(), wypisującą atrybuty obiektu na konsolę w formie "licznik/mianownik\n" następnie ustaw jedynkę obok `IMPLEMENTED_hasPrintFunction`

7. Do klasy dodaj statyczny atrybut (removedFractions_) i metodę statyczną: removedFractions() zwracającą ten atrybut Skladowa ta powinna byc incrementowana w destruktorze następnie ustaw jedynkę obok `IMPLEMENTED_counterOfDestructedFractionsImplemented`

8. Dodaj do klasy metody zapisz(std::ostream& os)/wczytaj(std::istream& is), zapisujące/odczytujące zawartość obiektu do przekazanego jako parametr strumienia wyjściowego/wejściowego, w formacie "licznik/mianownik". Metoda wczytująca może założyć, że format danych będzie poprawny (nie trzeba obslugiwac bledow). następnie ustaw jedynkę obok `IMPLEMENTED_readWriteImplemented`.

9. Proszę dodanie stałej odzwierciedlającej nazwę ułamka o nazwie `fractionName`, powinna być protected. Do niej proszę dodać metodę stałą: `getFractionName()` następnie ustaw jedynkę obok `IMPLEMENTED_fractionNameSettableFromConstructor`

10. Do klasy dodaj dwie stałe: a) stała statyczna odzwierciedlajaca niedopuszczalna wartość mianownika: `invalidDenominatorValue` b) metodę statyczną zwracającą powyższe: `getInvalidDenominatorValue()` c) stałą statyczną czasu kompilacji (constexpr) odzwierciedlajaca domyślna wartość mianownika: `defaultDenominatorValue` d) metodę constexpr `getDefaultDenominatorValue()` zwracającą powyższe następnie ustaw jedynkę obok `IMPLEMENTED_fractionConstStaticFieldsImplemented`

Po implementowaniu powyższych poleceń i zmiany wartości poniższych makr powinno przechodzić coraz więcej testów dostępnych w pliku fractionTests.cpp.

3.1.2 Pytania po implementacji ćwiczenia:

Note

A. Jaka jest różnica między składowymi: `const static` a `static`?

B. Jaka jest różnica między składowymi: `const static` a `constexpr static`?

3.2 fraction.h

Go to the documentation of this file.

```
1 #ifndef TASK1_FRACTION_H
2 #define TASK1_FRACTION_H
3
4 #include <iosfwd>
5 #include <string>
6
54 #define IMPLEMENTED_classFractionExists 0
55 #define IMPLEMENTED_hasNumeratorAndDenominator 0
56 #define IMPLEMENTED_hasDefaultConstructor 0
57 #define IMPLEMENTED_hasConstructorWhichInitialiseFields 0
58 #define IMPLEMENTED_hasGettersAndSetters 0
59 #define IMPLEMENTED_hasPrintFunction 0
60 #define IMPLEMENTED_counterOfDestructedFractionsImplemented 0
61 #define IMPLEMENTED_readWriteImplemented 0
62 #define IMPLEMENTED_fractionNameSettableFromConstructor 0
63 #define IMPLEMENTED_fractionConstStaticFieldsImplemented 0
64
68 // .. TODO:
69
70 #endif // TASK1_FRACTION_H
```

Index