Jonah Malriat
Dice Simulation: Explanation & Reflection
Introduction to Problem Solving & Programming – Python
10/11/2019

To find the probability of an event, this formula is needed.

$$P(x) = \frac{\textit{Number of intended outcomes}}{\textit{Number of total outcomes}}$$

The function *P(x)* is the probability of a specific intended outcome. We'll represent the "Number of intended outcomes" with the variable *x*, and the "Number of total outcomes" with the variable *y*. Therefore $P(x) = x/y$ and I can set up my proof as to why rolling two dice is different from rolling one die with sides containing the numbers 2 – 12.

<u>For two dice:</u>

For each die, the probability of rolling a single intended number is 1 / 6 or $16.\overline{6}\%$. This is because there is one intended outcome, for example rolling the number 2. You cannot roll 2 and another number at the same time, so x = 1. There are 5 other possible outcomes besides this, so the total number of outcomes is y = 6. Once another die is added things become more complicated, so I will demonstrate with a table.

Die Values	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

Fig. 1 – Addition of two dice

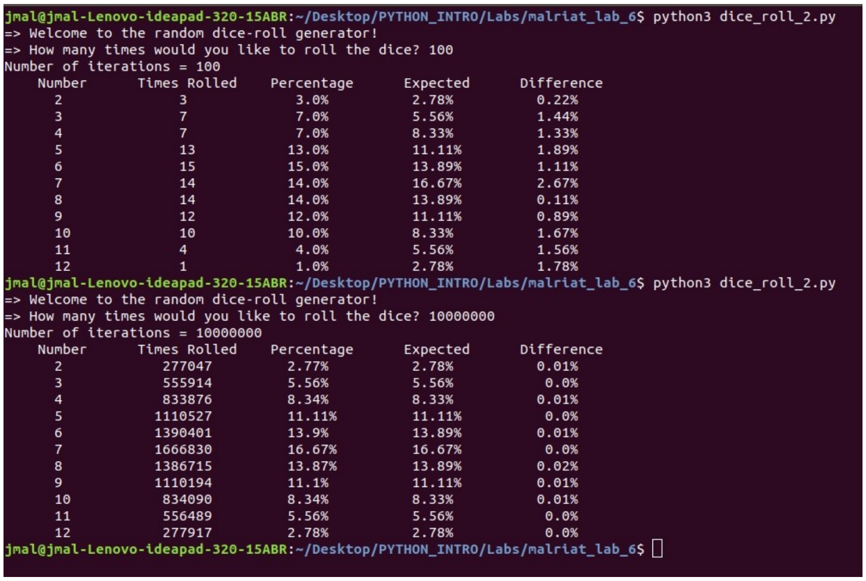
2 = 1/36 or $2.\overline{7}\%$
3 = 1/18 or $5.\overline{5}\%$
4 = 1/12 or $8.\overline{3}\%$
5 = 1/9 or $11.\overline{1}\%$
6 = 5/36 or $13.\overline{8}\%$
7 = 1/6 or $16.\overline{6}\%$
8 = 5/36 or $13.\overline{8}\%$
9 = 1/9 or $11.\overline{1}\%$
10 = 1/12 or $8.\overline{3}\%$
11 = 1/18 or $5.\overline{5}\%$
12 = 1/36 or $2.\overline{7}\%$

Fig. 2 – Probability of roll values

As you can see in figure 1, there is only one combination of dice rolls that will result in an outcome of 2, therefore *x* = 1. The total number of outcomes with two dice would be 36, shown in figure 1 by all the non-highlighted values, so *y* = 36. This results in the probability of rolling both dice and getting an outcome of 2 being 1/36 or $2.\overline{7}\%$. After doing the math for every value from 2 – 12, the pattern depicted in figure 2 was noticed with the outcome of 7 being the most common.

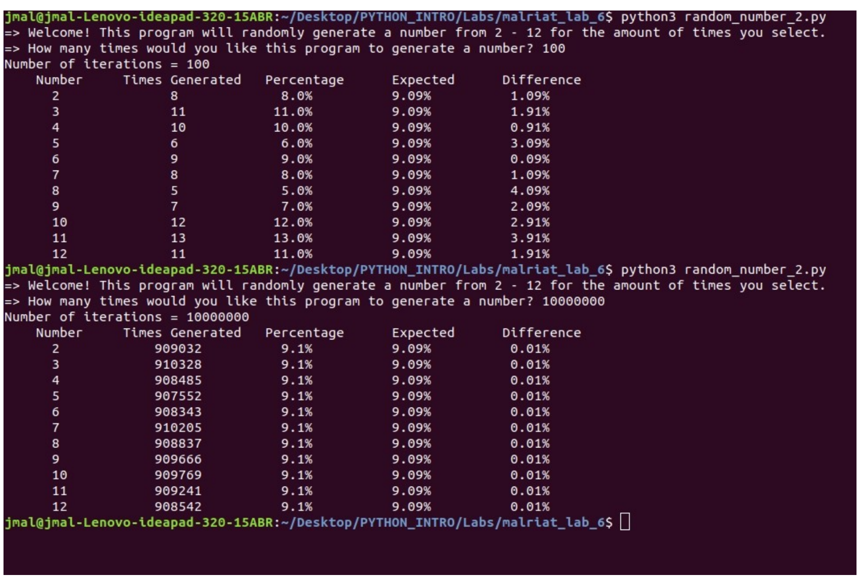
<u>For one die:</u>

When this same formula is applied to one die with 11 sides labeled 2 – 12, the probability completely changes. A table is not required to display this relationship, because all the outcomes have an equal probability. This is because every side has one intended outcome, and no more than one outcome can occur at the same time. Therefore *x* = 1 for each individual side of the die, and because there are 11 sides *y* = 11. Every side will then have the probability of 1/11 or $9.\overline{09}\%$. This is why the two methods, although dealing with the same outcomes of 2 – 12, have completely different probabilities.



```
jmal@jmal-Lenovo-ideapad-320-15ABR:~/Desktop/PYTHON_INTRO/Labs/malriat_lab_6$ python3 dice_roll_2.py
=> Welcome to the random dice-roll generator!
=> How many times would you like to roll the dice? 100
Number of iterations = 100
    Number      Times Rolled     Percentage      Expected       Difference
      2              3              3.0%          2.78%          0.22%
      3              7              7.0%          5.56%          1.44%
      4              7              7.0%          8.33%          1.33%
      5              13            13.0%         11.11%          1.89%
      6              15            15.0%         13.89%          1.11%
      7              14            14.0%         16.67%          2.67%
      8              14            14.0%         13.89%          0.11%
      9              12            12.0%         11.11%          0.89%
      10             10            10.0%          8.33%          1.67%
      11             4              4.0%          5.56%          1.56%
      12             1              1.0%          2.78%          1.78%
jmal@jmal-Lenovo-ideapad-320-15ABR:~/Desktop/PYTHON_INTRO/Labs/malriat_lab_6$ python3 dice_roll_2.py
=> Welcome to the random dice-roll generator!
=> How many times would you like to roll the dice? 10000000
Number of iterations = 10000000
    Number      Times Rolled     Percentage      Expected       Difference
      2           277047          2.77%          2.78%          0.01%
      3           555914          5.56%          5.56%           0.0%
      4           833876          8.34%          8.33%          0.01%
      5          1110527         11.11%         11.11%           0.0%
      6          1390401          13.9%         13.89%          0.01%
      7          1666830         16.67%         16.67%           0.0%
      8          1386715         13.87%         13.89%          0.02%
      9          1110194          11.1%         11.11%          0.01%
      10          834090          8.34%          8.33%          0.01%
      11          556489          5.56%          5.56%           0.0%
      12          277917          2.78%          2.78%           0.0%
jmal@jmal-Lenovo-ideapad-320-15ABR:~/Desktop/PYTHON_INTRO/Labs/malriat_lab_6$
```

Fig. 3 – Test of dice roll program



```
jmal@jmal-Lenovo-ideapad-320-15ABR:~/Desktop/PYTHON_INTRO/Labs/malriat_lab_6$ python3 random_number_2.py
=> Welcome! This program will randomly generate a number from 2 - 12 for the amount of times you select.
=> How many times would you like this program to generate a number? 100
Number of iterations = 100
    Number     Times Generated     Percentage       Expected       Difference
      2              8               8.0%           9.09%           1.09%
      3              11             11.0%           9.09%           1.91%
      4              10             10.0%           9.09%           0.91%
      5              6               6.0%           9.09%           3.09%
      6              9               9.0%           9.09%           0.09%
      7              8               8.0%           9.09%           1.09%
      8              5               5.0%           9.09%           4.09%
      9              7               7.0%           9.09%           2.09%
      10             12             12.0%           9.09%           2.91%
      11             13             13.0%           9.09%           3.91%
      12             11             11.0%           9.09%           1.91%
jmal@jmal-Lenovo-ideapad-320-15ABR:~/Desktop/PYTHON_INTRO/Labs/malriat_lab_6$ python3 random_number_2.py
=> Welcome! This program will randomly generate a number from 2 - 12 for the amount of times you select.
=> How many times would you like this program to generate a number? 10000000
Number of iterations = 10000000
    Number     Times Generated     Percentage       Expected       Difference
      2           909032             9.1%           9.09%           0.01%
      3           910328             9.1%           9.09%           0.01%
      4           908485             9.1%           9.09%           0.01%
      5           907552             9.1%           9.09%           0.01%
      6           908343             9.1%           9.09%           0.01%
      7           910205             9.1%           9.09%           0.01%
      8           908837             9.1%           9.09%           0.01%
      9           909666             9.1%           9.09%           0.01%
      10          909769             9.1%           9.09%           0.01%
      11          909241             9.1%           9.09%           0.01%
      12          908542             9.1%           9.09%           0.01%
jmal@jmal-Lenovo-ideapad-320-15ABR:~/Desktop/PYTHON_INTRO/Labs/malriat_lab_6$
```

Fig. 4 – Test of single die roll program

As the number of iterations increased in the dice roll program, the actual percentage became closer and closer to the expected value. This is represented in figure 3. The same result occurred in the single die roll program. As the number of iterations increased, the actual percentage became increasingly closer to the expected value. This is represented in figure 4. If this same experiment would be attempted with real dice, it would take an unreasonable amount of time such as weeks, months, or years of constant rolling. However, if this was attempted with real dice the results could be different. This would be because there are too many factors in real-life compared to the program. Factors such as table angle, dice with imperfect weights, how the dice are rolled, or even wind could affect results. Occasionally even the simulation would produce results significantly different from the results in figure 2. This was usually caused by not having a large enough amount of iterations as shown in the top of figures 3 and 4.

Seeing this program operate does provide some insight into how other models of real world phenomena are produced. While this simulation is small and insignificant to daily life, probabilities are used daily in models of finance, biology, medicine, sports, weather, etc. All these fields need probability and similar models in order to function as accurately as possible.

In attempting to solve this problem I struggled with making the program roll the dice repeatedly. I could easily make a program that rolled two dice and added the values, but attempting to add the values and store them repeatedly gave me an issue. I eventually realized I could store the values of each roll into a list after looking back at the textbook, this provided easy storage and access for each sum. I later had a logical error where my percentages did not add up to 100%. This took time to correct, but helped me understand both the program and probability itself more in-depth.