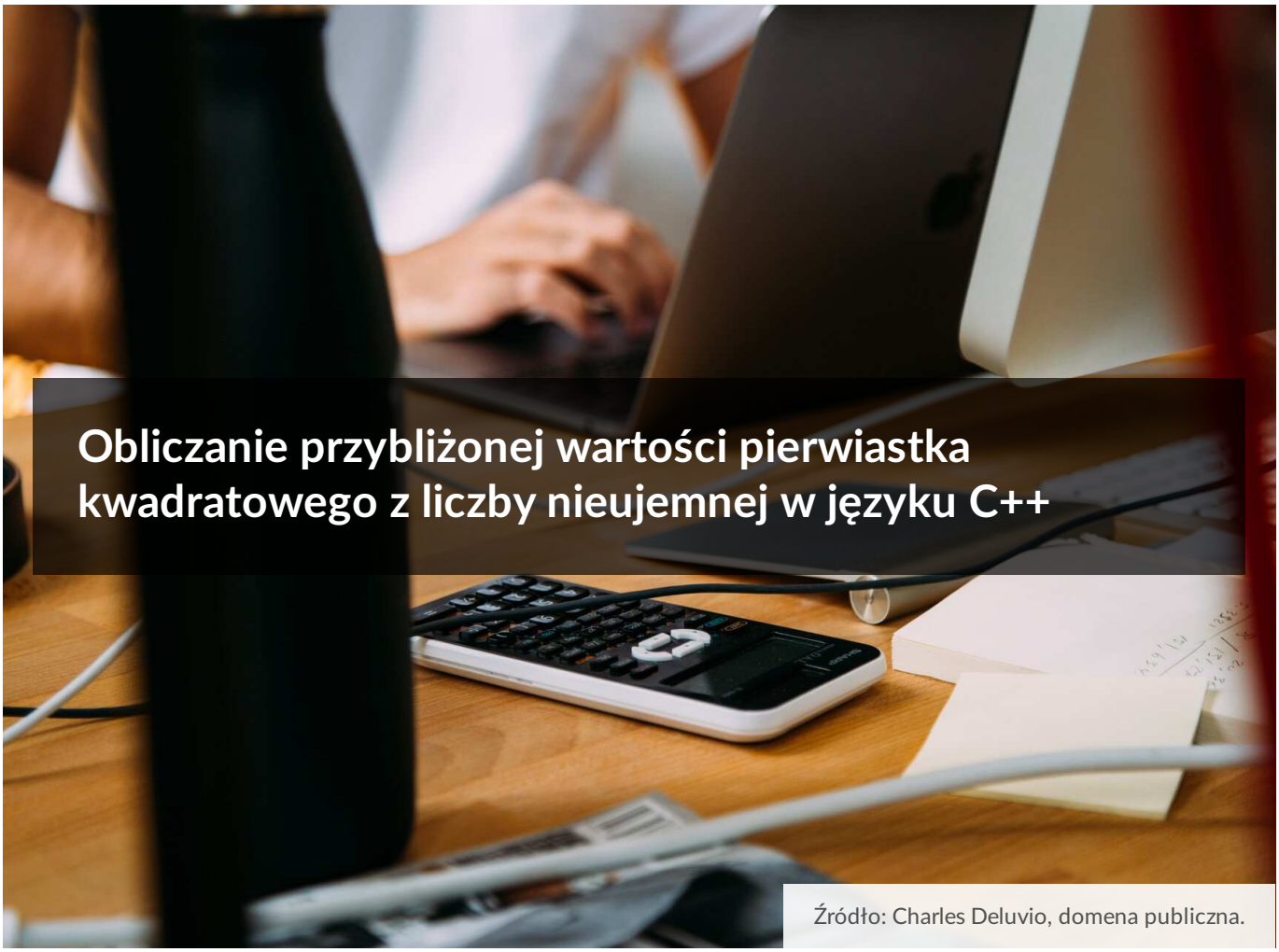
Obliczanie przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej w języku C++



Źródło: Charles Deluvio, domena publiczna.

Z e-materiału Obliczanie przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej znasz algorytm obliczania pierwiastka kwadratowego. Metoda ta pozwala wyznaczyć przybliżoną wartość pierwiastka kwadratowego z nieujemnej liczby rzeczywistej. W tym e-materiale zaimplementujemy ten algorytm w języku C++.

Implementację w innych językach programowania znajdziesz w e-materiałach:

- Obliczanie przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej w języku Java,
- Obliczanie przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej w języku Python.

Więcej zadań? Przejdź do e-materiału Obliczanie przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej – zadania maturalne.

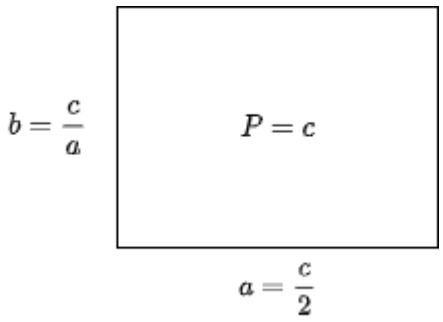
Twoje cele

- Utrwalisz założenia metody Newtona-Raphsona i sprawdzisz jej działanie w praktyce.
- Przeanalizujesz implementację algorytmu Newtona-Raphsona w języku C++.
- Zapiszesz programy wymagające obliczenia przybliżonej wartości pierwiastka kwadratowego w języku C++.

Przeczytaj

Implementacja metody Newtona-Raphsona w języku C++

W metodzie Newtona-Raphsona korzystamy z faktu, że pierwiastkowana liczba jest równa polu kwadratu, którego boki są takiej długości jak wartość wyniku pierwiastkowania. Będziemy więc tworzyć kolejne prostokąty, coraz bardziej zbliżone do poszukiwanej figury. Załóżmy, że pierwiastkujemy pewną liczbę c, a pierwszy wielokąt wyglądać będzie następująco:



Źródło: Contentplus.pl Sp. z o.o., licencja: CC BY-SA 3.0.

Na podstawie długości boków prostokąta wyliczać będziemy kolejne poszukiwane wartości. Zapiszmy więc funkcję odpowiedzialną za proces wyszukiwania docelowej długości boku a kwadratu. Pamiętajmy, że jedną z niezbędnych zmiennych będzie również `precyzja`, której rolą zajmiemy się w następnym kroku. Na razie w funkcji znajdą się jedynie założenia początkowe, dotyczące długości boków pierwszego prostokąta.

```
double pierwiastkowanie(double pierwiastkowany, double precyzja){
    double a = pierwiastkowany / 2,
    b = pierwiastkowany / a;
}
```

Tworzymy pętlę odpowiedzialną za wyliczanie kolejnych długości boków wielokątów. Operację tę przeprowadzamy do momentu, gdy wartość bezwzględna różnicy pomiędzy długościami obu boków będzie mniejsza niż zadana precyzja. Im mniejszą wartość zmiennej `precyzja` ustalimy, tym wynik będzie dokładniejszy. Długości boków prostokątów wyliczamy w następujący sposób:

$$a = \frac{(a + b)}{2}$$

$$b = \frac{c}{a}$$

Mając już niezbędne informacje, możemy przejść do dokończenia funkcji. Powtarzanie przedstawionych operacji, dopóki nie spełnimy wspomnianego warunku dotyczącego precyzji i różnicy boków, oznacza konieczność zastosowania pętli. Dodatkowo w warunku zastosujemy wartość bezwzględną. Użyjemy funkcji `fabs` z biblioteki cmath.

```
#include <iostream>
#include <cmath>
using namespace std;

double pierwiastkowanie(double pierwiastkowany, double precyzja){
    double a = pierwiastkowany / 2,
    b = pierwiastkowany / a;

    while(fabs(a - b) > precyzja){
        a = (a + b) / 2;
        b = pierwiastkowany / a;
    }
    return a;
}
```

Teraz wystarczy już tylko pobrać od użytkownika pierwiastkowaną liczbę oraz żądaną precyzję. Po wywołaniu funkcji otrzymamy wypisany przybliżony wynik pierwiastkowania kwadratowego.

```
int main(){
    double c, p;
    cout << "Wprowadz liczbe do pierwiastkowania" << endl;
    cin >> c;
    cout << "Wprowadz precyzje" << endl;
    cin >> p;
    cout << "Wynik pierwiastkowania" << endl;
    cout << pierwiastkowanie(c, p);
    return 0;
}
```

Cały program prezentuje się następująco:

```cpp
#include <iostream>
#include <cmath>
using namespace std;

double pierwiastkowanie(double pierwiastkowany, double precyzja){
    double a=pierwiastkowany/2,
           b=pierwiastkowany/a;

    while(fabs(a-b)>precyzja){
        a=(a+b)/2;
        b=pierwiastkowany/a;
    }
    return a;
}

int main(){
    double c,p;
    cout << "Wprowadz liczbe do pierwiastkowania" << endl;
    cin >> c;
    cout << "Wprowadz precyzje" << endl;
    cin >> p;
    cout << "Wynik pierwiastkowania" << endl;
    cout << pierwiastkowanie(c,p);
    return 0;
}
```

W ten sposób zapisaliśmy algorytm Newtona-Raphsona obliczania przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej w języku C++.

Słownik

cmath

używany w języku C++ nagłówek biblioteki przechowującej funkcje realizujące podstawowe operacje matematyczne, takie jak np. funkcje trygonometryczne, hiperboliczne czy zaokrąglenia; odpowiednikiem `cmath` w języku C jest nagłówek `math.h` (po dodaniu `#include <cmath>` funkcje stają się dostępne w programie); przykładem takiej funkcji jest funkcja `fabs(x)`, która zwraca wartość bezwzględną liczby rzeczywistej `x`

metoda Newtona-Raphsona

(inaczej: metoda Newtona lub metoda stycznych) algorytm wyznaczania wartości pierwiastka funkcji oparty na wyliczaniu kolejnych przybliżeń coraz bliższych rzeczywistej wartości pierwiastka; metoda ta może być wykorzystywana między innymi do omawianego wyznaczania wartości pierwiastka kwadratowego

parametr

element składni w określonym języku programowania umożliwiający komunikację pomiędzy podprogramem wywołanym a programem wywołującym; parametry określa się wraz z deklaracją określonego podprogramu w jego nagłówku

Symulacja interaktywna

Polecenie 1

Przeanalizuj przedstawiony w symulacji interaktywnej proces obliczania pierwiastka kwadratowego metodą Newtona-Raphsona. Zwróć uwagę na wizualizację prostokątów o odpowiednich wartościach boków.

Polecenie 2

Polecenie 3

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Napisz program, który za pomocą metody Newtona-Raphsona obliczy wartość pierwiastka kwadratowego liczby c przy ustalonej precyzji p. Oblicz, ile iteracji wykona algorytm, a następnie podaj wynik pierwiastkowania z dokładnością o wartości $0,01$ (bez zaokrąglania). Wypisz wyniki oddzielone pojedynczym znakiem odstępu. Swój program przetestuj dla następujących danych:

- $c = 35$
- $p = 0,0001$

Specyfikacja problemu:

Dane:

- c – liczba podpierwiastkowa; liczba rzeczywista
- p – precyzja, z jaką algorytm powinien wyznaczyć wartość pierwiastka; liczba rzeczywista

Wynik:

- `iteracje` – liczba iteracji wykonana przez algorytm; liczba naturalna
- `wynik` – wynik pierwiastkowania z dokładnością do $0,01$ bez zaokrąglania; liczba rzeczywista

Przykładowe wyjście:

```
5 5.91
```

Twoje zadania

1. Program ma obliczyć wartość pierwiastka kwadratowego liczby c przy ustalonej precyzji p i wypisać liczbę wykonanych iteracji oraz wynik pierwiastkowania z dokładnością do 0,01 bez zaokrąglania.

Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Obliczanie przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej w języku C++

Grupa docelowa:

Liceum ogólnokształcące i technikum, liceum ogólnokształcące, technikum, zakres rozszerzony

Podstawa programowa:

Cele kształcenia – wymagania ogólne

I. Rozumienie, analizowanie i rozwiązywanie problemów na bazie logicznego i abstrakcyjnego myślenia, myślenia algorytmicznego i sposobów reprezentowania informacji.

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera oraz innych urządzeń cyfrowych: układanie i programowanie algorytmów, organizowanie, wyszukiwanie i udostępnianie informacji, posługiwanie się aplikacjami komputerowymi.

Treści nauczania – wymagania szczegółowe

I. Rozumienie, analizowanie i rozwiązywanie problemów.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

2) do realizacji rozwiązania problemu dobiera odpowiednią metodę lub technikę algorytmiczną i struktury danych;

3) objaśnia dobrany algorytm, uzasadnia poprawność rozwiązania na wybranych przykładach danych i ocenia jego efektywność;

II. Programowanie i rozwiązywanie problemów z wykorzystaniem komputera i innych urządzeń cyfrowych.

Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

3) sprawnie posługuje się zintegrowanym środowiskiem programistycznym przy pisaniu, uruchamianiu i testowaniu programów;

I + II. Zakres rozszerzony. Uczeń spełnia wymagania określone dla zakresu podstawowego, a ponadto:

1) zapisuje za pomocą listy kroków, schematu blokowego lub pseudokodu, i implementuje w wybranym języku programowania, algorytmy poznane na wcześniejszych etapach oraz algorytmy:

g) obliczania przybliżonej wartości pierwiastka kwadratowego,

Kształtowane kompetencje kluczowe:

- kompetencje cyfrowe;
- kompetencje osobiste, społeczne i w zakresie umiejętności uczenia się;
- kompetencje matematyczne oraz kompetencje w zakresie nauk przyrodniczych, technologii i inżynierii.

Cele operacyjne (językiem ucznia):

- Utrwalisz założenia metody Newtona-Raphsona i sprawdzisz jej działanie w praktyce.
- Przeanalizujesz implementację algorytmu Newtona-Raphsona w języku C++.
- Zapiszesz programy wymagające obliczenia przybliżonej wartości pierwiastka kwadratowego w języku C++.

Strategie nauczania:

- konstruktywizm;
- konektywizm.

Metody i techniki nauczania:

- dyskusja;
- rozmowa nauczająca z wykorzystaniem multimedium i ćwiczeń interaktywnych;
- ćwiczenia praktyczne.

Formy pracy:

- praca indywidualna;
- praca w parach;
- praca w grupach;
- praca całego zespołu klasowego.

Środki dydaktyczne:

- komputery z głośnikami, słuchawkami i dostępem do internetu;

- zasoby multimedialne zawarte w e-materiale;
- tablica interaktywna/tablica, pisak/kreda;
- oprogramowanie dla języka C++, w tym kompilator GCC/G++ 4.5 (lub nowszej wersji) i Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Przebieg lekcji

Przed lekcją:

1. **Przygotowanie do zajęć.** Nauczyciel loguje się na platformie i udostępnia e-materiał: „Obliczanie przybliżonej wartości pierwiastka kwadratowego z liczby nieujemnej w języku C++". Uczniowie zapoznają się z treściami w sekcji „Przeczytaj" w kontekście programowania.

Faza wstępna:

1. Nauczyciel prosi wybraną osobę o odczytanie tematu lekcji, a następnie określa cele i kryteria sukcesu.
2. Prowadzący prosi uczniów, aby zgłaszali swoje propozycje pytań do tematu. Jedna osoba może zapisywać je na tablicy. Gdy uczniowie wyczerpią swoje pomysły, a pozostały jakieś ważne kwestie do poruszenia, nauczyciel je dopowiada.

Faza realizacyjna:

1. **Praca z tekstem.** Metodą burzy mózgów uczniowie referują najważniejsze informacje, jakie znaleźli w sekcji „Przeczytaj", przygotowując się do lekcji. W razie potrzeby nauczyciel wyjaśnia niezrozumiałe kwestie.
2. **Praca z multimedium.** Nauczyciel wyświetla zawartość sekcji „Symulacja interaktywna". Uczniowie zapoznają się z symulacją i wykonują polecenia 1–2.
3. **Ćwiczenie umiejętności.** Uczniowie wykonują indywidualnie ćwiczenie nr 1 z sekcji „Sprawdź się", a następnie porównują swoje odpowiedzi z kolegą lub koleżanką.

Faza podsumowująca:

1. Nauczyciel wyświetla na tablicy temat lekcji i cele zawarte w sekcji „Wprowadzenie". W kontekście ich realizacji podsumowuje przebieg zajęć, a także wskazuje mocne i słabe strony pracy uczniów.
2. Na koniec zajęć z programowania w C++ nauczyciel prosi uczniów o rozwinięcie zdania: „Na dzisiejszych zajęciach nauczyłam/łem się jak…".

Praca domowa:

1. Uczniowie wykonują Polecenie 3 z sekcji „Symulacja interaktywna".

Materiały pomocnicze:

- Oficjalna dokumentacja techniczna dla języka C++.
- Oficjalna dokumentacja techniczna dla kompilatora GCC/G++ 4.5 (lub nowszej wersji).
- Oficjalna dokumentacja techniczna dla oprogramowania Code::Blocks 16.01 (lub nowszej wersji), Orwell Dev-C++ 5.11 (lub nowszej wersji) lub Microsoft Visual Studio.

Wskazówki metodyczne:

- Treści w sekcji „Przeczytaj" można wykorzystać jako podsumowanie i utrwalenie wiedzy uczniów.