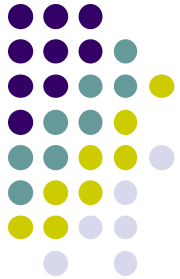
Программа: Число π и метод Монте Карло

Курс «Основы программирования в среде Delphi»
Преподаватель: Тюкавина Татьяна Михайловна.
МБОУ ДО Гатчинский Центр Непрерывного Образования
«Центр Информационных Технологий»

π - математическая константа, равная отношению длины окружности к длине её диаметра. Обозначается буквой греческого алфавита **π.** Старое название **— *лудольфово число.***
См. ПИ – википедия
Впервые обозначением этого числа греческой буквой воспользовался британский математик Джонс (1706), а общепринятым оно стало после работ Леонарда Эйлера в 1737. Это обозначение происходит от начальной буквы греческих слов περιφέρεια — окружность, периферия и περίμετρος — периметр. Очень интересно про историю этой константы и способы измерений изложены в wikia и в Словари и энциклопедии на Академике



1. Алгоритм Бюффона для определения числа Пи

Случайные величины использовались для решения прикладных задач достаточно давно. Примером может служить способ определения числа Пи, который был предложен Бюффоном еще в 1777 году. См. Википедия

2. Для написания программы используем Метод статистического моделирования, или метод **Монте-Карло.** (Паньгина Н.Н, Паньгин А.А. –Школа современного программирования.)

Методы Монте-Карло, ММК — общее название группы численных методов, основанных на получении большого числа реализаций случайного процесса, который формируется таким образом, чтобы его вероятностные характеристики совпадали с аналогичными величинами решаемой задачи. Используется для решения задач в различных областях физики, химии, математики, экономики, оптимизации, теории управления и др.

Метод получил свое название по имени города Монте-Карло в княжестве Монако, известного своими игорными заведениями , в которых публика растрачивает или увеличивает свои доходы, согласно *законам распределения случайных величин*. Дело в том, что одним из механических приборов для получения случайных величин является рулетка.

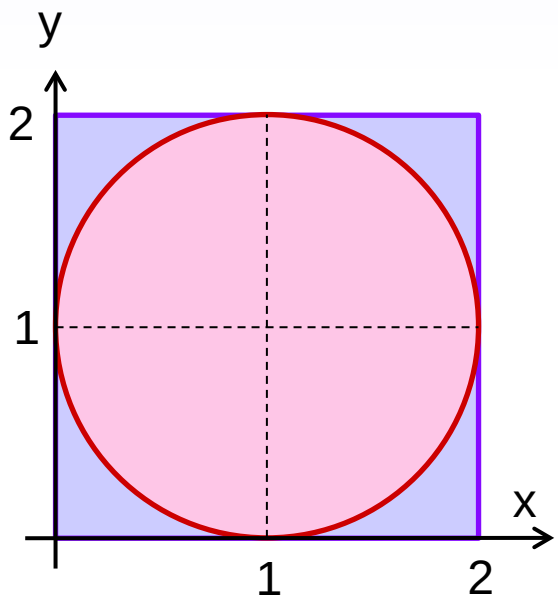
Метод Монте-Карло

Это метод статистических испытаний – численный метод, основанный на моделировании случайных величин и построении статистических оценок для искомых величин.

Метод статистического моделирования, или метод **Монте-Карло** используется для *моделирования игровых вероятностных ситуаций* (бросание монеты, кубика, блуждания), а также для *вычисления площадей фигур неопределенной формы*.

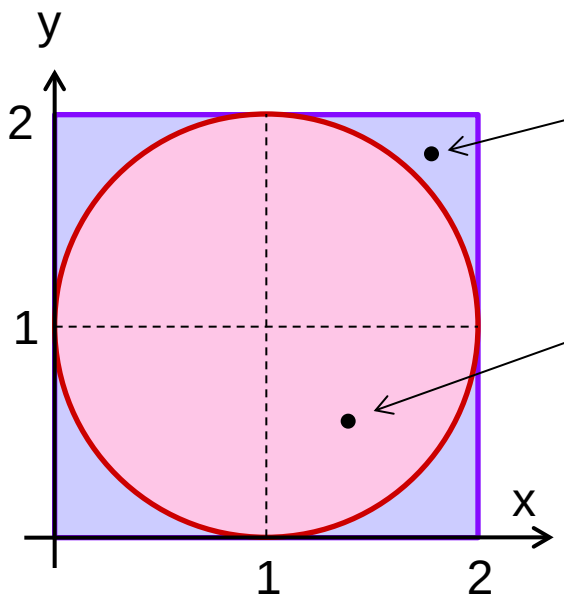
Для вычисления площади некоторой фигуры проведем эксперимент: поместим фигуру в квадрат и будем случайным образом бросать точки в этот квадрат. Чем больше площадь фигуры, тем чаще в нее будут попадать точки. Таким образом, можно сделать допущение: при большом числе точек, наугад выбранных внутри квадрата, доля точек, содержащихся в данной фигуре, приближенно равна отношению площади этой фигуры и площади квадрата.

С появлением ЭВМ этот метод несложно реализовать.



Рассмотрим имитационное моделирование на примере

Вычисление числа π = 3,1415922653…методом Монте-Карло.



Точка принадлежит квадрату, если:
$0 \le x \le 2$ и $0 \le y \le 2$

Точка принадлежит кругу, если:
$(x-1)^2 + (y-1)^2 \le 1$

Докажите это неравенство, решив геометрическую задачу.

Эксперимент: бросание случайных точек

m – число точек, попавших в круг

n – общее число точек

Тогда: $\frac{S_0}{S_s} = \frac{m}{n}$ (2)

Площадь круга : $S_0 = \pi r^2$

Площадь квадрата : $S_s = 4r^2$

Тогда: $\frac{S_0}{S_s} = \frac{\pi}{4}$ (1)

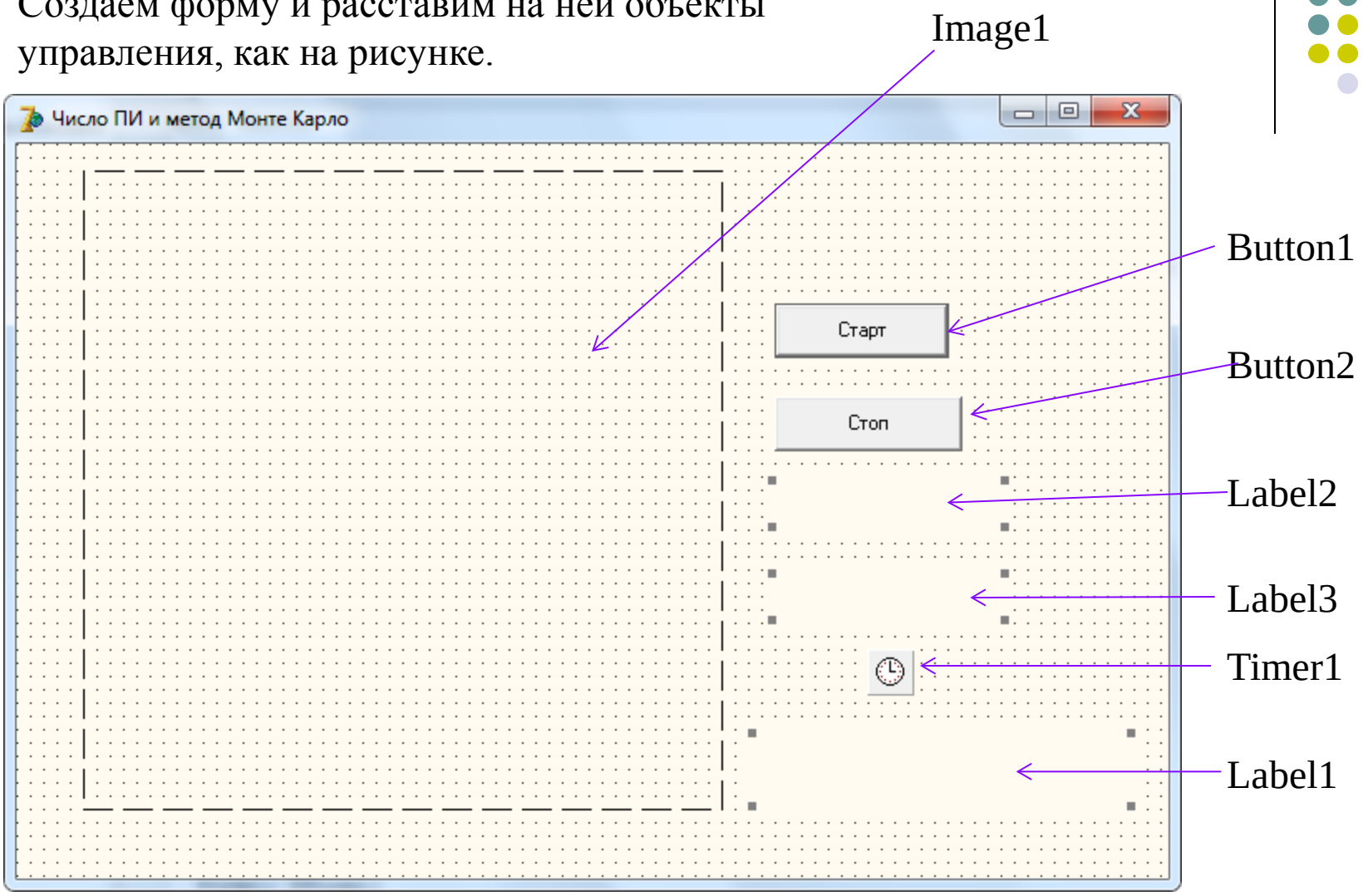
Из выражения (1) и (2) следует:

$$\pi = \frac{4 \cdot m}{n} \quad (3)$$

Программа в среде Delphi

I. Визуальное программирование.

1. Создаем форму и расставим на ней объекты управления, как на рисунке.



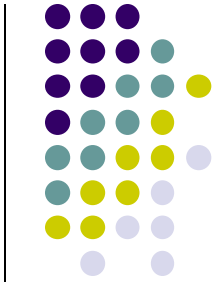
2. На объекте Image1 (Окно изображения) будем рисовать квадрат и вписанный в него круг.
   Задаем в Object Inspector свойства :
   Autosize = False
   Picture = None т.е. готовую картинку не загружаем, т.к. рисовать квадрат и вписанный в него круг будем в программе, используя соответствующие методы Окна изображения.
   Stretch = False
   Visible = True
   Width и Height должны *быть равными* для рисования квадрата

3. На форму помещаем невидимый объект управления Timer1 для задания скорости «разбрасывания» точек в случайное место квадрата. При конструировании программы задаем в Object Inspector свойства:
   Interval = 1 (что означает 1 миллисекунда)
   Enabled = False (т.е. в исходном состоянии часики стоят)

4. Для меток Label1, Label2, Label3 в Object Inspector задаем свойство Caption пустым, т.к. значения получим и выведем в программе.
   Label1 – в эту метку выводим вычисленное значение числа
   Label2 – в эту метку выводим число точек m, попавших в круг
   Label3 – в эту метку выводим общее число точек n, т.е. попавших в квадрат

5. Разместим на форме 2 командные кнопки с соответствующим свойством Caption «Старт» и «Стоп». При нажатии на кнопку «Старт» включается Timer1 и квадрат начинает заполняться случайными точками, причем точки, попавшие внутрь круга, изображаются синими. При нажатии на кнопку «Стоп» Timer1 выключается.

II. Написание кода программы.



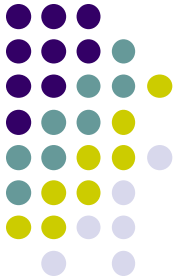
1. В данном приложении будут использоваться переменные, обозначающие общее количество точек n, число точек, попавших внутрь круга m, число ПИ, следовательно, их следует описать в общем разделе объявления переменных.

```
var
    Form1: TForm1;
    n, m :integer; { общее количество точек и число точек, попавших внутрь круга }
    x1, y1, x2, y2:integer; {угловые координаты квадрата в Окне изображения Image1}
    x0, y0, r :real; { координаты центра квадрата и вписанной в него окружности радиусом r}
    pi:real; { число ПИ }
```

2. В событийной процедуре загрузки формы необходимо включить генератор случайных чисел.

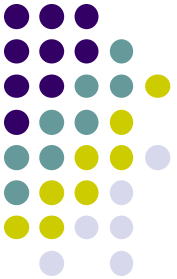
```
procedure TForm1.FormCreate(Sender: TObject);
begin
 Randomize;
end;
```

3. В событийной процедуре Щелчок на Кнопку «Старт» необходимо запустить Timer1. В окне изображения Image1 на свойстве Canvas с помощью графических методов нарисовать квадрат со стороной, равной ширине окна изображения и нарисовать круг в центре окна с радиусом, равным половине окна.



```
procedure TForm1.Button1Click(Sender: TObject);
begin
  timer1.Enabled := true;
  x1 := 0;
  y1 := 0;
  x2 := image1.ClientWidth;
  y2 := image1.ClientHeight;
  image1.Canvas.Pen.Color := clRed; {красный контур квадрата}
  image1.Canvas.Pen.Width := 3;
  image1.Canvas.Rectangle (x1, y1, x2, y2);
  image1.Canvas.Pen.Color := clBlue; {синий контур круга}
  image1.Canvas.Ellipse (x1, y1, x2, y2);
  r:=(x2 - x1) / 2; {радиус круга в половину стороны квадрата}
  x0:=x2  / 2;
  y0:=y2 / 2;
end;
```

4. Каждуб миллисекунду срабатывает событие Timer1, в результате которого должна быть «брошена» в квадрат случайная точка красного цвета с координатами x, y. Таким образом, счетчик точек (n) увеличивается на 1, формируются случайные (внутри квадрата) координаты точки и проверяется принадлежность этой точки вписанному в квадрат кругу. Если точка попадает в круг, то возрастает на 1 другой счетчик (m). Точки внутри круга для наглядности изображаются другим (синим) цветом.

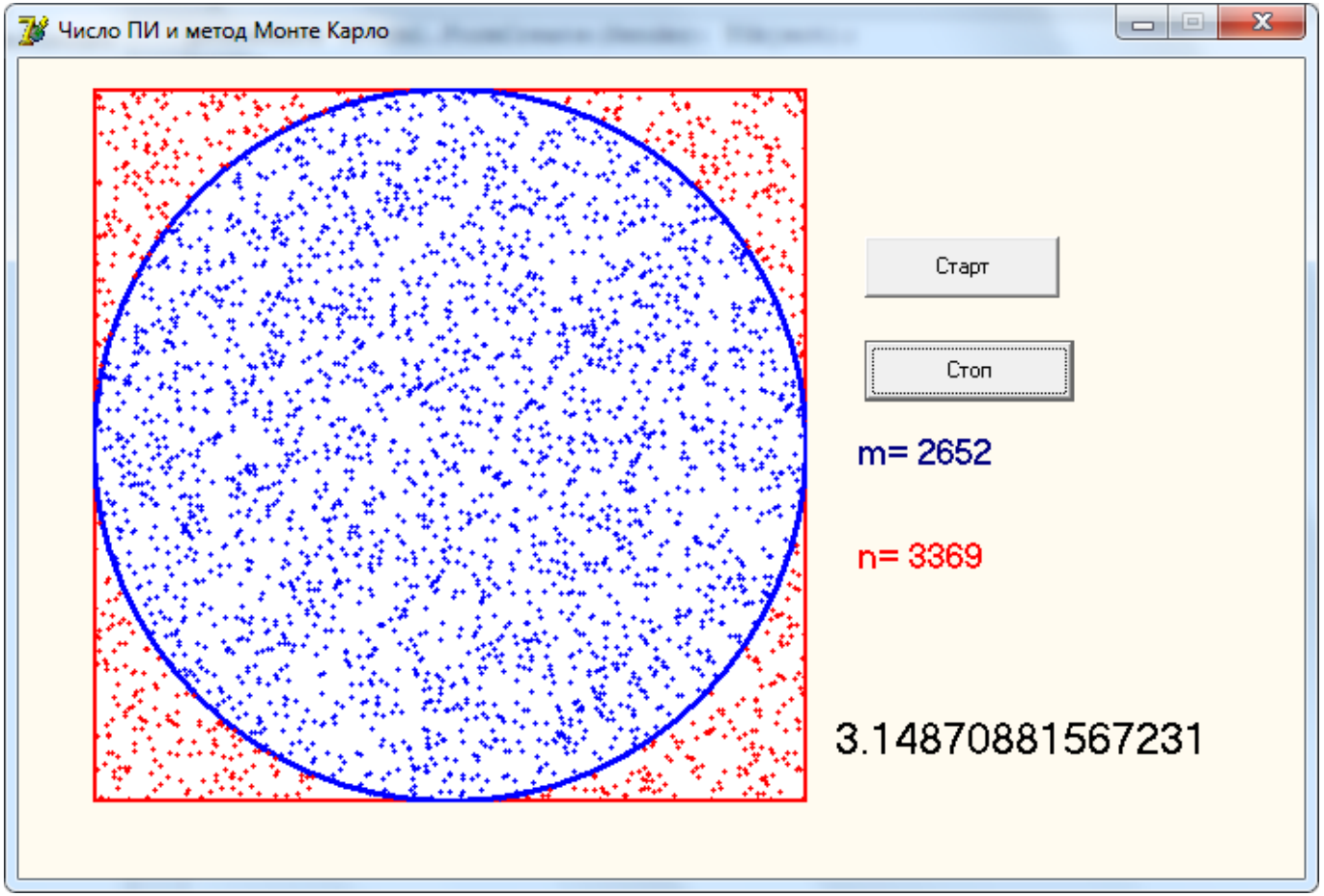


```
procedure TForm1.Timer1Timer(Sender: TObject);
var
  x, y :integer; { координаты случайной точки }
begin
  n:=n+1; { общее число точек, попавших в квадрат}
  x:=random(x2+1);
  y:=random(y2+1);
  image1.Canvas.Pen.Color:=clRed;
  if ((x-x0)*(x-x0)+(y-y0)*(y-y0))< r*r then { проверка условия принадлежности точки кругу }
  begin
    m:=m+1;
    image1.Canvas.Pen.Color:=clBlue;
  end ;
  image1.Canvas.Ellipse(x, y, x+1, y+1); { рисование случайной точки=кружочку радиусом 1}
  pi:=4*m/n; { вычисление и вывод числа ПИ в каждый момент, для наглядности процесса }
  label1.Caption:=FloatToStr(pi);
  label2.Caption:='m= '+IntToStr(m); {вывод числа m в каждый момент, для наглядности процесса }
  label3.Caption:='n= '+IntToStr(n);   {вывод числа n в каждый момент, для наглядности процесса }
end;
```

5. В событийной процедуре Щелчок на Кнопку «Стоп» необходимо выключить Timer1 и вычислить приближенное значение числа ПИ, согласно выведенной выше формуле (3)

```
procedure TForm1.Button2Click(Sender: TObject);
begin
   timer1.Enabled:=False;
   pi:=4*m/n;
   label1.Caption:=FloatToStr(pi); {вычисление и вывод числа ПИ }
end;
```

Результат статистического моделирования вычисления числа π в течении нескольких секунд работы программы представлен на рисунке.



Ресурсы:

При разработки этого примера была использована статья Паньгиной Нины Николаевны и Паньгина Андрея Александровича – «Статистическое моделирование: метод Монте-Карло.»
Школа современного программирования 2002г.