Obliczanie wartości Pi

5

Wygenerowano za pomocą Doxygen 1.12.0

Rozdział 1

Indeks klas

1.1 Lista klas

Tutaj znajdują się klasy, struktury, unie i interfejsy wraz z ich krótkimi opisami:

Rozdział 2

Indeks plików

2.1 Lista plikÃ^{3}w

Tutaj znajduje siÄ lista wszystkich plikÃ^{3}w wraz z ich krÃ^{3}tkimi opisami:

Rozdział 3

Dokumentacja klas

3.1 Dokumentacja klasy PiCalculator

A class to calculate the value of Pi using numerical integration.

`#include <integral.h>`

Metody publiczne

- PiCalculator (long long intervals, int threads)

  Constructs a PiCalculator with the specified number of intervals and threads.
- double calculate ()

  Calculates the value of Pi using numerical integration.
- double getExecutionTime () const

  Retrieves the execution time of the last calculation.

Metody prywatne

- double computeSegment (long long start, long long end)

  Computes the contribution to Pi from a segment of the interval.

Statyczne metody prywatne

- static void worker (PiCalculator ∗calculator, long long start, long long end, double &result)

  Worker function for parallel computation of Pi segments.

Atrybuty prywatne

- long long intervals_

  The total number of intervals for integration.
- int threads_

  The number of threads for parallel computation.
- double executionTime_

  The execution time of the last calculation.
- std::vector< double > results_

  Storage for partial results from each thread.

3.1.1 Opis szczegółowy

A class to calculate the value of Pi using numerical integration.

Definicja w linii 45 pliku integral.h.

3.1.2 Dokumentacja konstruktora i destruktora

3.1.2.1 PiCalculator()

```
PiCalculator::PiCalculator (
            long long intervals,
            int threads)
```

Constructs a PiCalculator with the specified number of intervals and threads.

Konstruktor klasy PiCalculator.

Parametry

intervals	The number of intervals for numerical integration.
threads	The number of threads to use for parallel computation.
intervals	Liczba przedziałów dla całkowania numerycznego.
threads	Liczba wątków do wykorzystania w obliczeniach.

Definicja w linii 29 pliku Intergral.cpp.

3.1.3 Dokumentacja funkcji składowych

3.1.3.1 calculate()

```
double PiCalculator::calculate ()
```

Calculates the value of Pi using numerical integration.

Oblicza wartość liczby Pi za pomocą numerycznej całki prostokątów.

Zwraca

The computed value of Pi.

Obliczenia są podzielone na segmenty, które są przetwarzane równolegle przez wiele wątków. Następnie wyniki częściowe są sumowane, a czas wykonywania jest mierzony.

Zwraca

Przybliżona wartość liczby Pi.

Definicja w linii 42 pliku Intergral.cpp.

3.1.3.2 computeSegment()

```
double PiCalculator::computeSegment (
            long long start,
            long long end)  [private]
```

Computes the contribution to Pi from a segment of the interval.

Oblicza wkład w wartość liczby Pi z danego segmentu przedziału.

Parametry

start	The start index of the interval segment.
end	The end index of the interval segment.

Zwraca

The computed partial result for the segment.

Parametry

start	Początek segmentu (indeks).
end	Koniec segmentu (indeks).

Zwraca

Wynik częściowy dla podanego segmentu.

Definicja w linii 88 pliku Intergral.cpp.

3.1.3.3 getExecutionTime()

```
double PiCalculator::getExecutionTime () const
```

Retrieves the execution time of the last calculation.

Zwraca czas wykonania ostatniego obliczenia.

Zwraca

The execution time in seconds.

Czas wykonania w sekundach.

Definicja w linii 77 pliku Intergral.cpp.

3.1.3.4 worker()

```
void PiCalculator::worker (
            PiCalculator * calculator,
            long long start,
            long long end,
            double & result)  [static], [private]
```

Worker function for parallel computation of Pi segments.

Funkcja robocza używana przez wątki do obliczania segmentów całki.

Parametry

calculator	A pointer to the PiCalculator instance.
start	The start index of the interval segment.
end	The end index of the interval segment.
result	Reference to a double where the result will be stored.
calculator	Wskaźnik na instancję klasy PiCalculator.
start	Początek segmentu (indeks).
end	Koniec segmentu (indeks).
result	Referencja na zmienną, w której zostanie zapisany wynik częściowy.

Definicja w linii 106 pliku Intergral.cpp.

3.1.4 Dokumentacja atrybutów składowych

3.1.4.1 executionTime_

```
double PiCalculator::executionTime_  [private]
```

The execution time of the last calculation.

Definicja w linii 86 pliku integral.h.

3.1.4.2 intervals_

```
long long PiCalculator::intervals_  [private]
```

The total number of intervals for integration.

Definicja w linii 84 pliku integral.h.

3.1.4.3 results_

```
std::vector<double> PiCalculator::results_  [private]
```

Storage for partial results from each thread.

Definicja w linii 87 pliku integral.h.

3.1.4.4 threads_

```
int PiCalculator::threads_  [private]
```

The number of threads for parallel computation.

Definicja w linii 85 pliku integral.h.

Dokumentacja dla tej klasy została wygenerowana z plików:

- Intergral/integral.h
- Intergral/Intergral.cpp

Rozdział 4

Dokumentacja plików

4.1 Dokumentacja pliku Intergral/integral.h

Deklaracja klasy PiCalculator do obliczania wartoci liczby Pi metod¹ numerycznej ca^{3}ki.

```
#include <vector>
#include <thread>
```

Komponenty

- class PiCalculator

  A class to calculate the value of Pi using numerical integration.

4.1.1 Opis szczegÃ^{3}Åowy

Deklaracja klasy PiCalculator do obliczania wartoci liczby Pi metod¹ numerycznej ca^{3}ki.

Biblioteka ta pozwala na obliczenie wartoci liczby Pi za pomoc¹ metody numerycznej ca^{3}ki prostok¹tów. Obliczenia mog¹ byæ równolegle wykonywane z u¿yciem wielow¹tkowoci, co przyspiesza czas wykonywania w przypadku du¿ych zakresów obliczeñ.

4.1.1.1 Kluczowe cechy:

- Obs^{3}uga podzia^{3}u pracy na wiele w¹tków.
- Pomiar czasu wykonywania obliczeñ.
- Wysoka precyzja dziêki du¿ej liczbie przedzia^{3}ów.

4.1.1.2 Wykorzystywane biblioteki:

- <vector>: Przechowuje wyniki czêciowych obliczeñ dla poszczególnych w¹tków.
- <thread>: Pozwala na uruchamianie w¹tków do równoleg^{3}ego przetwarzania segmentów danych.

4.1.1.3 Przyk³ad u¿ycia:

```
#include "integral.h"

int main() {
    PiCalculator calculator(1000000, 4); // 1 000 000 przedzia³ów, 4 w¹tki
    double pi = calculator.calculate();
    double time = calculator.getExecutionTime();

    std::cout « "Pi = " « pi « std::endl;
    std::cout « "Czas wykonania: " « time « " sekund" « std::endl;
    return 0;
}
```

Nota

Liczba przedzia³ów i w¹tków wp³ywa na precyzjê i czas obliczeñ.

Definicja w pliku integral.h.

4.2 integral.h

Idź do dokumentacji tego pliku.

```
00001
00037 #pragma once
00038 #include <vector>
00039 #include <thread>
00040
00045 class PiCalculator {
00046 public:
00052     PiCalculator(long long intervals, int threads);
00053
00058     double calculate();
00059
00064     double getExecutionTime() const;
00065
00066 private:
00073     double computeSegment(long long start, long long end);
00074
00082     static void worker(PiCalculator* calculator, long long start, long long end, double& result);
00083
00084     long long intervals_;
00085     int threads_;
00086     double executionTime_;
00087     std::vector<double> results_;
00088 };
```

4.3 Dokumentacja pliku Intergral/Intergral.cpp

```
#include "integral.h"
#include <iostream>
#include <cmath>
#include <chrono>
#include <mutex>
```

4.4 Intergral.cpp

Idź do dokumentacji tego pliku.

```
00001
00018 #include "integral.h"
00019 #include <iostream>
00020 #include <cmath>
```

```
00021 #include <chrono>
00022 #include <mutex>
00023
00029 PiCalculator::PiCalculator(long long intervals, int threads)
00030     : intervals_(intervals), threads_(threads), executionTime_(0) {
00031 }
00032
00042 double PiCalculator::calculate() {
00043     results_.resize(threads_);
00044     auto start_time = std::chrono::high_resolution_clock::now();
00045
00046     // Tworzenie i uruchamianie wątków.
00047     std::vector<std::thread> workers;
00048     long long segment_size = intervals_ / threads_;
00049     for (int i = 0; i < threads_; ++i) {
00050         long long start = i * segment_size;
00051         long long end = (i == threads_ - 1) ? intervals_ : (i + 1) * segment_size;
00052         workers.emplace_back(worker, this, start, end, std::ref(results_[i]));
00053     }
00054
00055     // Oczekiwanie na zakończenie pracy wątków.
00056     for (auto& thread : workers) {
00057         thread.join();
00058     }
00059
00060     auto end_time = std::chrono::high_resolution_clock::now();
00061     std::chrono::duration<double> elapsed = end_time - start_time;
00062     executionTime_ = elapsed.count();
00063
00064     // Sumowanie wyników częściowych.
00065     double total_sum = 0.0;
00066     for (double partial : results_) {
00067         total_sum += partial;
00068     }
00069
00070     return total_sum * 4.0; // Mnożenie przez 4 zgodnie z wzorem na całkę liczby Pi.
00071 }
00072
00077 double PiCalculator::getExecutionTime() const {
00078     return executionTime_;
00079 }
00080
00088 double PiCalculator::computeSegment(long long start, long long end) {
00089     double sum = 0.0;
00090     double step = 1.0 / intervals_;
00091     for (long long i = start; i < end; ++i) {
00092         double x = (i + 0.5) * step;
00093         sum += 1.0 / (1.0 + x * x);
00094     }
00095     return sum * step; // Skalowanie wyniku na podstawie długości kroku.
00096 }
00097
00106 void PiCalculator::worker(PiCalculator* calculator, long long start, long long end, double& result) {
00107     result = calculator->computeSegment(start, end);
00108 }
```

4.5 Dokumentacja pliku Intergral/main.cpp

G³ówny plik programu do obliczania liczby Pi za pomoc¹ numerycznej ca³ki.

```
#include <iostream>
#include <iomanip>
#include <fstream>
#include "integral.h"
```

Funkcje

- void runTests ()

  < Nag³ówek klasy PiCalculator, zawieraj¹cy deklaracje metod obliczaj¹cych liczbê Pi.

- int main ()

  Funkcja g³ówna programu, która umo¿liwia wybór trybu dzia³ania programu.

4.5.1 Opis szczegÃ³Åowy

G³ówny plik programu do obliczania liczby Pi za pomoc¹ numerycznej ca³ki.

Program oferuje dwa tryby dzia³ania:

- Uruchomienie testów automatycznych dla ró¿nych liczby podzia³ów i w¹tków.
- Rêczne obliczanie przybli¿onej wartoci liczby Pi dla podanej liczby podzia³ów i w¹tków.

Wyniki testów s¹ zapisywane w pliku CSV, a u¿ytkownik mo¿e ledziæ czas obliczeñ oraz przybli¿on¹ wartoæ liczby Pi.

4.5.1.1 Wykorzystywane biblioteki:

- `<iostream>`: Standardowa biblioteka wejcia/wyjcia, u¿ywana do komunikacji z u¿ytkownikiem (wypisywanie wyników i pobieranie danych).
- `<iomanip>`: Biblioteka do manipulacji formatem wyjcia, u¿ywana w celu ustawienia precyzyjnoci wywietlanej wartoci liczby Pi.
- `<fstream>`: Biblioteka umo¿liwiaj¹ca zapis wyników do pliku CSV.
- `"integral.h"`: Nag³ówek klasy `PiCalculator` zawieraj¹cej metody obliczania liczby Pi.

Definicja w pliku main.cpp.

4.5.2 Dokumentacja funkcji

4.5.2.1 main()

```
int main ()
```

Funkcja g³ówna programu, która umo¿liwia wybór trybu dzia³ania programu.

Program dzia³a w dwóch trybach:

- Uruchomienie testów automatycznych.
- Rêczne obliczanie liczby Pi dla podanych przez u¿ytkownika parametrów.

U¿ytkownik wybiera tryb dzia³ania, a nastêpnie program realizuje odpowiedni¹ funkcjonalnoæ. Wyniki obliczeñ s¹ wywietlane na ekranie i zapisywane w pliku CSV w przypadku testów automatycznych.

Zwraca

0 jeli program zakoñczy³ siê poprawnie.

< Uruchomienie funkcji przeprowadzaj¹cej testy automatyczne.

< Zmienna do przechowywania liczby przedzia³ów.

< Zmienna do przechowywania liczby w¹tków.

Definicja w linii 86 pliku main.cpp.

4.5.2.2 runTests()

```
void runTests ()
```

< Nag³ówek klasy PiCalculator, zawieraj¹cy deklaracje metod obliczaj¹cych liczbê Pi.

< Biblioteka do obs³ugi wejcia/wyjcia, w tym wypisywania na ekranie. Biblioteka do manipulacji formatem wyjcia (np. ustawianie precyzji liczby zmiennoprzecinkowej). Biblioteka do obs³ugi plików, w tym zapisywania wyników do pliku CSV.

Funkcja do uruchamiania testów automatycznych obliczeñ liczby Pi dla ró¿nych liczby podzia³ów i w¹tków.

Funkcja ta przeprowadza testy automatyczne, obliczaj¹c wartoæ liczby Pi dla ró¿nych liczby podzia³ów i w¹tków. Wyniki s¹ zapisywane w pliku CSV oraz wypisywane na ekranie.

Definicja w linii 31 pliku main.cpp.

4.6 main.cpp

IdÅº do dokumentacji tego pliku.

```
00001
00019 #include <iostream>
00020 #include <iomanip>
00021 #include <fstream>
00022 #include "integral.h"
00023
00031 void runTests() {
00032     // Lista liczby podzia³ów (w ró¿nych skalach) u¿ywanych w testach.
00033     const std::vector<long long> intervalsList = { 100000000, 1000000000, 3000000000 };
00034
00035     // Maksymalna liczba w¹tków, które bêd¹ testowane w ramach testów automatycznych.
00036     const int maxThreads = 50;
00037
00038     // Otwarcie pliku wynikowego, do którego bêd¹ zapisywane rezultaty testów w formacie CSV.
00039     std::ofstream resultsFile("results.csv");
00040
00041     // Zapisanie nag³ówka do pliku CSV (nag³ówki dla ka¿dego typu danych, który bêdzie zapisywany).
00042     resultsFile « "Intervals,Threads,Time (s),Pi Approximation\n";
00043
00044     // Pêtla przechodz¹ca przez ka¿dy przypadek liczby podzia³ów (z listy intervalsList).
00045     for (auto intervals : intervalsList) {
00046         // Pêtla przechodz¹ca przez liczbê w¹tków, które bêd¹ testowane.
00047         for (int threads = 1; threads <= maxThreads; ++threads) {
00048             // Tworzenie obiektu kalkulatora, który obliczy wartoæ Pi dla aktualnej liczby podzia³ów i
    w¹tków.
00049             PiCalculator calculator(intervals, threads);
00050
00051             // Obliczanie przybli¿onej wartoci liczby Pi oraz czasu potrzebnego na obliczenie.
00052             double pi = calculator.calculate();
00053             double time = calculator.getExecutionTime();
00054
00055             // Zapisanie wyników obliczeñ do pliku CSV (liczba podzia³ów, liczba w¹tków, czas
    wykonania, przybli¿ona wartoæ Pi).
00056             resultsFile « intervals « "," « threads « "," « time « "," « pi « "\n";
00057
00058             // Wypisanie wyników na ekranie dla u¿ytkownika.
00059             std::cout « "Intervals: " « intervals
00060                 « ", Threads: " « threads
00061                 « ", Time: " « time
00062                 « " s, Pi: " « pi « "\n";
00063         }
00064     }
00065
00066     // Zamkniêcie pliku po zakoñczeniu zapisywania wyników.
00067     resultsFile.close();
00068
00069     // Wypisanie komunikatu, ¿e testy zosta³y zakoñczone, a wyniki zosta³y zapisane w pliku.
00070     std::cout « "Testy zakoñczone. Wyniki zapisano w pliku 'results.csv'.\n";
00071 }
00072
00086 int main() {
00087     // Zmienna przechowuj¹ca wybór u¿ytkownika.
00088     int option;
```

```
00089
00090     // Wywietlenie komunikatu z prob¹ o wybór jednej z dwóch opcji.
00091     std::cout « "Wybierz opcje:\n";
00092     std::cout « "1 - Uruchom testy automatyczne i zapisz wyniki\n";
00093     std::cout « "2 - Oblicz Pi dla podanej liczby podzia³ów i w¹tków\n";
00094
00095     // Wczytanie wyboru u¿ytkownika.
00096     std::cin » option;
00097
00098     // Jeli u¿ytkownik wybierze opcjê 1, uruchomimy testy automatyczne.
00099     if (option == 1) {
00100         runTests();
00101     }
00102     // Jeli u¿ytkownik wybierze opcjê 2, zapytamy go o parametry i obliczymy liczbê Pi.
00103     else if (option == 2) {
00104         long long intervals;
00105         int threads;
00106
00107         // Poproszenie u¿ytkownika o podanie liczby przedzia³ów do obliczeñ.
00108         std::cout « "Podaj liczbe podzialow przedzialu (np. 100000000): ";
00109         std::cin » intervals;
00110
00111         // Poproszenie u¿ytkownika o podanie liczby w¹tków do obliczeñ.
00112         std::cout « "Podaj liczbe watkow (np. 4): ";
00113         std::cin » threads;
00114
00115         // Utworzenie obiektu kalkulatora z odpowiednimi parametrami.
00116         PiCalculator calculator(intervals, threads);
00117
00118         // Obliczenie liczby Pi i czasu obliczeñ.
00119         double pi = calculator.calculate();
00120         double executionTime = calculator.getExecutionTime();
00121
00122         // Wywietlenie wyników na ekranie z ustawion¹ precyzj¹ na 15 miejsc po przecinku.
00123         std::cout « std::fixed « std::setprecision(15);
00124         std::cout « "Przyblizona wartosc liczby Pi: " « pi « "\n";
00125         std::cout « "Czas obliczen: " « executionTime « " sekund\n";
00126     }
00127     // Jeli u¿ytkownik poda niepoprawn¹ opcjê, wywietlamy komunikat o b³êdzie.
00128     else {
00129         std::cout « "Niepoprawna opcja. Koniec programu.\n";
00130     }
00131
00132     // Zwrócenie wartoci 0, co oznacza zakoñczenie programu bez b³êdów.
00133     return 0;
00134 }
```

4.7 Dokumentacja pliku Intergral/results.csv

4.8 results.csv

Idź do dokumentacji tego pliku.

```
00001 Intervals,Threads,Time (s),Pi Approximation
00002 100000000,1,1.61505,3.14159
00003 100000000,2,0.832012,3.14159
00004 100000000,3,0.570136,3.14159
00005 100000000,4,0.473337,3.14159
00006 100000000,5,0.584576,3.14159
00007 100000000,6,0.501683,3.14159
00008 100000000,7,0.49434,3.14159
00009 100000000,8,0.470761,3.14159
00010 100000000,9,0.482449,3.14159
00011 100000000,10,0.502949,3.14159
00012 100000000,11,0.483042,3.14159
00013 100000000,12,0.549255,3.14159
00014 100000000,13,0.539976,3.14159
00015 100000000,14,0.607262,3.14159
00016 100000000,15,0.566931,3.14159
00017 100000000,16,0.49804,3.14159
00018 100000000,17,0.535247,3.14159
00019 100000000,18,0.524682,3.14159
00020 100000000,19,0.537596,3.14159
00021 100000000,20,0.558083,3.14159
00022 100000000,21,0.512799,3.14159
00023 100000000,22,0.534402,3.14159
00024 100000000,23,0.533752,3.14159
00025 100000000,24,0.541241,3.14159
00026 100000000,25,0.517592,3.14159
```

```
00027 100000000,26,0.521125,3.14159
00028 100000000,27,0.55971,3.14159
00029 100000000,28,0.572369,3.14159
00030 100000000,29,0.543633,3.14159
00031 100000000,30,0.546074,3.14159
00032 100000000,31,0.546711,3.14159
00033 100000000,32,0.53894,3.14159
00034 100000000,33,0.595751,3.14159
00035 100000000,34,0.560477,3.14159
00036 100000000,35,0.669299,3.14159
00037 100000000,36,0.574876,3.14159
00038 100000000,37,0.697501,3.14159
00039 100000000,38,0.565716,3.14159
00040 100000000,39,0.628777,3.14159
00041 100000000,40,0.554024,3.14159
00042 100000000,41,0.563855,3.14159
00043 100000000,42,0.552957,3.14159
00044 100000000,43,0.544211,3.14159
00045 100000000,44,0.577994,3.14159
00046 100000000,45,0.585933,3.14159
00047 100000000,46,0.595658,3.14159
00048 100000000,47,0.605307,3.14159
00049 100000000,48,0.584764,3.14159
00050 100000000,49,0.564672,3.14159
00051 100000000,50,0.621125,3.14159
00052 1000000000,1,15.1967,3.14159
00053 1000000000,2,8.10937,3.14159
00054 1000000000,3,5.5693,3.14159
00055 1000000000,4,5.14724,3.14159
00056 1000000000,5,4.78412,3.14159
00057 1000000000,6,4.81128,3.14159
00058 1000000000,7,4.62157,3.14159
00059 1000000000,8,5.4022,3.14159
00060 1000000000,9,4.51284,3.14159
00061 1000000000,10,4.53364,3.14159
00062 1000000000,11,4.49475,3.14159
00063 1000000000,12,4.59792,3.14159
00064 1000000000,13,4.70243,3.14159
00065 1000000000,14,4.5389,3.14159
00066 1000000000,15,4.44227,3.14159
00067 1000000000,16,4.63589,3.14159
00068 1000000000,17,4.57276,3.14159
00069 1000000000,18,4.43908,3.14159
00070 1000000000,19,4.61509,3.14159
00071 1000000000,20,4.39633,3.14159
00072 1000000000,21,4.59974,3.14159
00073 1000000000,22,4.59723,3.14159
00074 1000000000,23,4.39454,3.14159
00075 1000000000,24,4.68441,3.14159
00076 1000000000,25,4.23639,3.14159
00077 1000000000,26,4.51794,3.14159
00078 1000000000,27,4.54971,3.14159
00079 1000000000,28,4.33581,3.14159
00080 1000000000,29,4.22068,3.14159
00081 1000000000,30,4.57626,3.14159
00082 1000000000,31,4.08598,3.14159
00083 1000000000,32,4.41873,3.14159
00084 1000000000,33,4.43683,3.14159
00085 1000000000,34,4.22105,3.14159
00086 1000000000,35,4.44081,3.14159
00087 1000000000,36,4.35321,3.14159
00088 1000000000,37,4.58384,3.14159
00089 1000000000,38,5.03191,3.14159
00090 1000000000,39,5.206,3.14159
00091 1000000000,40,4.8026,3.14159
00092 1000000000,41,4.53192,3.14159
00093 1000000000,42,4.50739,3.14159
00094 1000000000,43,4.57059,3.14159
00095 1000000000,44,4.74928,3.14159
00096 1000000000,45,4.6719,3.14159
00097 1000000000,46,4.82827,3.14159
00098 1000000000,47,4.55738,3.14159
00099 1000000000,48,5.05816,3.14159
00100 1000000000,49,5.13644,3.14159
00101 1000000000,50,4.78547,3.14159
00102 3000000000,1,45.4363,3.14159
00103 3000000000,2,22.9197,3.14159
00104 3000000000,3,16.1554,3.14159
00105 3000000000,4,13.4999,3.14159
00106 3000000000,5,13.3157,3.14159
00107 3000000000,6,13.1905,3.14159
00108 3000000000,7,13.072,3.14159
00109 3000000000,8,12.8897,3.14159
00110 3000000000,9,12.8433,3.14159
00111 3000000000,10,12.7389,3.14159
00112 3000000000,11,12.5553,3.14159
00113 3000000000,12,12.6775,3.14159
```

```
00114 3000000000,13,12.4821,3.14159
00115 3000000000,14,12.3901,3.14159
00116 3000000000,15,12.6226,3.14159
00117 3000000000,16,12.3726,3.14159
00118 3000000000,17,12.2746,3.14159
00119 3000000000,18,12.5416,3.14159
00120 3000000000,19,12.4255,3.14159
00121 3000000000,20,12.2673,3.14159
00122 3000000000,21,12.5029,3.14159
00123 3000000000,22,12.4945,3.14159
00124 3000000000,23,13.7876,3.14159
00125 3000000000,24,16.2504,3.14159
00126 3000000000,25,13.257,3.14159
00127 3000000000,26,13.6385,3.14159
00128 3000000000,27,14.8137,3.14159
00129 3000000000,28,15.7194,3.14159
00130 3000000000,29,17.449,3.14159
00131 3000000000,30,14.1058,3.14159
00132 3000000000,31,12.9856,3.14159
00133 3000000000,32,13.8403,3.14159
00134 3000000000,33,14.5384,3.14159
00135 3000000000,34,13.4628,3.14159
00136 3000000000,35,13.5211,3.14159
00137 3000000000,36,13.102,3.14159
00138 3000000000,37,12.9221,3.14159
00139 3000000000,38,12.8209,3.14159
00140 3000000000,39,13.1884,3.14159
00141 3000000000,40,14.8233,3.14159
00142 3000000000,41,13.8895,3.14159
00143 3000000000,42,13.7052,3.14159
00144 3000000000,43,14.3186,3.14159
00145 3000000000,44,13.8766,3.14159
00146 3000000000,45,14.4345,3.14159
00147 3000000000,46,14.0099,3.14159
00148 3000000000,47,13.0898,3.14159
00149 3000000000,48,13.8203,3.14159
00150 3000000000,49,13.3146,3.14159
00151 3000000000,50,15.3979,3.14159
```

Skorowidz