This assignment is **due on May 17** and should be submitted on Gradescope. All submitted work must be *done individually* without consulting someone else's solutions in accordance with the University's "Academic Dishonesty and Plagiarism" policies.

As a first step go to the last page and read the section: "Advice on how to do the assignment".

Important: *This assignment is for all COMP3x27 students. COMP3027 students should do Problems 1 and 2 while COMP3927 students should do Problems 1, 2,* and *3.*

Background. Space is big. Mostly empty, quite cold, but crucially very, very big. Which is a bit of a problem for you, since, as a space warlord with limited resources, it's a pain to keep track of your empire of, wait, oh many? Wow, n planets already. Good on you!

You're a very organised space warlord though, and have established space routes between each of these planets, taking into account the risk of space dingos, the refuelling capabilities, and the areas where you just don't want to fly through (the comets are not worth it and the view is truly lame). So you have a nice holomap of these n planets of yours, along with m directional wormlinks that connect them through some elaborate, yet incredibly functional network of space routes. A holomap, or, as your annoying lieutenant calls it, "a directed graph $G = (V, E)$ with n vertices and m edges."

Problem 1. (30 points)

Anyways, space is big, troops are not cheap and you would like to figure out if instead of having a small army on every planet, you could not just have a few space soldiers in a small number of selected planets (space bases!), able to easily keep peace and order in your space warlord empire: that is, find the minimum number b of your planets on which to establish a base, such that every planet is reachable from a base in one wormlink hop. Formally: "find the minimum integer b such that there exists $B \subseteq V$ of size $|B| = b$ satisfying: for every $v \in V \setminus B$, there is a base $u \in B$ such that $(u, v) \in E$."

That's really a mouthful. Unfortunately, since you just threw your lieutenant into space for being a tad too annoying, you have to do it yourself... his last words before being sent towards a space crocodile? "This is hopeless, it's NP-*aaaarg*."

Your task is to show he was right.

a) Describe a polynomial-time reduction from the above problem to the decision problem: "given an integer b, decide whether there exists $B \subseteq V$ of size $|B| = b$ satisfying: for every $v \in V \setminus B$, there is a base $u \in B$ such that $(u, v) \in E$."

b) Briefly argue the correctness of this reduction and analyse its running time.

c) Show this decision problem is in NP.

d) Show this decision problem is NP-Complete.

e) Conclude the original problem is NP-Hard. Is it NP-Complete?

Problem 2. (70 points for COMP3027 (40 points for COMP3927)) This was a bummer, especially since that lieutenant could have been helpful for this next step. Oh well. Along with the rest of your space advisors, you have identified k space quadrants $Q_1, \ldots, Q_k$ partitioning your space empire in strategic ways, and have decided to establish one space base in each of these space quadrants, to secure your space empire. (You really love *space*, don't you?)

Quadrant i contains $|Q_i|$ of your n planets, and planet $\ell \in Q_i$ has "total space distance" $\delta_\ell \geq 0$ to the rest of the planets in disjoint quadrant Q_i, as well a base-building cost c_ℓ. Everything is a (non-negative) integer. You're a rich space warlord, but not *so* rich that you can simply build all of them, unfortunately, so you can only spend a total of D space dollars to establish your k bases. You'd like to do this while getting the most strategic value possible! That is, given n as well as all the Q_i's, δ_ℓ's, c_ℓ's, and D (where $c_\ell \leq D$ for every ℓ), your goal is to identify one planet $\ell_i \in Q_i$ for each quadrant Q_i, such that $\sum_{i=1}^{k} c_{\ell_i} \leq D$ while minimising $\sum_{i=1}^{k} \delta_{\ell_i}$.

a) What is (a reasonable upper bound on) the size of the input to the problem, *represented in bits*, as a function of n, k, D, $\Delta := \max_{1 \leq i \leq k} \max_{\ell \in |Q_i|} \delta_\ell$?
(You can upper bound the number of ways to partition a set of size n into k sets by the (loose, but OK) bound k^n.)

b) Define the decision version of the above search problem.

c) Prove that this decision version is in NP.

d) Prove that this decision version is NP-Complete.

e) Conclude that the search problem is NP-Hard. (But there's nobody to send to the space crocodiles anymore!)

This is annoying. And yet, you're not going to let NP-Hardness get in the way of your space bases! Let's design an algorithm to solve our problem and place our k bases. As long as it runs in time polynomial in n and D, you'll be happy... of course, the faster running time, the better.

f) Describe the algorithm in plain English.

g) Prove its correctness.

h) Establish its time complexity.

Problem 3. (30 points) **(COMP3927 only)** You got the planets. You got the bases. You got the space empire! But now you're a little worried. You've realised that your whole directional wormlink system (which was quite cheap, in hindsight) might actually fail catastrophically and destroy your space ships if used while 3 or more of the wormlinks are crossing. *(Don't cross the streams, they said!)*

Which could happen if any 3 of your n connected planets were to align while your wormlink system is used. So you need a way to quickly check, on any day, if there is an alignment of 3 of your planets. For this, you can assume each planet is only specified by 2 coordinates, so that planet i corresponds to a point (x_i, y_i). You want an algorithm to solve this problem running in time at most quadratic, i.e., $O(n^2)$.

a) Describe the algorithm in plain English.

b) Prove its correctness.

c) Establish its time complexity.

$O(n^2)$ is fast (good!). But not, you know, lightspeed fast. You'd like to do better than quadratic! Unfortunately, there is now a space crocodile following you, and its stomach seems to be screaming something like "it's impossible!"

Specifically, assume every algorithm for the following problem must have worst-case time complexity as least $\Omega(n^{1.999})$:

> Given an array A of n (possibly negative) integers, decide whether there exist 3 distinct indices $1 \leq i, j, \ell \leq n$ such that $A[i] + A[j] + A[\ell] = 0$. (‡)

Your task is to show that, assuming this conjecture, every algorithm for your wormlink-planet-alignment problem must also have worst-case time complexity $\Omega(n^{1.999})$ (so you cannot hope for much better than your $O(n^2)$-time algorithm). *Hint: recall that three points $(x_1, y_1), (x_2, y_2), (x_3, y_3)$ (with distinct x_i's) are collinear iff $\frac{y_3 - y_1}{x_3 - x_1} = \frac{y_2 - y_1}{x_2 - x_1}$, and that $a^3 - b^3 = (a - b)(a^2 + ab + b^2)$. Start with an instance of problem (‡), and transform each array element $A[i]$ into a point $(A[i], f(A[i]))$ for a suitably chosen function $f : \mathbb{R} \to \mathbb{R}$.*

d) Describe the reduction in plain English.

e) Prove its correctness.

Epilogue. Space is big. But you did it! *All your base are belong to you.*

Advice on how to do the assignment

- Assignments should be typed and submitted as pdf (no pdf containing text as images, no handwriting).
- Start by typing your student ID at the top of the first page of your submission. Do **not** type your name.
- Submit only your answers to the questions. Do **not** copy the questions.
- When asked to give a plain English description, describe your algorithm as you would to a friend over the phone, such that you completely and unambiguously describe your algorithm, including all the important (i.e., non-trivial) details. It often helps to give a very short (1-2 sentence) description of the overall idea, then to describe each step in detail. At the end you can also include pseudocode, but this is optional.
- In particular, when designing an algorithm or data structure, it might help you (and us) if you briefly describe your general idea, and after that you might want to develop and elaborate on details. If we don't see/understand your general idea, we cannot give you marks for it.
- Be careful with giving multiple or alternative answers. If you give multiple answers, then we will give you marks only for "your worst answer", as this indicates how well you understood the question.
- Some of the questions are very easy (with the help of the slides or book). You can use the material presented in the lecture or book without proving it. You do not need to write more than necessary (see comment above).
- When giving answers to questions, always prove/explain/motivate your answers.
- When giving an algorithm as an answer, the algorithm does not have to be given as (pseudo-)code.
- If you do give (pseudo-)code, then you still have to explain your code and your ideas in plain English.
- Unless otherwise stated, we always ask about worst-case analysis, worst case running times, etc.
- As done in the lecture, and as it is typical for an algorithms course, we are interested in the most efficient algorithms and data structures.
- If you use further resources (books, scientific papers, the internet,...) to formulate your answers, then add references to your sources and explain it in your own words. Only citing a source doesn't show your understanding and will thus get you very few (if any) marks. Copying from any source without reference is considered plagiarism.