Sam Chen - 013502214

EE 381

Lab 1

9-18-20

1. Function for a n-sided die

Write a function that simulates a single roll of a n-sided die. The inputs and outputs of the function are:

Inputs:

- The probabilities for each side, given as a vector $p = [p_1, p_2, \cdots p_n]$

Outputs:

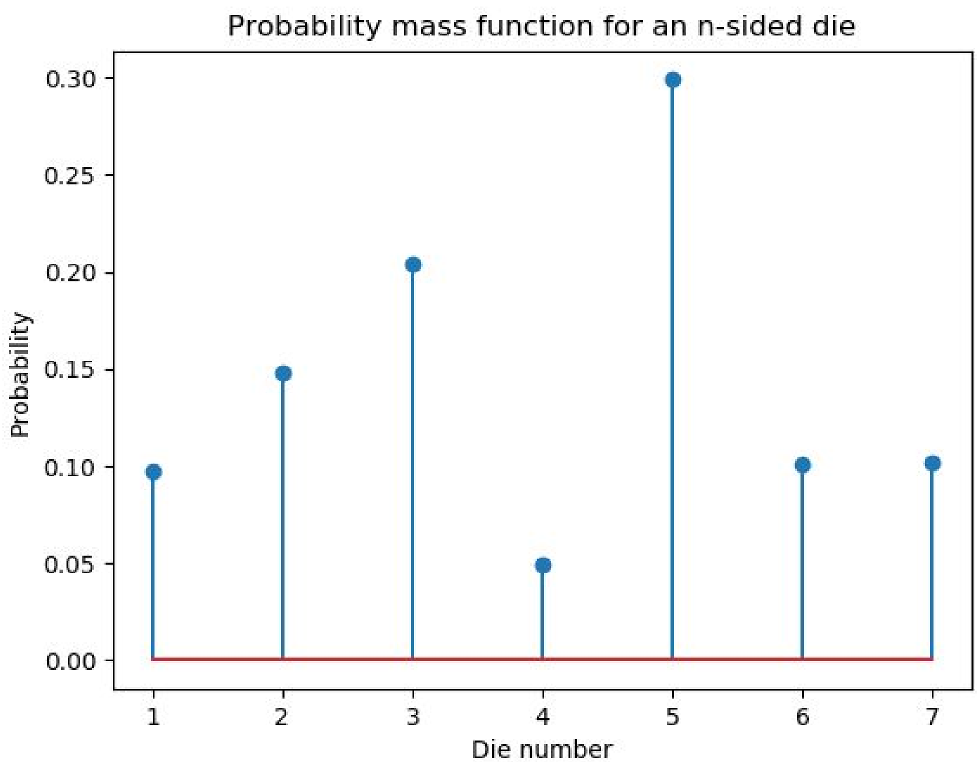
- The number on the face of the die after a single roll, i.e. one number from the set of integers $\{1, 2, \cdots n\}$

Note: The sum $p_1 + p_2 + \cdots p_n$ must be equal to 1.0, otherwise the probability values are incorrect.

Save the function as: `nSidedDie(p)`

Test the function using the probability vector $p = [p_1, p_2, \cdots p_n]$ which has been given to you. To create a random number with a single roll of the die you must use the following command: `r=nSidedDie(p)`

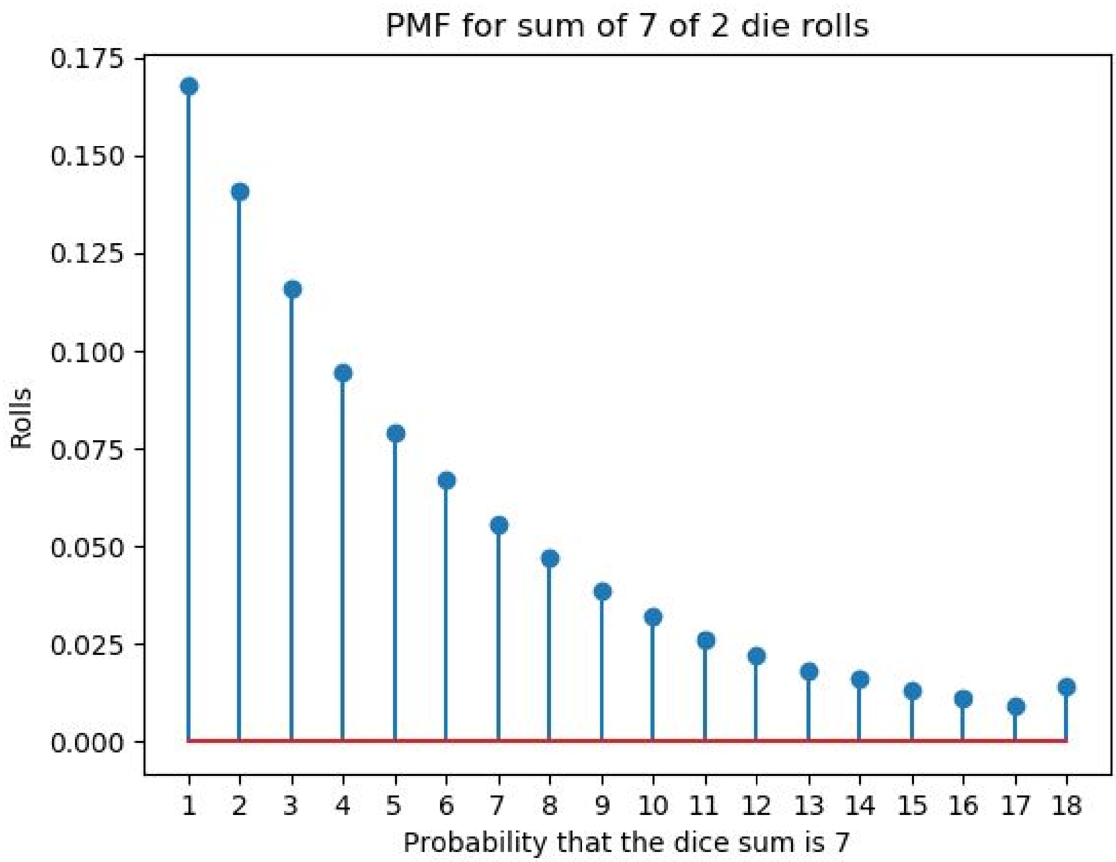
To validate your function, roll the die for N=10,000 times and plot the outcome as a stem plot.



2. Number of rolls needed to get a "7" with two dice

Consider the following experiment:

- You roll a pair of fair dice and calculate the sum of the faces. You are interested in the number of rolls it takes until you get a sum of "7". The first time you get a "7" the experiment is considered a "success". You record the number of rolls and you stop the experiment.
- You repeat the experiment N=100,000 times. Each time you keep track of the number of rolls it takes to have "success".



3. Getting 50 heads when tossing 100 coins

Consider the following experiment:

- You toss 100 fair coins and record the number of "heads". This is considered a single experiment. If you get exactly 50 heads, the experiment is considered a "success".
- You repeat the experiment N=100,000 times. After the N experiments are completed count the total successes, and calculate the probability of success, i.e. the probability of getting exactly 50 heads.

RESULTS:

Probability of 50 heads in tossing 100 fair coins	
Ans.	**p = 0.0793**

4. The Password Hacking Problem

Your computer system uses a 4-letter password for login. For our purposes the password is restricted to lower case letters of the alphabet only. It is easy to calculate that the total number of passwords which can be produced is $n = 26^4$.

- A hacker creates a list of m random 4-letter words, as candidates for matching the password. Note that it is possible that some of the m words may be duplicates. The number m that you must use has been given to you.
- You are given your own 4-letter password and you are going to check if the hacker's list contains at least one word that matches your password. This process of checking is considered one experiment. If a word in the list matches your password, the experiment is considered a success. Repeat the experiment for $N = 1000$ times and find the probability that **at least one of the words** in the hacker's list will match your password.
- The hacker creates a longer list of k*m random 4-letter words. The numbers k and m have been given to you. Repeat the previous experiment for $N = 1000$ times and find the probability that **at least one of the words** in the hacker's list will match your password.
- Repeat the previous experiment for $N = 1000$ times to find the **approximate number** (m) of words that must be contained in the hacker's list so that the probability of at least one word matching the password is $p = 0.5$. You should do this by trial and error: assume a value for (m) and calculate the corresponding probability as you did in the previous part. The answer will be value of (m) that makes this probability **approximately equal** to $p = 0.5$.

RESULTS:

Hacker creates m words Probability that at least one of the words matches the password	**p = 0.172**
Hacker creates k*m words Probability that at least one of the words matches the password	**p = 0.725**
P = 0.5 Approximate number of words in the list	**m = 0.505**