TODO-Your Group Number and Names Here due: 18 October 2019

CSCI 432 Problem 3-1

Collaborators: *TODO-list your collaborators here*

If 23 people are in a room, then the probability that at least two of them have the same birthday is at least one half. This is known as the birthday paradox, since the number 23 is probably much lower than you would expect. How many people do we need in order to have 50% probability that there are three people with the same birthday? As a reminder, when giving an algorithm as an answer, you are expected to give:

- A prose explanation of the problem and algorithm.
- Psuedocode.
- The decrementing function for any loop or recursion, or a runtime justification.
- Justification of why the runtime is linear.
- The loop invariant for any loops, with full justification.

TODO-Your Group Number and Names Here		due: 18 October 2019
	CSCI 432 Problem 3-2	
Collaborators: *TODO-list your collaborators here*		

Suppose we have a graph $G = (V, E)$ and three colors, and randomly assign a color each node (where each color is equally likely).

1. What is the probability that every edge has two different colors on assigned to its two nodes?
2. What is the expected number of edges that have different colors assigned to its two nodes?

TODO-Your Group Number and Names Here due: 18 October 2019

CSCI 432 Problem 3-3

Collaborators: *TODO-list your collaborators here*

CLRS, Question 15-6.

TODO-Your Group Number and Names Here	due: 18 October 2019

CSCI 432 Problem 3-4

Collaborators: *TODO-list your collaborators here*

For the Greedy make change algorithm described in class on 10/02, describe the problem and solution in your own words, including the use of pseudocode (with more details than what was written in class). Note: you do not need to give a loop invariant and the proof of termination/runtime complexity.

TODO-Your Group Number and Names Here due: 18 October 2019

CSCI 432 Problem 3-5

Collaborators: *TODO-list your collaborators here*

Suppose we have n items hat we want to put in a knapsack of capacity W. The i-th item has weight w_i and value v_i. The knapsack can hold a total weight of W and we want to maximize the value of the items in the knapsack. The *0-1 knapsack problem* will assign each item one of two states: in the knapsack, or not in the knapsack. The *fractional knapsack problem* allows you to take a percentage of each item.

1. Give an $O(n \log n)$ greedy algorithm for the fractional knapsack problem.
2. Give an $O(nW)$ time algorithm that uses dynamic programming to solve the 0-1 knapsack problem.