

## § 18. Составление алгоритмов для работы с графикой

### 18.1. Расчеты в графических построениях

**Пример 18.1.** Нарисовать прямоугольный треугольник, соответствующий рисунку (катеты треугольника параллельны осям координат). Длины катетов и координаты прямого угла вводятся.



Этапы выполнения задания

I. Исходные данные:  $a$  и  $b$  (длины катетов),  $x$  и  $y$  (координаты вершины прямого угла).

II. Результат: изображение прямоугольного треугольника.

III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Чтобы изобразить треугольник, нужно выполнить следующие действия:

1) построить линии из точки с координатой  $(x; y)$  в точки с координатами  $(x + a; y)$  и  $(x; y + b)$ ;

2) соединить линией точки  $(x + a; y)$  и  $(x; y + b)$ ;

3) закрасить треугольник. Для закрасивания треугольника нужно знать координаты какой-либо точки внутри треугольника. Такой точкой в

#### Пример 18.1.

V. Программа:

```
uses GraphABC;
var a,b,x,y,x_c, y_c:integer;
begin
  writeln('Введите координаты');
  read(x,y); writeln(x,' ',y);
  writeln('Введите длины катетов');
  read(a,b); writeln(a,' ',b);
  Line(x,y,x+a,y); Line(x,y,x,y+b);
  Line(x+a,y,x,y+b);
  //Координаты точки
  //Внутри треугольника
  x_c := x + 2; y_c := y + 2;
  FloodFill(x_c,y_c,clRandom);
end.
```

VI. Тестирование.

Запустить программу и ввести значения: координаты (100; 100), длины катетов 70 и 120. Результат:



Другой вариант:



**Пример 18.2\*.**

V. Программа:

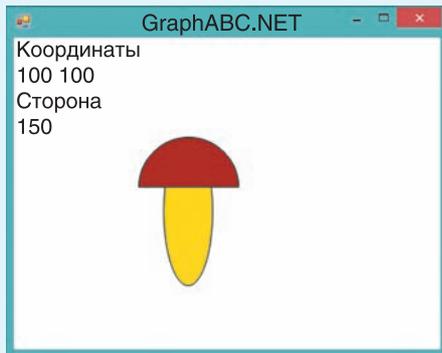
```

uses GraphABC;
var x,y,d: integer;
begin
  writeln('Координаты');
  read(x,y);
  writeln(x,' ',y);
  writeln('Сторона');
  read(d);
  writeln(d);
  SetBrushColor(clYellow);
  Ellipse(x + d div 3,y,
          x + 2 * d div 3,y + d);
  SetBrushColor(clBrown);
  Pie(x + d div 2,y + d div 3,
      d div 3,180);
end.

```

VI. Тестирование.

Запустить программу и ввести значения: координаты (100; 100), сторона 150. Результат:



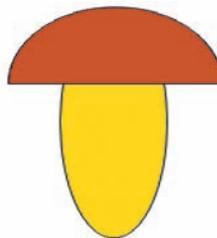
\* Гриб можно вписывать не в квадрат, а в прямоугольник. В этом случае нужно задавать две величины, определяющие размеры прямоугольника: длину ( $d_1$ ) и ширину ( $d_2$ ).

данном случае может быть точка с координатами  $(x + 2; y + 2)$ <sup>1</sup>.

IV. Описание переменных: все переменные имеют тип `integer`.

Многие графические построения можно обобщить, если предположить, что фигура должна быть вписана в квадрат. В этом случае для построения фигуры достаточно задать координаты  $(x; y)$  верхнего левого угла квадрата и длину его стороны —  $d$ . Используя эти величины, можно получить координаты других вершин квадрата:  $(x + d; y)$ ,  $(x; y + d)$ ,  $(x + d; y + d)$ . Можно получить координаты середины стороны  $(x + d \text{ div } 2; y)$  или центра квадрата  $(x + d \text{ div } 2; y + d \text{ div } 2)$ .

**Пример 18.2\*.** Нарисовать в графическом окне гриб. Задать координаты верхнего левого угла квадрата и длину его стороны для определения местоположения и размеров гриба.



Этапы выполнения задания

I. Исходные данные:  $x$  и  $y$  — координаты верхнего левого угла квадрата, в который вписан гриб,  $d$  — длина его стороны.

II. Результат: изображение гриба.

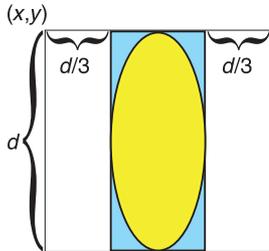
<sup>1</sup> Для произвольного треугольника можно воспользоваться формулами  $\left( \frac{x_1 + x_2 + x_3}{3}; \frac{y_1 + y_2 + y_3}{3} \right)$ .

### III. Алгоритм решения задачи.

1. Ввод исходных данных.

2. Для того чтобы построить гриб, нужно выполнить следующие действия:

1) построить овал для изображения ножки гриба. Параметры команды для изображения эллипса определим следующим образом: `ellipse(x + d div 3, y, x + 2*d div 3, y + d);`



2) нарисовать шляпку гриба. Для этого можно использовать команду `Pie` (построение сектора круга). Координаты центра  $(x + \frac{d}{2}; y + \frac{d}{3})$ , радиус —  $\frac{d}{3}$ . Углы равны  $0^\circ$  и  $180^\circ$  соответственно.

IV. Описание переменных: все переменные имеют тип `integer`.

Случайные числа имеют широкое применение в программировании. Они используются в шифровании и в моделировании. Многие компьютерные игры используют случайные числа. На основе случайных чисел генерируются капчи и пароли, реализуются различные лотереи.

В `PascalABC` для получения случайного числа используют функцию `random`. Способы записи функции:

`Random(a, b);` — возвращает случайное целое в диапазоне от  $a$  до  $b$ ;

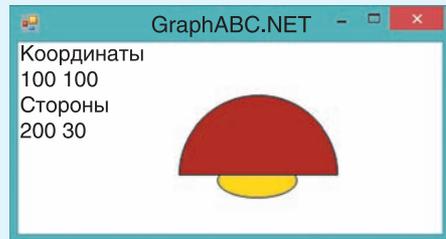
`Random(a);` — возвращает случайное целое в диапазоне от  $0$  до  $a - 1$ ;

#### Пример 18.2\*. Продолжение.

В программу рисования гриба нужно внести изменения, позволяющие рассчитать положение ножки и шляпки гриба относительно координат  $(x; y)$  и величин  $d1$  и  $d2$ .

```
var x,y,d1,d2: integer;
...
writeln('Стороны');
read(d1, d2);
writeln(d1, ' ', d2);
SetBrushColor(clYellow);
Ellipse(x+d1 div 3, y,
        x + 2 * d1 div 3, y + d2);
SetBrushColor(clBrown);
Pie(x + d1 div 2, y + d2 div 3,
    max(d1,d2) div 3, 0, 180);
```

Результат:



**Случайное число** — число, которое принимает одно значение из множества, причем появление того или иного значения нельзя точно предсказать. Например, если бы числа появились в результате вытягивания бочонков в лото, то такая последовательность чисел была бы случайной.

В языках программирования используют **псевдослучайные числа**, которые получают с использованием генератора случайных чисел — алгоритма, порождающего последовательность чисел, элементы которой почти независимы друг от друга и обычно распределены равномерно на заданном интервале.

**Пример 18.3.**

V. Программа:

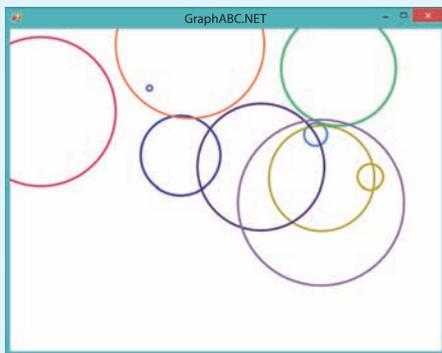
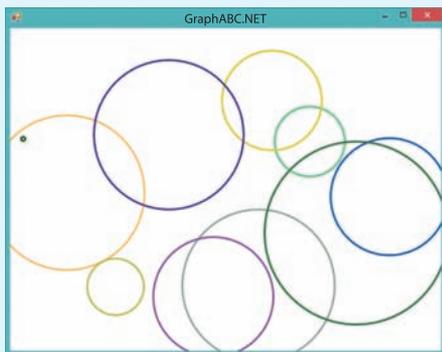
```

uses GraphABC;
var x,y,r: integer;
begin
  SetPenWidth(3);
  SetBrushStyle(bsClear);
  for var i:= 1 to 10 do
  begin
    x := random(600);
    y := random(400);
    R := random(150);
    SetPenColor(clRandom);
    circle(x,y,r);
  end;
end.

```

VI. Тестирование.

Запустить программу. Должно быть нарисовано 10 окружностей. Разные варианты работы программы:



Random; — возвращает случайное вещественное в диапазоне [0..1).

Функция clRandom позволяет задать случайный цвет.

**Пример 18.3.** Написать программу для рисования на экране 10 разноцветных окружностей. Расположение окружностей, их радиусы и цвет определяются случайным образом.

Этапы выполнения задания

I—II. Результатом работы программы, не зависящей от исходных данных, будет изображение 10 окружностей.

III. Алгоритм решения задачи.

1. Установим толщину линий в 3 пикселя и прозрачную заливку.

2. Значения координат центра окружности и ее радиуса определяются функцией random. Значение цвета для границы круга — clRandom.

3. Так как количество повторений известно, будем использовать цикл for.

IV. Описание переменных:  $x$ ,  $y$  (координаты центра),  $r$  (радиус) имеют тип integer.

## 18.2. Использование вспомогательных алгоритмов

Построение фигур можно оформлять в виде вспомогательных алгоритмов. Это позволит использовать такие алгоритмы для решения других задач.

Все графические процедуры, которые использовались ранее, имели параметры. Они позволяли определять местоположение и размер фигур. Пользователь также может составить свой вспомогательный алгоритм с параметрами.

Общий вид процедуры с параметрами:

```
procedure <имя> (<список
    параметров>:тип);
var ...
begin
    <команды>
end;
```

При вызове процедуры важно помнить, что количество параметров и их порядок должны соответствовать тому, как процедура описана.

**Пример 18.4.** Написать программу для построения  $n$  равнобедренных прямоугольных треугольников с длиной катета  $a$ . Расположение треугольников определяется случайным образом.

Этапы выполнения задания

I. Исходные данные:  $n$  (количество треугольников),  $a$  (длина катета).

II. Результат: изображение  $n$  треугольников.

III. Алгоритм решения задачи.

1. В примере 18.3 изображали окружности, расположенные случайным образом, а в примере 18.1 — прямоугольные треугольники. Воспользуемся программами этих примеров.

2. Ввод значений переменных  $n$  и  $a$ .

3. Так как количество повторений известно, будем использовать цикл **for**.

4. Изменим программу из примера 18.3. Для этого команду **circle** (построение окружности) заменим на команду построения треугольника:

1) местоположение треугольника задается координатами прямого угла, которые определим случайным образом;

2) катеты прямоугольного треугольника имеют одинаковую длину — значение  $a$ .

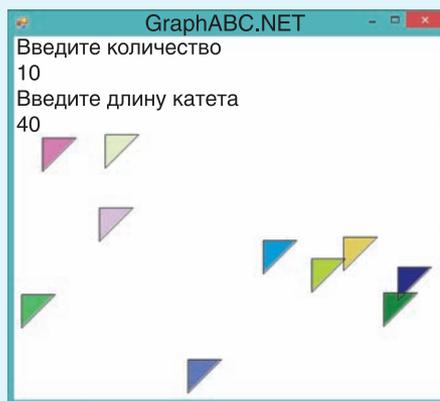
#### Пример 18.4.

V. Программа:

```
uses GraphABC;
var n, x, y, a : integer;
procedure pr_treug (x, y, a,
    b : integer);
var x_c, y_c:integer;
begin
    line(x, y, x + a,y);
    line(x, y, x, y + b);
    line(x + a, y, x, y + b);
    x_c := x + 2; y_c := y + 2;
    FloodFill(x_c,y_c,clRandom);
end;
begin
    writeln('Введите количество');
    read(n); writeln (n);
    writeln('Введите длину катета');
    read(a); writeln (a);
    for var i:= 1 to n do
        begin
            x:= random(500);
            y:= random(400);
            pr_treug(x, y, a, a);
        end;
end.
```

VI. Тестирование.

Запустить программу. Результат:



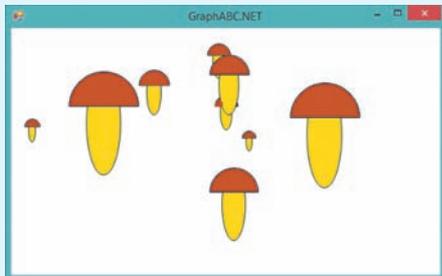
При вводе других значений результаты будут иными.

**Пример 18.5\*.**

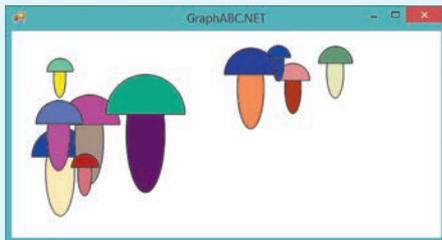
```
V. Программа:
uses GraphABC;
var x,y,d: integer;

procedure grib(x,y,d:integer);
begin
  SetBrushColor(clYellow);
  Ellipse(x + d div 3, y,
    x + 2*d div 3, y + d);
  SetBrushColor(clBrown);
  Pie(x + d div 2, y + d div 3,
    d div 3, 0, 180);
end;

Begin
  for var i:= 1 to 10 do
  begin
    x:= random(400);
    y:= random(200);
    d:= random(150);
    grib(x,y,d);
  end;
end.
VI. Тестирование.
Результат может быть следующим:
```



Можно добавить раскраску случайным цветом:



5. Построение одного прямоугольного треугольника опишем во вспомогательном алгоритме `pr_treug`. Параметры процедуры построения треугольника — координаты вершины прямого угла и длины катетов. Алгоритм описан в примере 18.2.

IV. Описание переменных: все переменные имеют тип `integer`.

**Пример 18.5\*.** Нарисовать 10 грибов. Расположение и их размеры определяются случайным образом.

Этапы выполнения задания

I—II. Результатом работы программы, не зависящим от исходных данных, будет изображение 10 грибов.

III. Алгоритм решения задачи.

1. Построение одного гриба опишем во вспомогательном алгоритме. Алгоритм описан в примере 18.2.

2. Значения координат верхнего левого угла и размер гриба определяются функцией `random`.

3. Так как количество повторений известно, будем использовать цикл `for`.

IV. Описание переменных:  $x$ ,  $y$  (координаты верхнего левого угла),  $d$  (размер) — `integer`.

**Пример 18.6.** Заполнить графическое окно окружностями с радиусом 10.

Этапы выполнения задания

I—II. Исходные данные отсутствуют. Окружности должны заполнить все графическое окно.

III. Алгоритм решения задачи.

1. Задача является обобщением задачи примера 17.1. Команды программы следует повторить для нескольких рядов окружностей. Количество рядов определяется высотой окна. Рисование

одного ряда оформим как вспомогательный алгоритм `row`.

2. Положение любого ряда окружностей определяется координатой  $y$ . Для каждого значения  $y$ , пока он не станет большим, чем вертикальный размер экрана, выполняем в цикле следующее:

- 1) рисуем ряд окружностей;
- 2) изменяем  $y$ .

IV. Описание переменных:  $x$ ,  $y$ ,  $r$  — `integer`.

В примере 18.6 показано, как заполнить графическое окно окружностями. Внеся небольшие изменения в эту программу, можно заполнять графическое окно любыми другими фигурками. Для этого достаточно заменить команду `Circle(x,y,r)` в процедуре `row` на другую команду. Можно выбрать графический примитив из библиотеки `GraphABC` или самостоятельно написать процедуру рисования фигурки (например, использовать процедуру рисования гриба из примера 18.5).

-  1. Как задать случайное число?  
2. Как задать случайный цвет?  
3. Как описать процедуру с параметрами?

### Упражнения

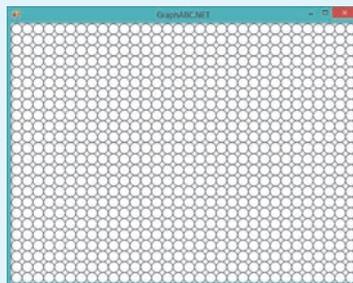
- 1 Выполните задания для примера 18.1.
  1. Поэкспериментируйте с программой, вводя различные значения исходных данных.
  2. Объясните, что происходит при вводе отрицательных значений длин катетов.
  3. Что произойдет, если ввести отрицательные значения координат? Объясните результат.
- 2 Выполните задания для примера 18.3.
  1. Выполните программу несколько раз. Уберите прозрачную заливку. Объясните, почему некоторые окружности не видны.

#### Пример 18.6.

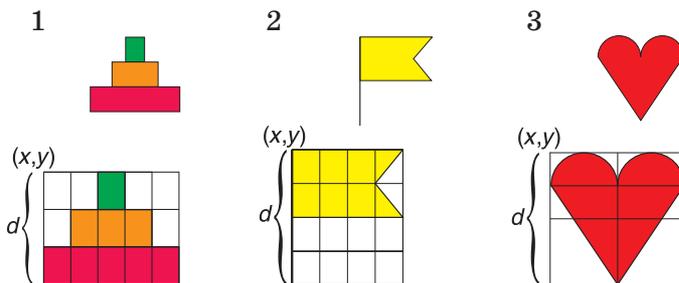
```
V. Программа:
uses GraphABC;
var x, y, r : integer;
procedure row(y : integer);
begin
  x :=10; R := 10;
  while x < WindowWidth do
    begin
      Circle(x,y,r); x := x+20;
    end;
end;
begin
  y := 10;
  while y <= WindowHeight do
    begin
      Row(y); y := y + 20;
    end;
end.
```

VI. Тестирование.

Результат:



2. Внесите в программу такие изменения, чтобы можно было изобразить 20 кругов; 100 кругов.
  3. Какой максимальный размер может иметь радиус круга в программе? Внесите в программу изменения так, чтобы рисовались круги с радиусом не более 20. Количество кругов установите равным 10 000.
  4. Внесите изменения в программу так, чтобы пользователь мог вводить количество отображаемых на экране кругов.
- 3 Выполните задания для примера 18.4.
1. Запустите программу несколько раз. Объясните, почему при некоторых запусках треугольники рисуются поверх текста в верхнем левом углу экрана. Измените программу так, чтобы треугольники рисовались ниже текста (правее текста).
  2. Добавьте в программу возможность ввода длины второго катета.
  3. Измените программу так, чтобы длины катетов задавались случайным образом.
- 4 Напишите программу, которая строит случайным образом изображения 20 горизонтальных отрезков длиной 30 пикселей. Разработайте два варианта решения задачи. Один с использованием цикла `while`, а другой — цикла `for`.
1. Сравните две программы решения задачи. Какой вариант решения данной задачи представляется вам лучшим? Почему?
  2. Задайте в программе толщину отрезка в 3 пикселя.
  3. Какие изменения нужны в программе, чтобы толщина отрезка была случайным числом из промежутка  $[2; 8]$ ?
  4. Внесите изменения в программу так, чтобы пользователь мог вводить количество отображаемых на экране отрезков.
  5. Какие изменения нужно внести в программу, чтобы вместо горизонтальных отрезков изображались вертикальные? Диагональные?
- 5 Используя процедуру рисования треугольника из примера 18.4, нарисуйте ряд треугольников вдоль верхнего (левого) края графического окна.
- 6\* Напишите программу для рисования одной из фигурок. Задаются координаты верхнего левого угла и длина стороны квадрата (длины сторон прямоугольника):



- 7\* Напишите программу, которая нарисует ряд фигурок вдоль края графического окна. Используйте фигурки, которые рисовали в задании 6.
- 8 Выполните задания для примера 18.6.
1. Измените в программе значение  $r = 10$  на  $r = 20$ . Почему получился такой рисунок? Поэкспериментируйте со значениями радиуса, установив прозрачную заливку.
  2. Какие изменения нужно внести в программу, чтобы экран заполнялся кругами с радиусом 20 без пересечений?
  3. Внесите изменения в программу так, чтобы все круги были красными или разноцветными.
  4. Внесите в программу изменения так, чтобы графическое окно можно было заполнять кругами введенного радиуса.
- 9\* Напишите программу, которая заполнит все графическое окно:
1. Грибами (пример 18.2).
  2. Фигурками из задания 6.

## § 19. Использование основных алгоритмических конструкций для решения практических задач

### 19.1. Использование числовых последовательностей

Числовые последовательности позволяют описывать многие процессы, происходящие в природе и обществе.

Например, последовательность чисел 2, -1, 0, 2, 0, 1, -2 может задавать значения температуры по дням недели; последовательность 746, 751, 758 — значения средней заработной платы сотрудников предприятия за квартал и т. д.

Последовательности могут задаваться формулой, в которой значение элемента зависит от того, какой у него номер в последовательности (пример 19.1).

Другим способом задания элементов последовательности является определение значения нового элемента

Подтверждением важности числовых последовательностей является тот факт, что создана целая энциклопедия числовых последовательностей<sup>1</sup>.

**Пример 19.1.** Элементы последовательности нечетных положительных чисел можно описать с помощью формулы  $a_n = 2n - 1$ . В этой формуле  $n$  — номер элемента в последовательности. Минимальное значение числа  $n = 1$ . Используя формулу, получим последовательность: 1, 3, 5, 7, ... .

Элементы последовательности могут быть действительными числами. Например, формула  $a_n = \frac{n}{n^2 + 1}$  задает следующую последовательность: 0.5, 0.4, 0.3, 0.235, 0.192, ... .

<sup>1</sup> Онлайн-энциклопедия целочисленных последовательностей. Режим доступа: <http://oeis.org/?language=russian> (дата доступа: 17.01.2018).