Comp 305 Algorithms and
Spring 2021 Complexity

Final Project

1 Submission and Grading

Projects information will be stored Google Sheet and your progress will be followed via GitHub. Please open this link Google Sheet Link, and enter the reuired information in the google sheet. Please email to *agharelar20@ku.edu.tr* or *COMP305-staff group*, if there is any problem. You solution will be evaluated by some Test cases to measure both your algorithm's correctness and efficiency. Therefore, try to do your best and make your code efficient.

As a next step in the project settings, please open a GitHub public repository, include all of your members as contributors and add the repository link to the given Google Sheet link. This step is quite important for us to see your progress and has to be done quickly. Therefore, in the README file please keep a list of completed steps, a TO DO list and the results retrieved if there are any. You will also prepare a presentation and present to the TAs. Therefore, you should also create a Google Slides presentation and include its link to the Google Sheet document.

2 Presentation Details

Each of the presentations should take ∼10 minutes and there will be a 5-minute Q&A session afterwards. If a presentation lasts longer than 10 minutes, then it will be interrupted. During the presentation each of the groups should explain and report:

- The algorithm you designed to solve the problem, the choices of the data structures you used and your reasoning.
- The time complexity of your algorithm (and the space complexity if applicable).
- Your run times for each of the test cases.
- Further improvements that can be done as future works.

This project does not expect from you to come up with just one solution and then test only that solution. For each of the problems you can start with some baseline approaches with more complexity and improve the baseline algorithm step by step. Be as creative as possible. Report different approaches you tested and why did you decide on the final algorithm you present. Your grading will be based on your creativity, your cumulative progress and how well did you present your approach.

3 Deadlines

You can work on your project until the day of presentation. The project presentations will be held between ***24th-28th of May, 2021***.

In the following pages, you can see each of the available projects:

My Precious Bitcoins

Bitcoin is a cryptocurrency invented in 2008 by a unknown person. In the first day's of its introduction, Bitcoin literally worth nothing, however, as of March 2021, its value fluctuates around 50 thousands dollars.

Ali, who has been software engineer for the last 15 years, did an outsourced project for a company in 2010, but his wage was paid by 10000 Bitcoins, and he saved these Bitcoins in his digital wallet. Ali now potentially is a millioner, nonetheless, there is a small problem. He forgets the password of his wallet!
Fortunately, he remembers some information about the password which is as follow:

- The structure of the password is **(x,y,z)**, where **x**, **y** and **z** are none-negative integers
- He also knows the **x**, **y** and **z** satisfy the following equation, where **n** is a number that will be given to you:
  x+y+z=n

Our digital wallet gets triples **(x,y,z)** as an input, and it tells Ali whether the password candidate is correct or not. Unfortunately, there is a limitation about inputs of the digital wallet. We need to give a new **x**,**y** and **z** in each of our trials. For example, if we already tested **x** with the number 28, we could not use the value of 28 for **x** in the next attempts.

In this project you need to help Ali claims his money. To be more specific, you will find triples of **(x,y,z)** to feed them to the digital wallet. It is clear that you need to find the maximum number of candidates to boost the chance of success.

Input

The only input is **n**, ($1 \leq \mathbf{n} \leq 300$)

Output

In the first line, print a single integer denoting the number candidates, and in the following lines print out each candidates. In each line you only need to print one candidate.

Example

Input

```
3
```

Output

```
3
0 1 2
2 0 1
1 2 0
```

Deliveries

You are suppose to find efficient algorithm that can solve the question for any **n** lower than 300 in a reasonable time. You also need to make a document that clearly explain your approach and your reasoning for asymptotic complexity.