Assumptions

·       We consider the birthday distribution over a year so we have possible values from 1 to 365

·       Real statistics are taken from the web

·       m (number of people considered) < 365

·       To run different experiments, we use different seeds

2.       Input parameters

·       m

3.       Output metrics

·       average number of people to observe a conflict and the comparison with the theoretical result.

·       Evaluate the probability of birthday conflict in function of m and the comparison with the theoretical result.

·       Correlated graphs

4.       Main data structures

·       Array representing 365 days of the year, value at index i represent the number of people with birthday at day i.

5.       Main algorithm

·       We slightly differ the algorithm depending what we need to compute ( mean of number people or probability)

·       In both cases we run differen experiments

·       Case: find the average number of people

                                                               i.      Compute theoretical result

                                                             ii.      Extract randomly (uniformly or based on real distribution) a day (index) for the birthday, update a counter C used to track the number of inserted people

                                                           iii.      Update the array in the corresponding index

                                                           iv.      Loop this till we find a birthday conflict (i.e. array has already updated in that position)

                                                             v.      When find a conflict store the counter C and compute the average between all the experiments.

·       Case: evaluating the probability of birthday conflict in function of m

                                                               i.      Compute theoretical result

                                                             ii.      For each people in m, extract randomly (uniformly or based on real distribution) a day (index) for the birthday

                                                           iii.      Update the array in the corresponding index

                                                           iv.      Loop till we finish m

                                                             v.      Run different experiments and find the probability to have conflicts

·       For both cases plot the comparison between empirical and theoretical results

6.       Possible extensions

·       Consider the generalized problem

Assumptions:

* m < n
* in the case of the birday paradox n = 366 (don't consider leap years)

Input parameters:

* m
* n
* k: the number of runs
* type of distribution of the birthday:

1. Uniform distribution
2. Realistic distribution

Output metrics:

* E[m]: average number of people to observe a conflict
* C.I. of E[m]
* p(m): probability of a birthday conflict
* C.I. of p(m)

Main data structures:

* list to collect the people already in class
* dictionary to collect the number of person to get a conflict  (k experiments to then compute the average and C.I.)
* dictionary to collect the values of p(m) (one for each random seed)

Main algorithms:

2 different algorithms:

* E[m]

    for k times:

        generate a number from the distribution

        add the number to the list of birthdays in the classroom

        if conflict repeat

    compute average value over the k runs

* p(m)

    for each random seed:

        take k class of size m

        for each class check if there is conflict or no

        p = class with conflict / k

Possible extensions:

* generalize the model, having a parameter n instead of 366
* compute statistics non only in function of m but also as a function o n

**ASSUMPTIONS**:

- birthdays during the year are uniformally distributed

-  birthdays during the year follows a realistic distribution

- m < n

**INPUT PARAMETERS:**

- m is the number of people taken into account

- n = 365 is the cardinality of the set

**OUTPUT METRICS:**

- average number of people to observe a conflict

- probability of birthday conflict in function of m

**MAIN ALGORITHMS:**

**Evaluate the average number of people to observe a conflict and compare with the theoretical result.**

- generate birthdays until you reach a conflict. when you find the conflict, store the the number of birthdays needed

- repeat it k times

- compute the empirical average

- compare it with the theoretical formula

**Evaluate the probability of birthday conflict in function of m and compare with the theoretical result.**

repeat k times:

* fix m (different value each time)
* check if you have a conflict
* compute the probablity as the number of classes I found a conflict / total number of classes

**POSSIBLE EXTENSIONS :**

Implementation of the generalized version with a generic n