Konstrukcje fraktalne w języku C++

Źródło: Nick Cooper, domena publiczna.

Wiemy już, czym są fraktale oraz konstrukcje fraktalne. Poznaliśmy paproć Barnsleya oraz smoka Heighwaya, czyli obiekty samopodobne, które można generować za pomocą algorytmów IFS, tzn. sytemu funkcji iterowanych.

W tym e-materiale zajmiemy się stworzeniem fraktali w języku C++.

Ciekawi cię, jak wyglądają implementacje w innych językach programowania? Możesz się z nimi zapoznać w dwóch pozostałych e-materiałach z tej serii:

- Konstrukcje fraktalne w języku Java,
- Konstrukcje fraktalne w języku Python.

Więcej zadań? Sięgnij do Konstrukcje fraktalne – zadania maturalne.

> **Twoje cele**
>
> - Przeanalizujesz sposób wyznaczania liczb pseudolosowych w celu utworzenia obiektów samopodobnych.
> - Zbudujesz graficzny program wyświetlający fraktale na ekranie.
> - Utworzysz paproć Barnsleya oraz smoka Heighwaya, korzystając z przekształcenia afinicznego oraz generatora liczb pseudolosowych.

Przeczytaj

Problem 1

Napisz program, którego zadaniem będzie wygenerowanie smoka Heighwaya w trybie tekstowym.

Specyfikacja problemu:

Dane:

- `tablica` – tablica napisów
- `H` – łańcuch znaków; pozioma kreska
- `W` – łańcuch znaków; pionowa kreska
- `L1` – łańcuch znaków; skręt w prawo od dołu
- `L2` – łańcuch znaków; skręt w lewo od dołu
- `L3` – łańcuch znaków; skręt w lewo od góry
- `L4` – łańcuch znaków; skręt w prawo od góry
- `x`, `y` – liczby naturalne
- `kierunek` – łańcuch znaków

Wynik:

Program wyświetla wygenerowany fraktal (smoka Heighwaya).

Polecenie 1

Porównaj swoje rozwiązanie z przedstawionym w filmie.

Film dostępny pod adresem /preview/resource/Rqp1c9HFVopcG

Film nawiązujący do treści materiału: Konstrukcje fraktalne jako stosowanie rekurencji 2.

Kod programu zaprezentowanego w filmie:

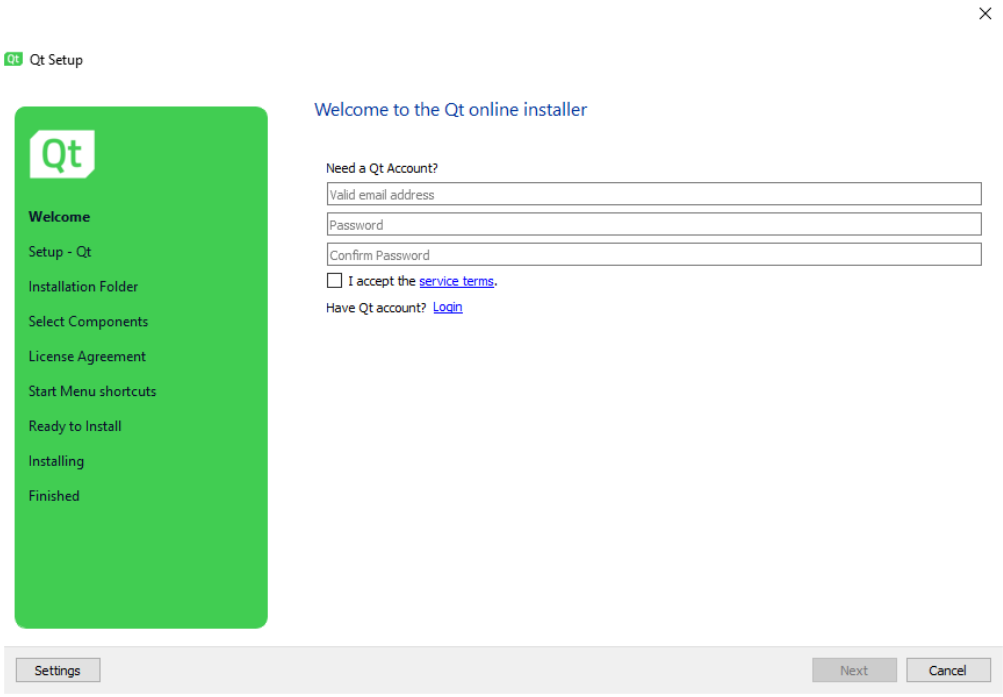
Plik o rozmiarze 2.51 KB w języku polskim

Biblioteka graficzna Qt5 – konfiguracja środowiska

W zaprezentowanym filmie wygenerowaliśmy smoka Heighwaya w trybie tekstowym. W kolejnym kroku, aby dokonać wizualizacji wyników obliczeń naszego programu, posłużymy się biblioteką graficzną Qt5. Zaletą tego rozwiązania jest prostota konfiguracji.

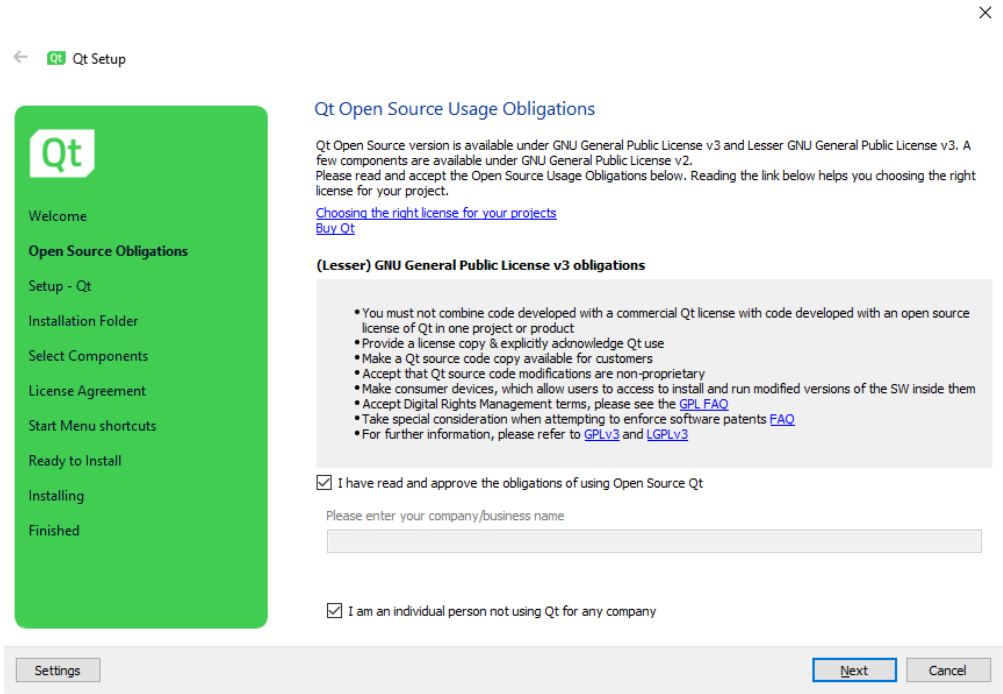
W celu uruchomienia środowiska należy pobrać instalator z oficjalnej strony Qt. Znajdziemy ją, wpisując do dowolnej przeglądarki internetowej frazę „qt5” – jednym z pierwszych wyników, które się wyświetlą, będzie właśnie szukana strona internetowa.

Po uruchomieniu instalatora zostaniemy poproszeni o zalogowanie się. Jeśli nie mamy konta, możemy je utworzyć bezpośrednio z poziomu instalatora. W tym celu należy wybierać opcję `Sign up` (zarejestruj się), zarejestrować konto, a następnie potwierdzić je za pomocą adresu e-mail. Po poprawnym utworzeniu konta wystarczy zalogować się w instalatorze i można kontynuować instalację.



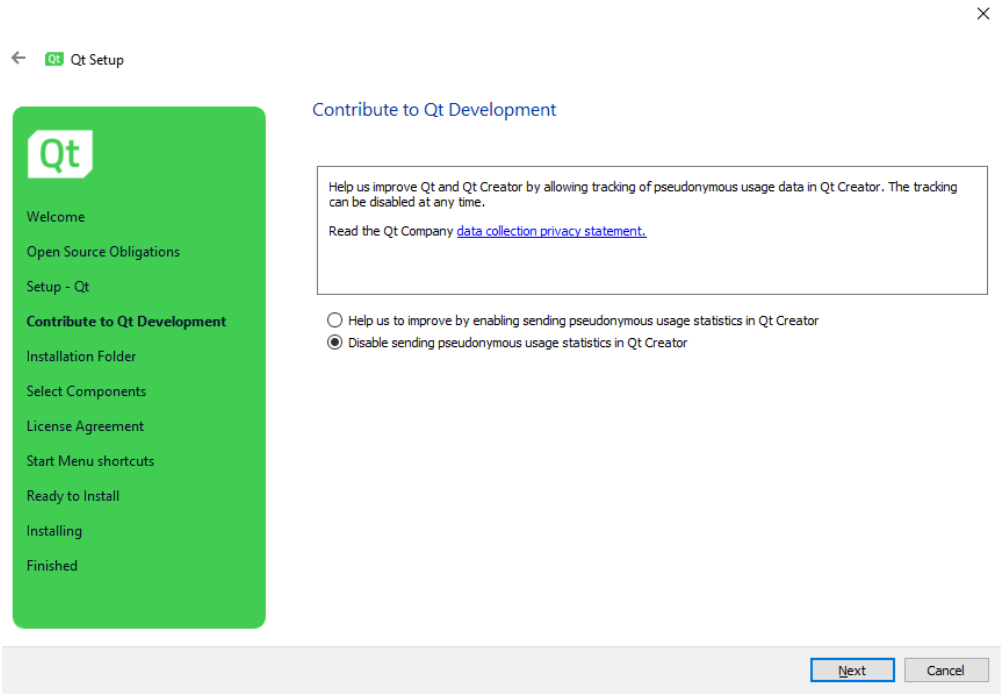
Źródło: Contentplus.pl Sp. z o.o., licencja: CC BY-SA 3.0.

W kolejnym etapie należy zaakceptować zobowiązania licencyjne, a także podać nazwę firmy/imię biznesowe lub wybrać opcję `I am individual person not using Qt for any company`, która zapewne będzie dotyczyć znacznej większości użytkowników. Kontynuujemy instalację, wybierając opcję `Next`.



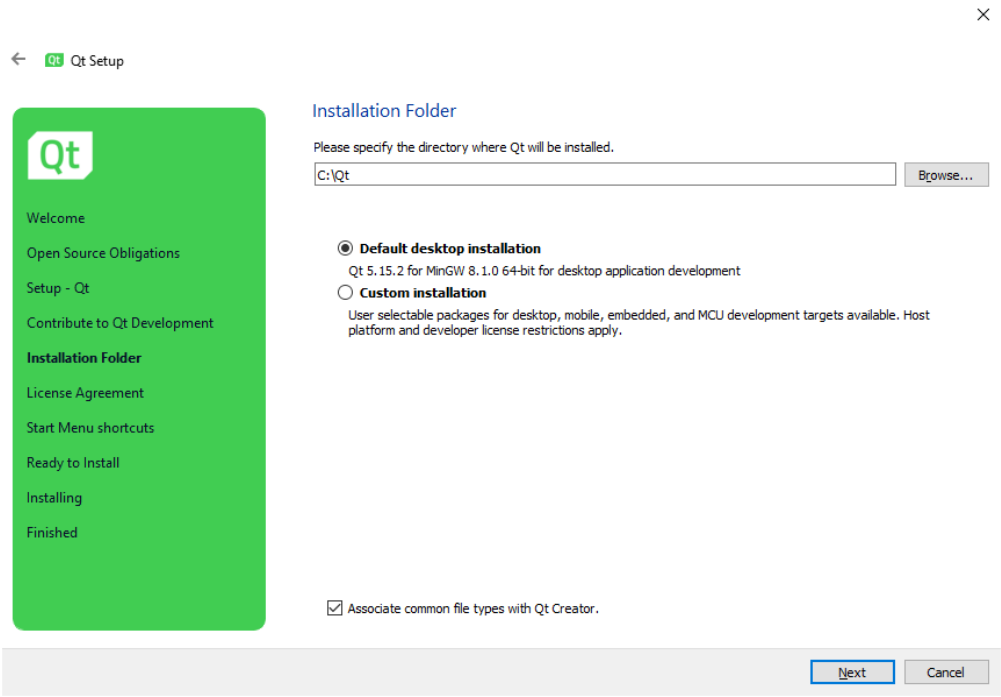
Źródło: Contentplus.pl Sp. z o.o., licencja: CC BY-SA 3.0.

Następnie możemy zdecydować, czy chcemy przesłać dodatkowe dane w celu ulepszenia przyszłych wersji Qt5. Zaznaczamy wybraną opcję i kontynuujemy instalację.



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

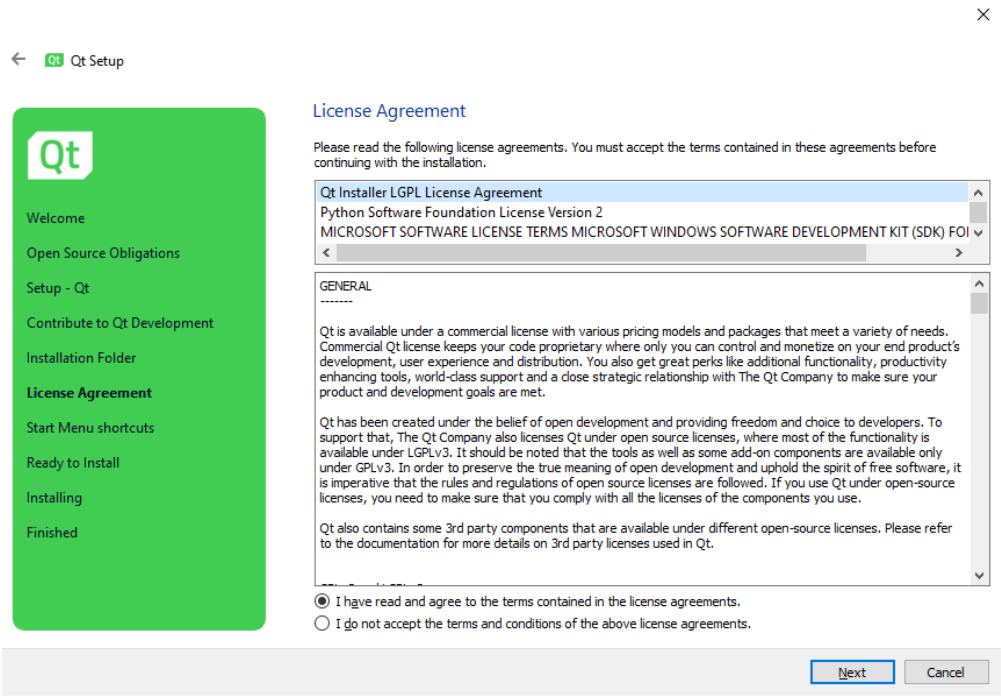
W kolejnym kroku podajemy lokalizację, w której umieszczone zostaną pliki programu. W przypadku początkujących użytkowników zaleca się wybranie domyślnej opcji `Default desktop installation`, aby wraz ze środowiskiem został zainstalowany kompilator oraz edytor graficzny. Następnie wybieramy opcję `Next`.



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

Teraz należy zapoznać się z postanowieniami licencyjnymi środowiska oraz produktów wraz z nim dołączonych i zaakceptować je, a następnie wybrać `Next`. W kolejnym etapie

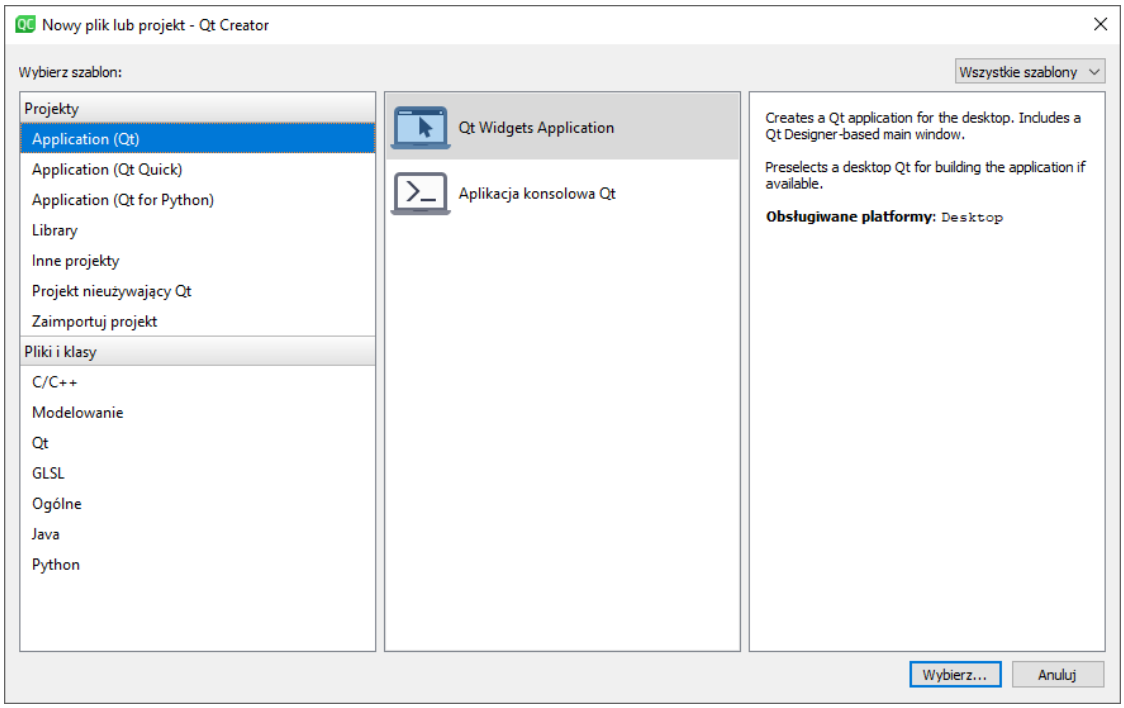
można ewentualnie zmienić nazwę grupy, która zostanie utworzona w menu `Start`, po czym ponownie wybrać `Next`.



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

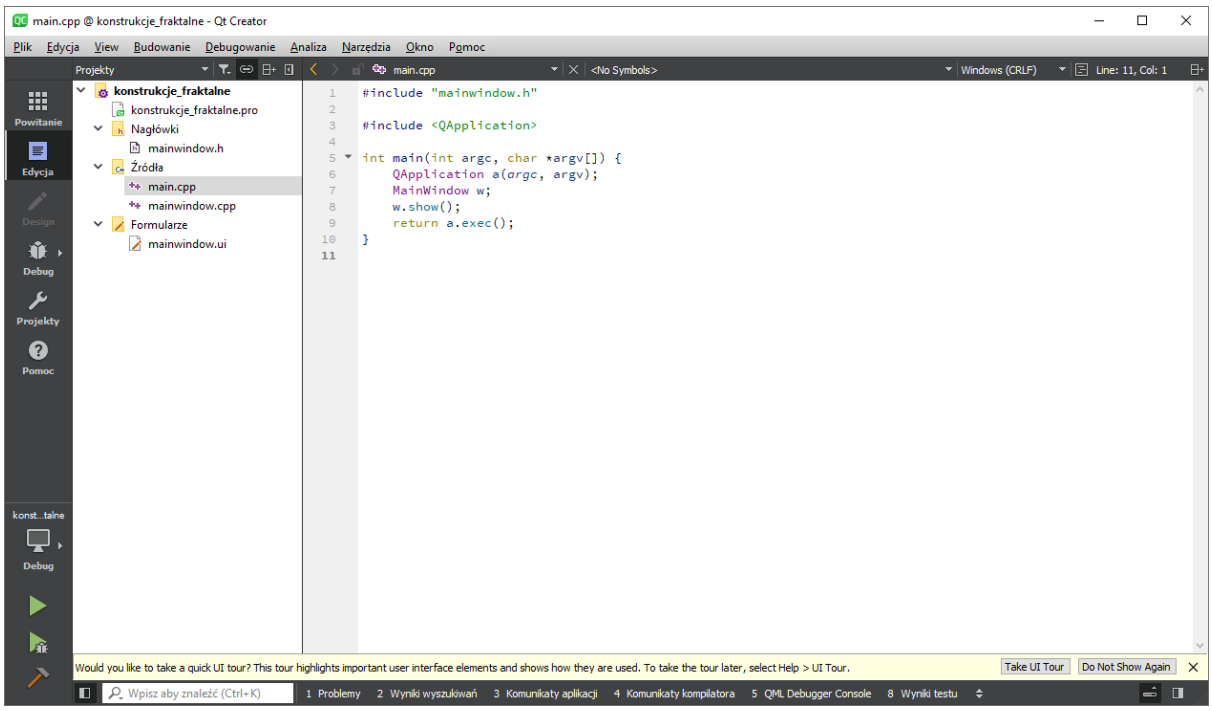
W tym momencie uruchomi się proces instalacji. Po jego zakończeniu możemy uruchomić edytor graficzny Qt Creator. Domyślnie IDE wykryje język systemu operacyjnego.
W przypadku, gdybyśmy chcieli zmienić język z angielskiego na polski, można to zrobić, wybierając:`Tools/Options.../Environment/Interface/Language`. Wówczas wybieramy z listy interesujący nas język i potwierdzamy wybór opcją `OK`, a następnie `Restart now`.

Pierwsza rzecz, jaką trzeba zrobić, aby zbudować przykładowy program, to utworzenie nowego projektu z menu `Plik`. Wybieramy opcję `Nowy plik lub projekt`, a dalej – szablon `Application (Qt)` oraz `Qt Widgets Application`.



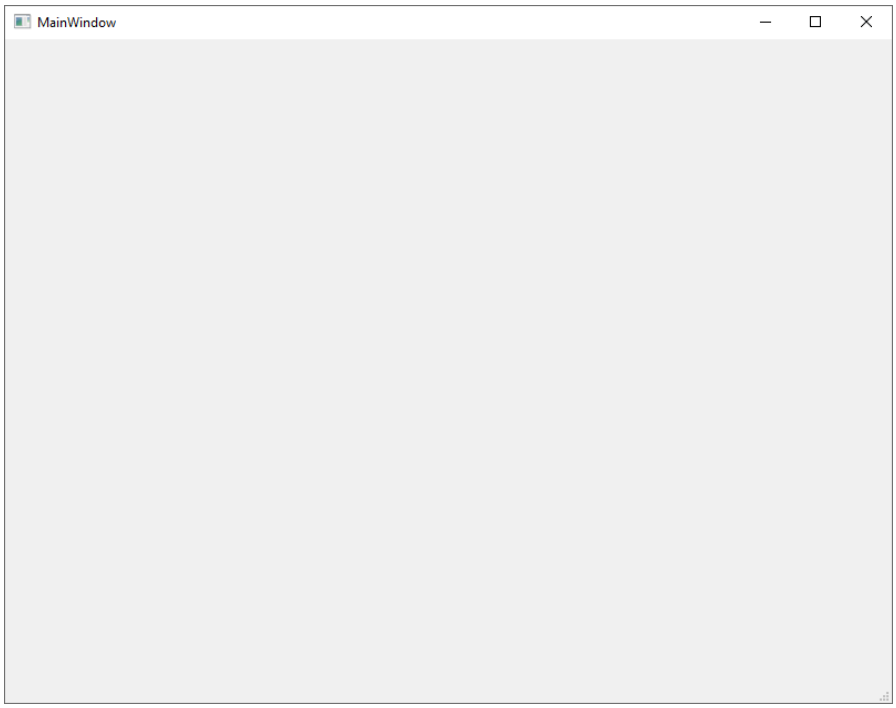
Źródło: Contentplus.pl Sp. z o.o., licencja: CC BY-SA 3.0.

Następnie należy wprowadzić nazwę projektu oraz wybrać katalog, w którym chcemy go zapisać. W następnych oknach zalecany jest wybór domyślnych opcji aż do momentu, w którym będziemy mogli wybrać `Zakończ`. Po utworzeniu projektu według powyższych zaleceń ukaże się nam następująca struktura plików programu:



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

Tak utworzony program możemy uruchomić, wybierając opcję `Uruchom` (zielony trójkąt w lewym dolnym rogu). Po poprawnym wykonaniu tej czynności na ekranie ukaże się następujący program:



Ponieważ tematem tego e-materiału nie jest bezpośrednio obsługa biblioteki graficznej Qt5 ani związana z nią obiektowość, dalej będziemy korzystać z szablonu napisanego specjalnie na potrzeby rysowania punktów na ekranie. W szablonie zawarta jest metoda `drawPoint(double x, double y)` obiektu `MainWindow`, która pozwoli nam narysować w powyższym oknie punkt na określonej pozycji. Ponieważ piksele w programie numerowane są od zera oraz w stronę przeciwną na osi OY niż w układzie współrzędnych, szablon zawiera przekształcenia, które pozwolą nam traktować przestrzeń programu jak układ kartezjański o skali dobranej do tworzonego fraktala.

Smok Heighwaya

Przed przystąpieniem do omówienia procedury generowania konstrukcji smoka Heighwaya przygotujmy strukturę programu.

W pliku `Nagłówki/mainwindow.h` należy umieścić następujący kod:

```
#ifndef MAINWINDOW_H
#define MAINWINDOW_H

#include <QMainWindow>
#include <list>

using namespace std;

QT_BEGIN_NAMESPACE
```

```
namespace Ui { class MainWindow; }
QT_END_NAMESPACE

class MainWindow : public QMainWindow
{
    Q_OBJECT

public:
    MainWindow(QWidget *parent = nullptr);
    ~MainWindow();

    virtual void paintEvent(QPaintEvent *event) override;
    void drawPoint(double x, double y);

private:
    Ui::MainWindow *ui;
    list<QPoint> points;
};
#endif // MAINWINDOW_H
```

Plik `Źródła/mainwindow.cpp` powinien zawierać:

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "qpainter.h"

using namespace std;

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    this->setFixedSize(700, 480);
    this->setWindowTitle("Smok Heighwaya");
}

MainWindow::~MainWindow() {
    delete ui;
}

```

```
void MainWindow::drawPoint(double x, double y) {
    int new_x = x * 400.0 + 180;
    int new_y = 300 - y * 400.0;

    points.push_back(QPoint(new_x, new_y));
}

void MainWindow::paintEvent(QPaintEvent *event) {
    QPainter painter(this);

    for (auto& point : points) {
        painter.drawPoint(point);
    }
}
```

Oraz ostatni plik – `Źródła/main.cpp`:

```
#include "mainwindow.h"

#include <QApplication>
#include <cstdlib>

using namespace std;

int main(int argc, char *argv[]) {
    // Początek kodu - utworzenie okna do rysowania
    QApplication a(argc, argv);
    MainWindow w;

    /* początek właściwej części kodu */

    /* koniec właściwej części kodu */

    // Koniec programu - wyświetlenie wyniku
    w.show();
    return a.exec();
}
```

W tym momencie w pliku `main.cpp` możemy pisać właściwy kod aplikacji, odpowiedzialny za generowanie fraktali. Aby narysować punkt na określonej pozycji, możemy skorzystać

z linijki:

```
w.drawPoint(x, y);
```

Na początku warto podjąć samodzielną próbę wygenerowania smoka Heighwaya, zgodnie z omówionym w poprzedniej lekcji algorytmem, którego pseudokod wygląda następująco:

```
// punkt startowy
x = 0.0
y = 0.0

// liczba generowanych punktów
n = 10000

wykonuj n razy:
    // kopia zmiennej x
    x0 = x

    g = losuj liczbę z [0, 1]

    jeżeli g == 0 wykonuj
        x = 0.5 * x - 0.5 * y
        y = 0.5 * x0 + 0.5 * y
    w przeciwnym wypadku
        x = - 0.5 * x - 0.5 * y + 1
        y = 0.5 * x0 - 0.5 * y

    rysuj punkt (x, y)
```

Najpierw utworzymy punkt początkowy, od którego rozpoczniemy przekształcenia przy użyciu funkcji IFS. Ponieważ punkt w układzie współrzędnych reprezentowany jest jako dwie liczby, utworzymy tylko dwie zmienne typu zmiennoprzecinkowego `double` o nazwach `x` oraz `y`, które będą stanowiły współrzędną x oraz y:

```
// punkt startowy
double x = 0.0, y = 0.0;
```

Kolejnym krokiem jest zdefiniowanie liczby punktów `n`, których użyjemy do utworzenia fraktala. Następnie utworzymy pętlę, która wykona się n razy. W tym celu możemy

wykorzystać zarówno pętlę while, jak i for. W naszym przypadku skorzystamy z drugiej opcji:

```
// liczba generowanych punktów
int n = 10000;

for (int i = 0; i < n; ++i) {
    // tworzenie kolejnych punktów
}
```

Zastanówmy się, w jaki sposób możemy wygenerować losową liczbę całkowitą z przedziału $\langle 0, 1 \rangle$. We wcześniejszych lekcjach poznaliśmy już sposób obsługi funkcji `rand()` z biblioteki `<cstdlib>`. Wiemy też, że można ograniczyć jej zakres działania przez operację dzielenia modulo. Do wyboru losowej liczby możemy zatem wykorzystać poniższy zapis:

```
// losowa liczba całkowita z przedziału <0, 1>
int q = rand() % 2;
```

Teraz możemy już przejść do modelowania funkcji IFS. Przed przystąpieniem do tej czynności należy jednak zrobić kopię zmiennej `x`, aby podczas generacji nowej wartości `y` nie wykorzystać nowej wartości `x`.

```
// kopia zmiennej x
double x0 = x;

if (q < 1) {
    x = 0.5 * x - 0.5 * y;
    y = 0.5 * x0 + 0.5 * y;
}
else {
    x = - 0.5 * x - 0.5 * y + 1;
    y = 0.5 * x0 - 0.5 * y;
}
```

Ostatnim krokiem będzie dodanie w pętli linijki, która wyśle utworzony punkt do narysowania.

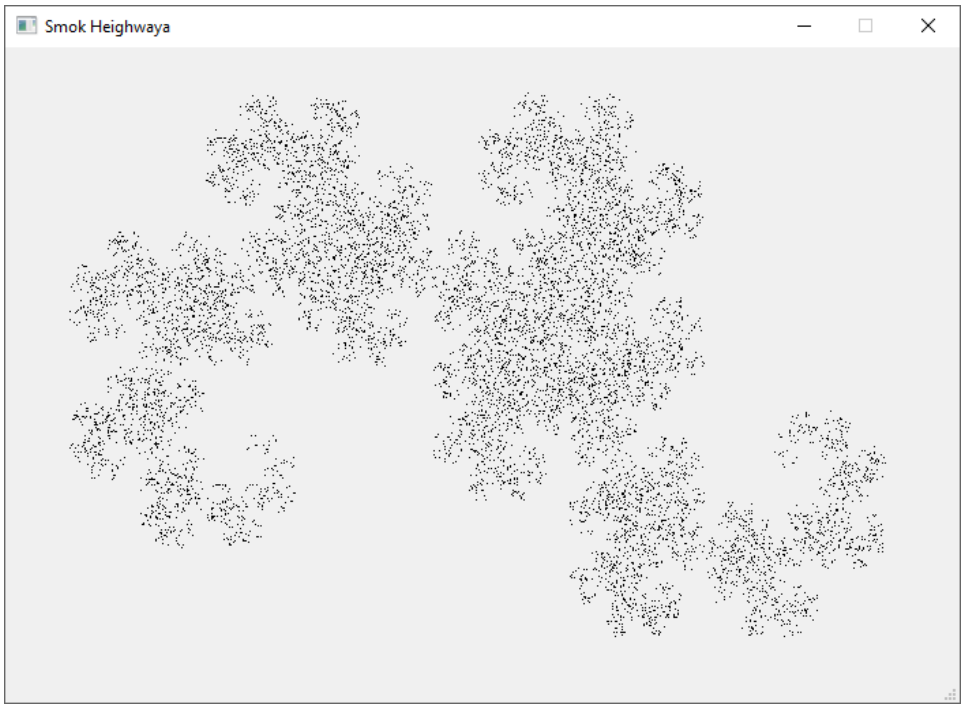
```
// Wysłanie punktu do narysowania
w.drawPoint(x, y);
```

Całość kodu prezentuje się zatem następująco:

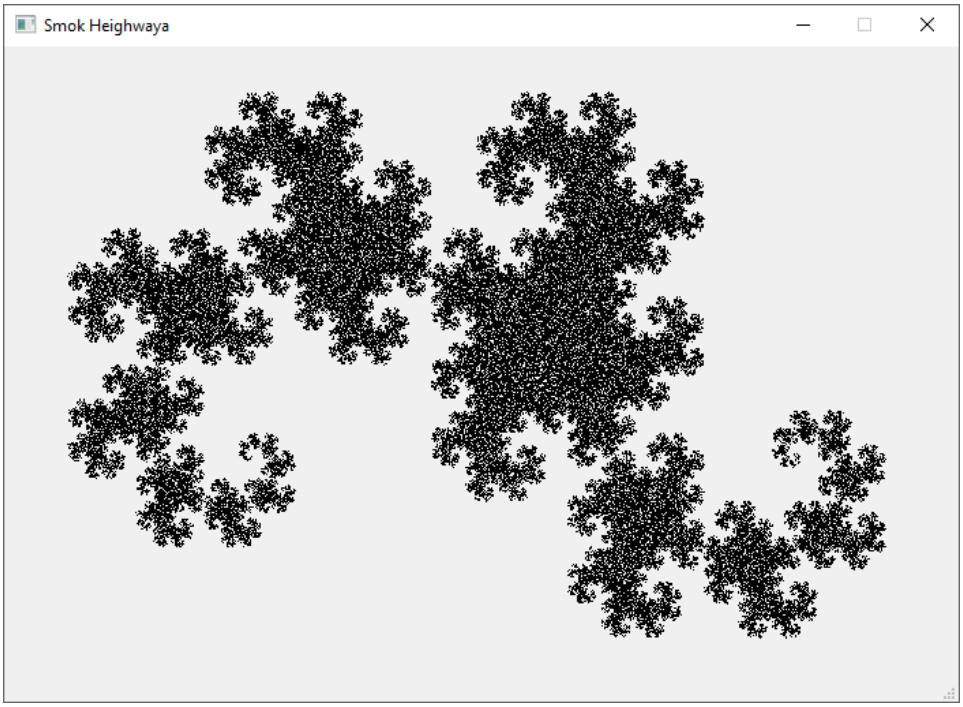
```
#include "mainwindow.h"

#include <QApplication>
#include <cstdlib>

using namespace std;

int main(int argc, char *argv[]) {
    // Początek kodu - utworzenie okna do rysowania
    QApplication a(argc, argv);
    MainWindow w;

    /* początek właściwej części kodu */
    // punkt startowy
    double x = 0.0, y = 0.0;
    // liczba generowanych punktów
    int n = 10000;

    for (int i = 0; i < n; ++i) {
        // losowa liczba całkowita z przedziału <0, 1>
        int q = rand() % 2;

        // kopia zmiennej x
        double x0 = x;

        if (q < 1) {
            x = 0.5 * x - 0.5 * y;
            y = 0.5 * x0 + 0.5 * y;
        }
        else {
            x = - 0.5 * x - 0.5 * y + 1;
            y = 0.5 * x0 - 0.5 * y;
        }

        // Wysłanie punktu do narysowania
        w.drawPoint(x, y);
    }

    /* koniec właściwej części kodu */
```

```
40
41      // Koniec programu - wyświetlenie wyniku
42      w.show();
43      return a.exec();
44  }
```

Uruchomienie tak napisanego programu spowoduje wyświetlenie następującego okna:



Jak widzimy, warto zwiększyć liczbę generowanych punktów. Oto jak będzie wyglądał wynik programu dla parametru n = 1000000:



Paproć Barnsleya

Ponieważ skala wielkości kształtów smoka Heighwaya oraz paproci Barnsleya, a także ich pozycja w układzie współrzędnych jest różna, szablon w przypadku paproci musi zostać zmodyfikowany. Jedyny plik, który zmienimy w tym celu, to `mainwindow.cpp` – należy umieścić w nim następujący kod:

```
#include "mainwindow.h"
#include "ui_mainwindow.h"
#include "qpainter.h"

MainWindow::MainWindow(QWidget *parent)
    : QMainWindow(parent)
    , ui(new Ui::MainWindow)
{
    ui->setupUi(this);
    this->setFixedSize(380, 580);
    this->setWindowTitle("Paproć Barnsleya");
}

MainWindow::~MainWindow() {
    delete ui;
}

void MainWindow::drawPoint(double x, double y) {
```

```
    int new_x = x * 50.0 + 175;
    int new_y = 525 - y * 50.0;

    points.push_back(QPoint(new_x, new_y));
}

void MainWindow::paintEvent(QPaintEvent *event) {
    QPainter painter(this);

    for (auto& point : points) {
        painter.drawPoint(point);
    }
}
```

Jeżeli udało ci się wygenerować poprzedni fraktal, na pewno poradzisz sobie także z utworzeniem paproci Barnsleya. Algorytm zapisany za pomocą pseudokodu, który wtedy opracowaliśmy, prezentuje się następująco:

```
// punkt startowy
x = 0.0
y = 0.0

// liczba generowanych punktów
n = 10000

wykonuj n razy:
    // kopia zmiennej x
    x0 = x

    g = losuj liczbę z [0, 1, ..., 99]

    jeżeli q < 1 wykonuj
        x = 0;
        y = 0.16 * y;
    w przeciwnym wypadku jeżeli q < 86 wykonuj
        x = 0.85 * x + 0.04 * y;
        y = -0.04 * x0 + 0.85 * y + 1.6;
    w przeciwnym wypadku jeżeli q < 93 wykonuj
        x = 0.2 * x - 0.26 * y;
        y = 0.23 * x0 + 0.22 * y + 1.6;
    w przeciwnym wypadku
```

```
        x = -0.15 * x + 0.28 * y;
        y = 0.26 * x + 0.24 * y + 0.44;

    rysuj punkt (x, y)
```

Różnica pomiędzy generacją paproci Barnsleya a generacją smoka Heighwaya polega tylko na zastosowaniu innych funkcji IFS w innych proporcjach. Linijka, która odpowiada za wygenerowanie losowej liczby, zostanie tu zmieniona w taki sposób, aby generować liczbę całkowitą z przedziału $\langle 0, 99 \rangle$:

```
// losowa liczba całkowita z przedziału <0, 99>
int q = rand() % 100;
```

Natomiast część warunkową, która odpowiada za funkcje IFS, zmodyfikujemy do następującej postaci:

```
// kopia zmiennej x
double x0 = x;

if (q < 1) {
    x = 0;
    y = 0.16 * y;
}
else if (q < 86) {
    x = 0.85 * x + 0.04 * y;
    y = -0.04 * x0 + 0.85 * y + 1.6;
}
else if (q < 93) {
    x = 0.2 * x - 0.26 * y;
    y = 0.23 * x0 + 0.22 * y + 1.6;
}
else {
    x = -0.15 * x + 0.28 * y;
    y = 0.26 * x0 + 0.24 * y + 0.44;
}
```

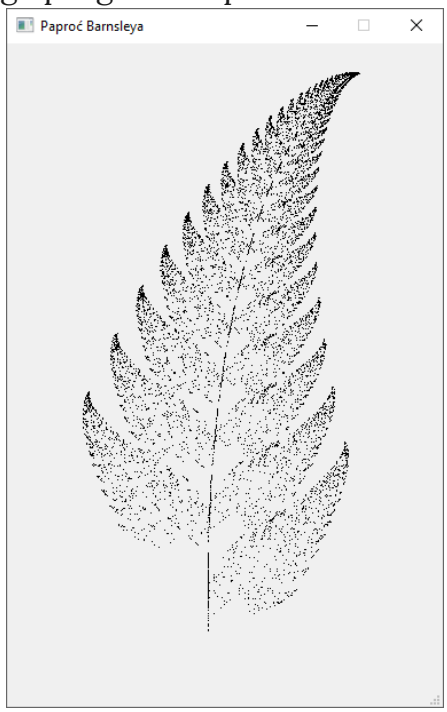
Ostateczny program będzie zatem wyglądał tak:

```
#include "mainwindow.h"
```

```
2
3  #include <QApplication>
4  #include <cstdlib>
5
6  using namespace std;
7
8  int main(int argc, char *argv[]) {
9      // Początek kodu - utworzenie okna do rysowania
10     QApplication a(argc, argv);
11     MainWindow w;
12
13     /* początek właściwej części kodu */
14     // punkt startowy
15     double x = 0.0, y = 0.0;
16     // liczba generowanych punktów
17     int n = 10000;
18
19     for (int i = 0; i < n; ++i) {
20         // losowa liczba całkowita z przedziału <0, 99>
21         int q = rand() % 100;
22
23         // kopia zmiennej x
24         double x0 = x;
25
26         if (q < 1) {
27             x = 0;
28             y = 0.16 * y;
29         }
30         else if (q < 86) {
31             x = 0.85 * x + 0.04 * y;
32             y = -0.04 * x0 + 0.85 * y + 1.6;
33         }
34         else if (q < 93) {
35             x = 0.2 * x - 0.26 * y;
36             y = 0.23 * x0 + 0.22 * y + 1.6;
37         }
38         else {
39             x = -0.15 * x + 0.28 * y;
40             y = 0.26 * x0 + 0.24 * y + 0.44;
41         }
42
43         // Wysłanie punktu do narysowania
```

```
44          w.drawPoint(x, y);
45      }
46
47      /* koniec właściwej części kodu */
48
49      // Koniec programu - wyświetlenie wyniku
50      w.show();
51      return a.exec();
52  }
```

A oto rezultat uruchomienia tego programu z parametrem n = 10000:



Warto znów dopasować liczbę wygenerowanych punktów. Oto wynik programu uruchomionego z parametrem n = 1000000:

Słownik

IDE

(od ang. *integrated development environment*) zintegrowane środowisko programistyczne; oprogramowanie służące do tworzenia oraz modyfikowania projektów programistycznych

IFS

(z ang. *iterated function system*) systemem funkcji iterowanych; rodzina funkcji służących do generowania fraktali

Prezentacja multimedialna

Polecenie 1

Przeanalizuj prezentację, w której omówimy metodę tworzenia fraktali podobną do przekształceń IFS. Podczas analizy staraj się tworzyć program lokalnie na swoim komputerze.

1

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Przekształcenia IFS to nie jedyne sposoby generowania fraktali. Innym opartym na losowości może być na przykład gra w chaos. Algorytm takiej metody polega na wybraniu pewnych punktów startowych, a następnie utworzeniu punktu zwanego dalej punktem gry. Jako pierwotne współrzędne punktu gry ustawimy współrzędne losowego punktu spośród punktów startowych. Następnie w pętli dokonujemy pewnej operacji matematycznej z użyciem punktu gry oraz losowo wybranego punktu startowego (np. operacja wyznaczenia środka pomiędzy punktami).

2

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Utworzony w wyniku określonej operacji punkt rysujemy, a następnie oznaczamy go jako punkt gry. Pętlę powtarzamy określoną liczbę razy. Jeśli w odpowiedni sposób dobierzemy operację oraz punkty startowe, na koniec otrzymamy kształt fraktalny.

3

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Za pomocą gry w chaos możemy utworzyć trójkąt Sierpińskiego. Jeśli wybierzemy trzy punkty startowe, które będą wierzchołkami trójkąta równobocznego, a operacją, którą wykonujemy w pętli, będzie wybranie punktu środkowego między punktem gry a losowo wybranym punktem startowym, wówczas narysujemy kształt trójkąta Sierpińskiego.

4

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Jako punkty startowe wybierzmy punkty $(0,\ 0)$, $(1,\ 0)$ oraz $\left(0.5, \frac{\sqrt{3}}{2}\right)$.

Dostosujmy szablon, aby w odpowiedni sposób wyświetlał obszar pomiędzy tymi punktami. W projekcie w programie Qt Creator w pliku *mainwindow.cpp* umieśćmy następujący kod:

5

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Utwórzmy tablicę, w której zdefiniujemy punkty startowe. W celu obliczenia pierwiastka skorzystamy z funkcji `std::sqrt()` z biblioteki `<cmath>`

6

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Wybierzmy punkt startowy. W tym celu wygenerujmy losową liczbę całkowitą od 0 (włącznie) do 2 (włącznie). Następnie przekopiujmy wybrany losowy punkt do punktu gry. W celu wygenerowania losowej liczby skorzystamy z funkcji std::rand() z biblioteki `<cstdlib>` oraz operacji dzielenia modulo, aby ograniczyć przedział losowanych liczb.

7

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Utwórzmy pętlę, którą wykonamy n = `1000000` razy:

8

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

W pętli w pierwszej kolejności należy wyznaczyć losowy punkt startowy. Ponieważ operacja, którą chcemy przeprowadzić, to wyznaczenie punktu środkowego, wystarczy dodać współrzędne x-owe punktu gry oraz wybranego punktu, a następnie podzielić je przez 2. Tak samo postępujemy dla

współrzędnej y-owej. Wówczas wyznaczony punkt możemy ustawić jako punkt gry oraz wysłać go do rysowania.

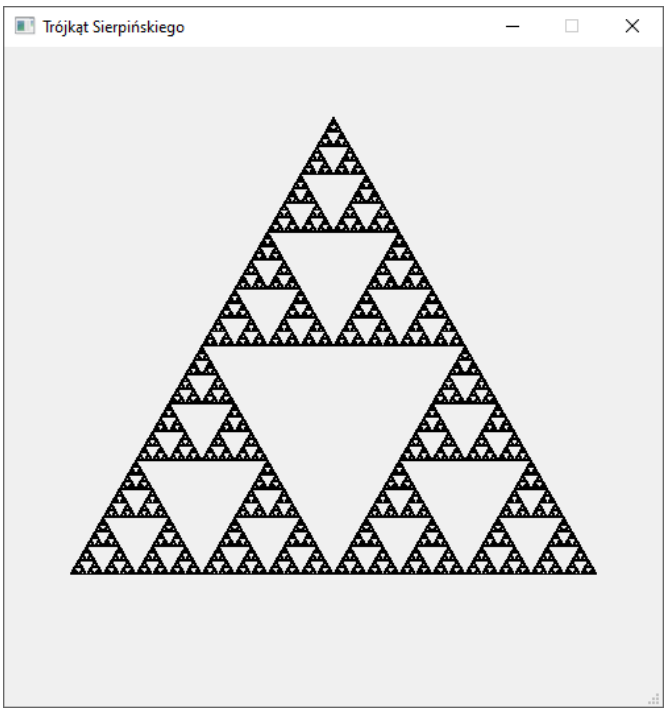
9

Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Program po złożeniu wszystkich jego części w całość wygląda następująco:

10



Materiał audio dostępny pod adresem:

https://zpe.gov.pl/b/Pvq5X39MF

Po uruchomieniu tak stworzonego programu zostanie wyświetlone zaprezentowane okno.

Polecenie 2

Opracuj notatkę podsumowującą najważniejsze informacje przedstawione w prezentacji.

Sprawdź się

Pokaż ćwiczenia: 

Ćwiczenie 1

Napisz program, który wygeneruje fraktal smok Heighwaya, jednak zamiast rysować go na ekranie, wyznacz pole prostokątnego obszaru określonego przez proste prostopadłe lub równoległe do osi układu współrzędnych styczne do smoka Heighwaya. Wynik zaokrąglij do jednego miejsca po przecinku. Działanie programu przetestuj dla `n = 100000`.

Specyfikacja problemu:

Dane:

- n – liczba naturalna
- x, y – liczba wymierne

Wynik:

Program wyświetla zaokrąglone do jednego miejsca po przecinku pole prostokątnego obszaru wyznaczonego przez proste prostopadłe lub równoległe do osi układu współrzędnych styczne do smoka Heighwaya.

Twoje zadania

1. Program wyznacza pole prostokątnego obszaru określonego przez proste prostopadłe lub równoległe do osi układu współrzędnych styczne do smoka Heighwaya. Wynik zaokrągla do jednego miejsca po przecinku.

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

int main() {
    // W tym miejscu dodaj implementację swojego rozwiązania
}
```

```
1
```

Ćwiczenie 2

Napisz program, który przeprowadzi generację paproci Barnsleya, jednak zamiast rysować ją na ekranie, wyznacz pole prostokątnego obszaru wyznaczonego przez proste styczne do paproci Barnsleya prostopadłe lub równoległe do osi układu współrzędnych. Wynik zaokrąglij do jednego miejsca po przecinku. Działanie programu przetestuj dla `n = 100000`.

Specyfikacja problemu:

Dane:

- n – liczba naturalna
- x, y – liczba wymierne

Wynik:

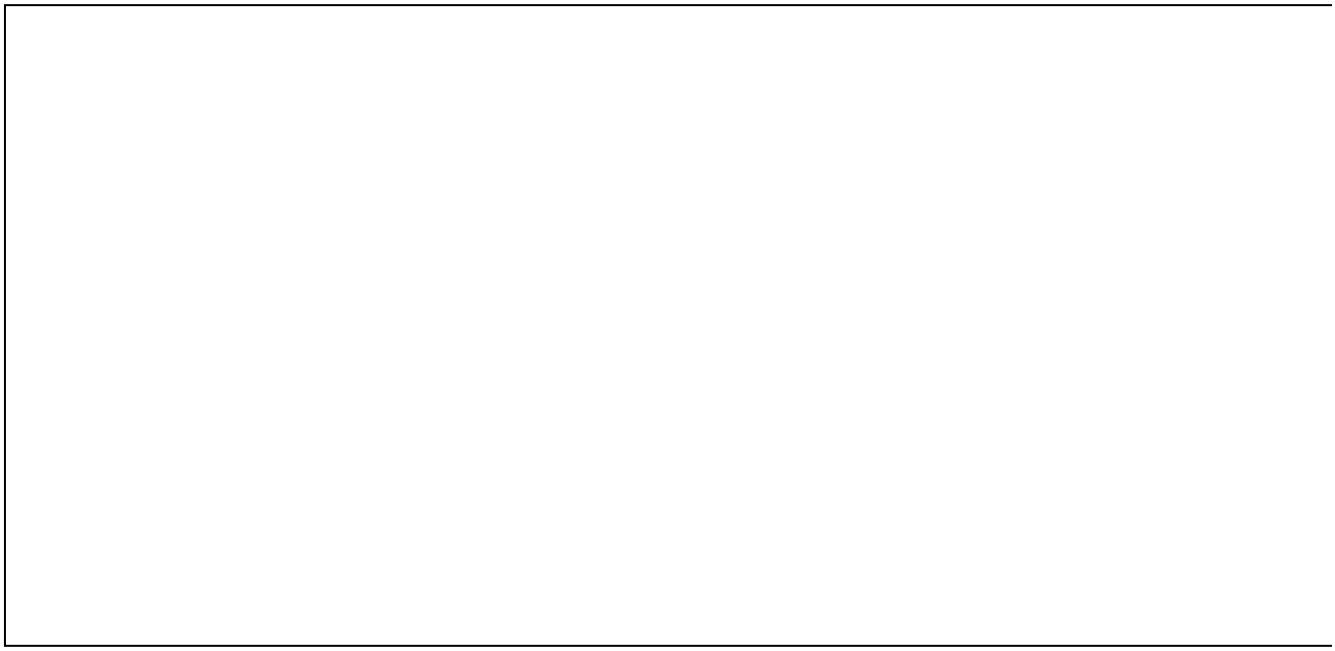
Program wyświetla zaokrąglone do jednego miejsca po przecinku pole prostokątnego obszaru wyznaczonego przez proste styczne do paproci Barnsleya prostopadłe lub równoległe do osi układu współrzędnych.

Twoje zadania

1. Program wyznacza pole prostokąta wyznaczonego przez proste styczne do paproci Barnsleya prostopadłe lub równoległe do osi układu współrzędnych.

```cpp
#include <iostream>

int main() {
    // W tym miejscu dodaj implementację swojego rozwiązania
}
```

```
1
```



Dla nauczyciela

Autor: Maurycy Gast
Przedmiot: Informatyka

Temat: Konstrukcje fraktalne w języku C++

Grupa docelowa:

Szkoła ponadpodstawowa, 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

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:

1a) rekurencyjnego tworzenia fraktali: zbiór Cantora, drzewo binarne, dywan Sierpińskiego, płatek Kocha;

3) objaśnia, a także porównuje podstawowe metody i techniki algorytmiczne oraz struktury danych, wykorzystując przy tym przykłady problemów i algorytmów, w szczególności:

b) rekurencję (do generowania ciągów liczb, potęgowania, sortowania liczb, generowania fraktali),

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):

- Przeanalizujesz sposób wyznaczania liczb pseudolosowych w celu utworzenia obiektów samopodobnych.
- Zbudujesz graficzny program wyświetlający fraktale na ekranie.
- Utworzysz paproć Barnsleya oraz smoka Heighwaya, korzystając z przekształcenia afinicznego oraz generatora liczb pseudolosowych.

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: „Konstrukcje fraktalne w języku C++". Uczniowie mają zapoznać się z problemem 1 oraz filmem z sekcji „Przeczytaj".

Faza wstępna:

1. Nauczyciel wyświetla i odczytuje temat lekcji oraz cele zajęć. Prosi uczniów o sformułowanie kryteriów sukcesu.
2. **Rozpoznanie wiedzy uczniów.** Nauczyciel prosi wybranego ucznia lub uczniów o przedstawienie sytuacji problemowej związanej z tematem lekcji.

Faza realizacyjna:

1. **Praca z tekstem.** Uczniowie przystępują do cichego czytania tekstu e-materiału. Indywidualnie zapoznają się z treścią w sekcji „Przeczytaj".
2. **Praca z multimedium.** Nauczyciel czyta polecenie nr 1 „Przeanalizuj prezentację, w której omówimy metodę tworzenia fraktali podobną do przekształceń IFS. Podczas analizy staraj się tworzyć program lokalnie na swoim komputerze." w sekcji „Prezentacja multimedialna". Prosi uczniów, aby w parach przeanalizowali rozwiązanie problemu. Uczniowie odtwarzają kolejne kroki na swoich komputerach.
3. **Ćwiczenie umiejętności.** Prowadzący zapowiada uczniom, że będą rozwiązywać ćwiczenie nr 1 z sekcji „Sprawdź się". Każdy z uczniów robi to samodzielnie. Po ustalonym czasie następuje porównanie napisanych kodów podczas wspólnego omówienia rozwiązań.

Faza podsumowująca:

1. Wybrany uczeń podsumowuje zajęcia z programowania w C++, zwracając uwagę na nabyte umiejętności.

Praca domowa:

1. Uczniowie wykonują ćwiczenie 2 z sekcji „Sprawdź się".

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 „Prezentacja multimedialna” można wykorzystać jako materiał służący powtórzeniu materiału.