Лабораторная работа по графике №2, 26 декабря

В этой работе мы продолжим учиться работать с графикой в python. Теперь не будет поясняющих примеров. Вам до всего нужно будет догадаться самостоятельно :-) Правила такие же, как и в предыдущей контрольной работе. Вам дается код программы, вы его набираете, смотрите, что получается и комментируете каждую строчку. После выполнения присылаете в **одном файле** все упражнения на почты *dontwritehim@gmail.com* и *i_khisambeev@mail.ru*.

1 Первое упражение — гистограмма

```
import matplotlib.pyplot as plt
import numpy as np

y = np.random.randn(1000)

plt.hist(y, 25)
plt.show()
```

Вам нужно догадаться, что делает функция *randn*. Это не просто случайные числа. Можете попробовать заполучить подсказку у Ильдара :-)

2 Второе упражение: учет ошибок

В практической науке, эксперимент даже при максимальной точности измерений всегда вносит свою погрешность. Для того, чтобы учесть это и указать возможный разброс вокруг значения, считаемого истинным, вводят error bars, которые на кривой для полученных точек показывают своеобразный доверительный интервал:

```
import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0, 4, 0.2)
y = np.exp(-x)
e1 = 0.1 * np.abs(np.random.randn(len(y)))

plt.errorbar(x, y, yerr=e1, fmt='.-')
plt.show()
```

Особенно тщательно разберитесь с тем, как работает *np.arange*. Можно воспользоваться гуглом ;-)

3 Третье упражнение: круговая диаграмма

Такой вид диаграмм используется, когда для нас существенно сравнение вклада каждого участника в общее дело. Поскольку это дело похоже на разрезание пирога, то и диаграммы также называют pie charts, а функцию pie(). Первым аргументом она принимает x-последовательность из внесенных единиц.

```
import matplotlib.pyplot as plt

plt.figure(figsize=(7,5))
x = [18, 15, 11, 9, 8, 6]
labels = ['Java', 'C', 'C++', 'PHP', '(Visual) Basic', 'Python']
explode = [0, 0, 0, 0, 0, 0.2]

plt.pie(x, labels = labels, explode = explode, autopct = '%1.1f%%', shadow=True);
plt.show()
```

Кстати, эта статистика уже давно неверная.

4 Четвертое упражнение: график рассеяния

Такой тип графиков позволяет изображать одновременно два множества данных, которые не образуют кривой, а именно двухмерное множество точек. Каждая точка имеет две координаты. График рассеяния часто используется для определения связи между двумя величинами и позволяет определить более точные пределы измерений.

```
import matplotlib.pyplot as plt
import numpy as np

x = np.random.randn(1000)
y = np.random.randn(1000)

size = 50*np.random.randn(1000)
colors = np.random.rand(1000)

plt.scatter(x, y, s=size, c=colors)
plt.show()
```

5 Пятое упражнение: полярные координаты

Кроме наиболее часто используемой декартовой системы координат, довольно широко применяется и полярная система координат, удобная в различных радиальных задачах, координаты точек в ней задается с помощью радиус-вектора ρ, идущего из начала координат и угла θ. Угол может быть задан в радианах или градусах, matplotlib использует градусы.

```
import matplotlib.pyplot as plt
import numpy as np

theta = np.arange(0., 2., 1./180.)*np.pi

plt.polar(3*theta, theta/5)
plt.polar(theta, np.cos(4*theta))
plt.polar(theta, [1.4]*len(theta))

plt.show()
```