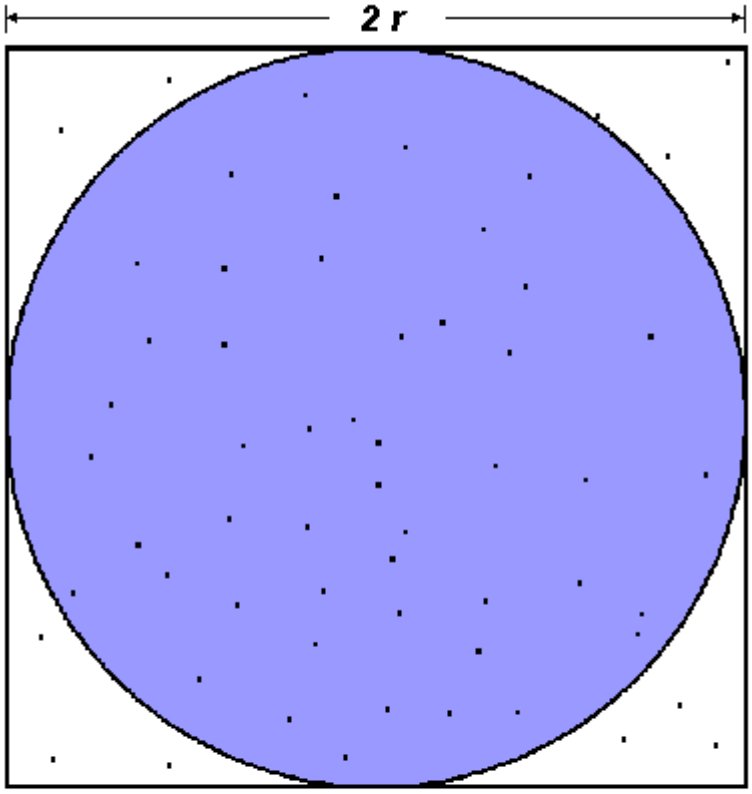
Зад. 6 (Пресмятане на Pi)

Числото (стойността на) Pi може да бъде изчислена по различни начини. Разглеждаме следния метод за приближено пресмятане на Pi:

1) Имаме окръжност, вписана в квадрат;
2) Генерираме по произволен начин точки в квадрата;
3) Определяме броя на точките, които се намират в окръжността;
4) Нека **`k`** е число равно на броя на точките в окръжността, разделен на броя на точките в квадрата;
5) Тогава **`Pi ~ 4 * k`**;

Бележка: Колкото повече точки генерираме толкова по-добра ще бъде точността с която пресмятаме Pi.

Визуално, можем да представим метода със следната картинка:



$$A_S = (2r)^2 = 4r^2$$

$$A_C = \pi r^2$$

$$\pi = 4 \times \frac{A_C}{A_S}$$

Разгледаме сериен псевдокод реализиращ този метод:

```
=== cut ===
npoints = 10000
circle_count = 0

do j = 1,npoints
  generate 2 random numbers between 0 and 1
  xcoordinate = random1
  ycoordinate = random2
  if (xcoordinate, ycoordinate) inside circle
  then circle_count = circle_count + 1
end do

PI = 4.0*circle_count/npoints
=== cut ===
```

Както се вижда от кода, по-голямата част от времето за изпълнение на програмата ще премине в изпълнение на цикъла.

Вашата задача е да напишете програма за изчисление на числото Pi. по-описания метод, която използва паралелни процеси (нишки). Изискванията към програмата са следните:

(o) Размерността на квадрата се задава в точки (пиксели) от подходящо избран команден параметър – например **"-s 10240"**. Точките в квадрата, генерираме произволно с помощта на **Math.random()** или класа **java.util.Random;**

(o) Друг команден параметър задава максималния брой нишки (задачи) на които разделяме работата по пресмятането на Pi – например **"-t 1"** или **"-tasks 3";**

(o) Програмата извежда подходящи съобщения на различните етапи от работата си, както и времето отделено за изчисление и резултата от изчислението (стойността на Pi);

(o) Да се осигури възможност за „quiet" режим на работа на програмата, при който се извежда само времето отделено за изчисление на Pi (и самото число) , отново чрез подходящо избран друг команден параметър – например **"-q";**

ЗАБЕЛЕЖКА:

(o) При желание за направата на подходящ графичен потребителски интерфейс (GUI) с помощта на класовете от пакета **javax.swing** задачата може да се изпълни от **двама души**;

(o) Задачата може да се реши и с помощта на RMI (**java.rmi**). За целта трябва да се помисли за разпределения достъп до общия ресурс в случая квадрата съдържащ случайно генерираните точки;

NOTE(s):

(o) В условието на задачата се говори за разделянето на работата на две или повече нишки. Работата върху съответната задача на една нишка ще служи за еталон, по който да измерваме евентуално ускорение. Тоест в кода реализиращ решенията на задачите трябва да се предвиди и тази възможност – задачата да бъде решавана от единствена нишка (процес); Пускайки програмата да работи върху задачата с помощта на единствена нишка, ще считаме че използваме серийното решение на задачата;