Nazwisko i imię: **Soczyński Krzysztof**

Kierunek: **Informatyka w inżynierii komputerowej**

Rok: **I** Semestr: **I** Forma studiów: **niestacjonarne**

Grupa: **Lk4**

Przedmiot: **Podstawy programowania w C/C++**

Zajęcia: **Laboratorium komputerowe 2** Data: **27.11.2022**

1. Treść zadania
   1. *Napisz program obliczający oraz n-krotnie.*
   2. *Napisz program mnożący dwie macierze.*
2. Kod programu









1. Opis programu
   1. Dołączone biblioteki

* <cmath>  
  biblioteka dająca dostęp do funkcji matematycznych zawartych w bibliotece standardowej C++
* <iostream>  
  biblioteka pozwalająca wczytywać dane i wypisywać dane z/do strumienia wejścia/wyjścia
* <numbers>  
  biblioteka dająca dostęp do stałych matematycznych (dostępna od C++20)
* <stdexcept>  
  biblioteka dająca możliwość tworzenia std::runtime\_error
* <vector>  
  biblioteka udostępniająca tablicę dynamiczną  
  1. Kod „linijka po linijce”

  
Funkcja obliczająca dowolną potęgę całkowitą liczby a. Użycie template pozwala przekazać do funkcji parametr dowolnego typu zamiast przeładowywać funkcję kilka razy. Korzystając z operatora warunkowego można w krótki sposób zapisać zwracanie zależne od tego czy potęga jest dodatnia czy ujemna.

  
Funkcja obliczająca silnię z liczby x.  
  
Funkcja obliczająca korzystając ze wzoru . Niedokładność wynikająca z zapisu liczb zmiennoprzecinkowych w komputerach niwelowana jest warunkiem sprawdzającym różnicę pomiędzy dwoma kolejnymi wynikami sumowania i gdy jest ona dostatecznie mała uznajemy wynik za poprawny.

  
Funkcja obliczająca n-krotne podniesienie .

  
Funkcja main() programu. Użytkownik podaje x – potęgę do której podniesiemy e. Następnie wyliczamy trzema dostępnymi metodami: funkcją exp() z biblioteki standardowej języka oraz napisanymi funkcjami myExp() i myPow() i wyświetlamy wyniki dla porównania.

  
Definicja bardzo prostej klasy Matrix. Wartości przechowywany w macierzy dzięki użyciu szablonu mogą być dowolnego typu. Klasa udostępnia pięć konstruktorów:

* domyślny
* przyjmujący rozmiary macierzy
* przyjmujący vector<vector<T>> przechowujący dane do zapisania w macierzy
* domyślny konstruktor kopiujący
* domyślny konstruktor przenoszenia

Przeładowanie operatora \* pozwala na wykonanie mnożenia macierzy. Operacja może wyrzucić wyjątek std:runtime\_error w sytuacji kiedy któraś z mnożonych macierzy jest pusta lub nie został spełniony warunek zgodności ilości kolumn macierzy A z ilością wierszy macierzy B.  
Przeładowanie operatora [] pozwala uzyskać dostęp do danych zawartych w macierzy.  
Funkcja empty() zwraca true jeżeli macierz jest pusta.  
Funkcja numRows() zwraca liczbę wierszy macierzy.  
Funkcja numColumns() zwraca liczbę kolumn macierzy.  
[[nodiscard]] informuje kompilator, że wartość zwracana z funkcji nie powinna zostać zignorowana, a jeżeli taka sytuacji nastąpi kompilator zgłosi ostrzeżenie.  
noexcept(false) oznacza, że funkcja może potencjalnie wyrzucić wyjątek.

  
Funkcja main() programu. Tworzymy dwa obiekty stworzonej klasy Matrix. W bloku   
try-catch wykonujemy mnożenie macierzy oraz wyświetlamy macierz wynikową. Try-catch pozwala przechwycić wyjątki i obsłużyć je nie doprowadzając do crasha programu.

1. Wyjście programu



