SigmaCamp's *Problem of the Month* Contest

DECEMBER 2024

Starting from September 2024, we are requiring all submissions to be `.pdf` files (except for CS, which requires `.py` or `.java` files). If you are using Word, you may export to PDF by clicking File > Export > Create PDF/XPS Document.

Mathematics

For all mathematics problems, please provide full justification. **Do not include any code** in your submission — all code submissions will be awarded no points.

5 points:

You have probably noticed that a square of any number ending in 5 ends in 5 (e.g. $15^2 = 225$) and a square of any number ending in 25 ends in 25 (e.g. $125^2 = 15625$).

Similarly, we can observe that

- since $6^2 = 36$, a square of any number ending with 6 will end with 6;
- since $76^2 = 5776$, a square of any number ending with 76 will end with 76;
- since $376^2 = 141376$, a square of any number ending with 376 will end with 376.

Explain how to continue the sequence $a_1 = 6, a_2 = 76, a_3 = 376, a_4 = 9376$ indefinitely without trial and error, using at most one multiplication to calculate a_{n+1} from a_n, and find a_{14}. (You can use Wolfram Alpha to multiply long numbers exactly.)

Hint:

No hint this month.

Solution:

Answer: 40 081 787 109 376.

This solution uses the standard notation $\overline{xyz}$ to designate a number written using the digits x, y, z, so that $\overline{xyz} = 100x + 10y + z$. We'll also use $\square$ as a placeholder for an unknown digit, like $\overline{x\square z}$.

Let us first consider how to figure out a_3. We know that the last two digits are 76, so $a_3 = \overline{x76}$, and

$$a_3^2 = \overline{x76}^2 = (100x + 76)^2 = 10000x^2 + 2x \cdot 100 \cdot 76 + 76^2.$$

We are only interested in the hundreds digit (third from the right) of a_3^2. The first term $10000x^2$ does not contribute anything to the hundreds, the last term is $5\mathbf{7}76$ and contributes 7, and the second term $2x \cdot 100 \cdot 76 = 15200x$ can be represented as $\overline{\Box \ldots \Box (2x)00}$ — the hundreds digit is the last digit of $2x$. Note that the 2 in $2x$ is the last digit of $2 \times 6 = 12$.

To have $a_3^2 = \overline{x76}^2 = \overline{\Box \ldots \Box x76}$ we then need to have either $7+2x = x$, which is impossible, or $7+2x = 10+x$, which gives $x = 10 - 7 = 3$, and $a_3 = 376$.

To go from a_n to a_{n+1}, the process is exactly the same, and the only thing that changes is the contribution from the last term. If the $n+1$-th digit (from the end) of a_n^2 is y, the equation is again $y + 2x = 10 + x$, so that $x = 10 - y$.

To summarize, to find a_{n+1} given a_n, we should:

- Find a_n^2 (actually, we only need the last $n+1$ digits of it).
- Let y be the $n+1$-th digit (from the end) of a_n^2. Set $x = 10 - y$ (if $y = 0$ then set $x = 0$).
- Set $a_{n+1} = x \cdot 10^n + a_n$ (write digit x to the left of a_n).

It takes about 2 minutes to find, for example, $a_{21} = 607743740081787109376$.

10 points:

A function $f(x_1, \ldots, x_{2023})$ is defined by

$$f(x_1, \ldots, x_{2023}) = \sqrt{x_1^2 + 1^2} + \sqrt{x_2^2 + 2^2} + \sqrt{x_3^2 + 3^2} + \cdots + \sqrt{x_{2023}^2 + 2023^2}.$$

We know that the minimal value of f when $x_1 + x_2 + x_3 + \cdots + x_{2023} = k$ is 2023×2024. Find k.

Hint:

This is a geometry problem.

Solution:

Answer: $2023 \times 1012 \times \sqrt{3}$**.**

Note that $f(x_1, \ldots, x_{2023})$ is equal to the length of a polygonal chain connecting the points

$$(0,0), (x_1, 1), (x_1 + x_2, 3), \ldots, (x_1 + x_2 + \cdots + x_{2023}, 1 + 2 + \cdots + 2023).$$

Therefore, the minimal value of f corresponds to the shortest length of the line, which is achieved when the line is the straight line connecting the first point $(0,0)$ to the last point $(x_1 + x_2 + \cdots + x_{2023}, 1 + 2 + \cdots + 2023)$. It is given as a constraint that $x_1 + x_2 + \cdots + x_{2023} = k$. To find $1 + 2 + \cdots + 2023$, we can use the formula $\frac{n(n-1)}{2} = \frac{2024 \times 2023}{2} = 1012 \times 2023$, so the shortest length is the distance from $(0,0)$ to $(k, 1012 \times 2023)$. Now we need to solve the equation

$$\sqrt{k^2 + 1012^2 \times 2023^2} = 2023 \times 2024,$$

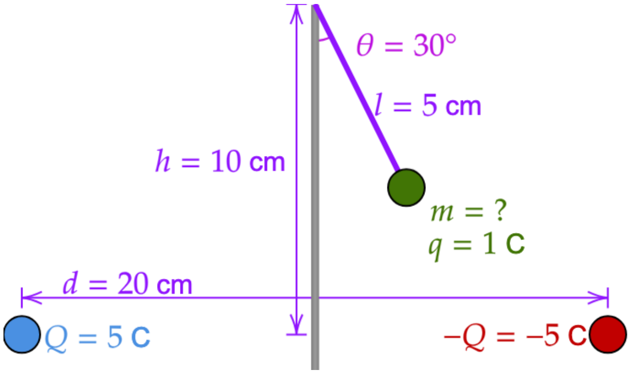
to find

$$k = \sqrt{2023^2 \times 2024^2 - 1012^2 \times 2023^2} = 2023 \times 1012 \times \sqrt{4-1} = 2023 \times 1012 \times \sqrt{3}.$$

Physics

5 points:

Two oppositely charged metal balls are secured 20 cm apart. Exactly halfway between them, a wooden pole is placed. At its top, 10 cm above the centers of the metal balls, a 5 cm-long string is attached, with a metal ball at the end (see diagram). Find the mass of the ball if the angle the string forms with the pole is 30°.



Hint:

No hint this month.

Solution:

There are four forces on the mass: the tension force from the string, the two electric forces from the metal balls, and the gravitational force. The tension force from the string will perfectly cancel any force parallel to the string, so we will only consider the perpendicular components of the forces.

There are many ways of finding perpendicular components, but the simplest if you know a bit of linear algebra is to find the unit vector in the direction of choice and then take the dot product. Here, our unit vector will be perpendicular to the string. There are two choices, and we will pick $\hat{u} = (-\cos 30°, -\sin 30°)$. The gravitational force on the charge q is

$$F_g = (0, -mg). \tag{1}$$

The perpendicular component of the gravitational force is

$$F_g^{\perp} = mg \sin 30°. \tag{2}$$

The electric force takes the form

$$\vec{F}_{1 \text{ on } 2} = k \frac{q_1 q_2}{d^3} \vec{d}_{1 \text{ to } 2}, \tag{3}$$

where q_1 and q_2 are the charges, $\vec{d}_{1 \text{ to } 2}$ is the vector from the first charge to the second charge, and

$$k = \frac{1}{4\pi\varepsilon_0} = 8.99 \cdot 10^9 \text{ N}\cdot\text{m}^2\cdot\text{C}^{-2}. \tag{4}$$

We are given the charges in the problem, but we must do a bit of work to find the distances. So that we don't get confused about units, we'll do all our work in SI units (that means m instead of cm). We will call the position of the charge q $\vec{r}_3$, the position of the charge $+Q$ $\vec{r}_1$, and the position of the charge $-Q$ $\vec{r}_2$. Then

$$
\begin{aligned}
\vec{r}_1 &= (-0.1, 0) \text{ m} \\
\vec{r}_2 &= (0.1, 0) \text{ m} \\
\vec{r}_3 &= (0.05 \sin 30^\circ, 0.1 - 0.05 \cos 30^\circ) = (0.025, 0.057) \text{ m} \\
\vec{d}_1 &= \vec{r}_3 - \vec{r}_1 = (0.125, 0.057) \text{ m} \\
\vec{d}_2 &= \vec{r}_3 - \vec{r}_2 = (-0.075, 0.057) \text{ m}
\end{aligned}
$$

Then the electrical forces on the charge q will be

$$\vec{F}_1 = k\frac{qQ}{|\vec{d}_1|^3}\vec{d}_1 = (2.17 \cdot 10^{12}, 9.85 \cdot 10^{11}) \text{ N} \tag{5}$$

$$\vec{F}_2 = k\frac{q(-Q)}{|\vec{d}_2|^3}\vec{d}_2 = (4.06 \cdot 10^{12}, -3.07 \cdot 10^{12}) \text{ N.} \tag{6}$$

The total electrical force in the perpendicular direction will be

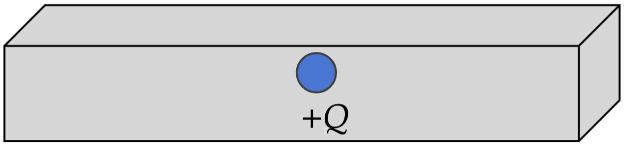
$$F_e^{\perp} = (\vec{F}_1 + \vec{F}_2) \cdot \hat{u} = -4.35 \cdot 10^{12} \text{ N.} \tag{7}$$

The sum of the perpendicular components of the forces will be 0, so $F_g^{\perp} = -F_e^{\perp}$. Thus, combining (2) and (7),

$$m = -\frac{F_e^{\perp}}{g \sin 30^\circ} = 8.88 \cdot 10^{11} \text{ kg.} \tag{8}$$

10 points:

A ball with charge $+Q$ is placed at the center of a neutral metal box that is a long rectangular prism.



(a) Draw the (approximate) electric field lines and equipotentials in the plane that is parallel to the bottom of the metal box and runs through the charge (don't forget to think about both the inside and the outside of the box).

(b) Now imagine the metal box is grounded. Draw the new electric field lines and equipotentials and find the charge on the metal tetrahedral box.

For each part, please provide a short explanation as to why you drew the field lines and equipotentials the way you did. For example, if the equipotentials were parabolas (which they aren't), you would have to both draw them as parabolas and provide an explanation as to why they are parabolas to get points for that part of the question.

Hint:

No hint this month.

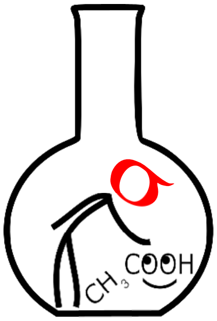
Solution:

a) The field lines close to the charge will look radially symmetric, pointing outwards, as the charge is positive. As the field lines approach the box, they will be oriented to hit its walls perpendicularly.
Note that the box is metallic and therefore a conductor. The electric field is proportional to the force that would be exerted by the charge Q on some other charge. If the electric field was not perpendicular to the box, then the electrons on the box would experience a force and move around. A steady current would then be generated which is not possible in this electrostatic system.

   As equipotential surfaces must be perpendicular to the electric field lines, they are circular close to the charge and take the shape of the box as they approach the box shape.

   Far away outside of the box, the electric field looks radial outwards and equipotential lines are circular

b) Since the metal box stays a conductor, the reasoning for the shape of electric field lines inside the box is the same.
The box collects negative charge, which is attracted by the positive charge. Overall the box then becomes neutrally charged and no electric field is observed outside.



Chemistry

5 points:

Alice, Bob's graduate student, was standing by the bench, gazing at the flasks on the shelf. Bob noticed that Alice appeared disappointed and asked her what was wrong. "Bob, I made a silly mistake. I needed to dissolve the compound I prepared in ethyl acetate, but I accidentally added distilled water instead. Now I have 10 grams of my valuable compound dissolved in 1 liter of water, and I don't know how to retrieve it. I'm worried that I will have to evaporate all the water, which will take more than two days. Do you have any suggestions?"

"Just perform an extraction with ethyl acetate," Bob advised. "Ethyl acetate and water don't mix, so if you add ethyl acetate to your solution and shake it up, your compound will migrate into the ethyl acetate layer, making it easy to separate."

"Yes, I understand that, Bob. However, we only have one liter of ethyl acetate in the lab; I've asked our technician to buy more, but the order will be delivered only after Christmas. I know that the partition coefficient of my compound between water and ethyl acetate is 1, which means by using one liter of ethyl acetate, I can only recover 5 grams of it; the rest will remain in the water. You know that my compound is very expensive, and I cannot afford to lose half of it."

"Not necessarily," Bob replied. "You can extract more. Just do the following: ..."

What did Bob say? Explain the procedure and calculate how much of this compound can be recovered.

Hint:

Split

Solution:

Bob, probably, meant one of the methods described below. The first one is easier to do, but it is less efficient.

1. If Alice takes 1 L of her compound's aqueous solution and extracts it with 1 L of ethyl acetate, she can recover no more than 5 g of the compound because its partition coefficient between water and ethyl acetate is 1. When the distribution coefficient is equal to one, the extraction process can be represented as a simple dilution: by adding 1 L of ethyl acetate to 1 L of the aqueous solution and stirring vigorously, the concentrations in both the aqueous and organic phases can be calculated as if the solution were diluted by an equal volume of solvent. Thus, 5 g of Alice's compound would partition into the ethyl acetate phase, while 5 g would remain in the aqueous phase.

However, if she uses only 100 mL of ethyl acetate to extract 1 L of the aqueous solution, it can be calculated that 9.0909 g of her substance will remain in the aqueous solution, while 0.9091 g will transfer to the organic phase (i.e., the ethyl acetate). Since Alice still has 900 mL of ethyl acetate, she can repeat this process with another 100 mL of fresh ethyl acetate. After the second extraction, the amount of her compound in the aqueous phase will drop to 8.2644 g, while the ethyl acetate phase will contain 0.82644 g. The total amount of Alice's compound in both ethyl acetate extracts (first and second) will be 1.7355 g, which exceeds the amount she could recover from a single extraction with 200 mL of ethyl acetate (in which case she would obtain only 1.6666 g). With 800 mL of fresh ethyl acetate remaining, she can perform eight more extractions, and at the end of this process, only 3.855 g of Alice's compound will remain in the aqueous solution, while the ten ethyl acetate extracts will contain a total of 6.1446 g of her compound.

By dividing the ethyl acetate into more smaller portions, the extraction efficiency can be enhanced further.

2. The efficiency of extraction can also be improved by dividing both the aqueous solution and ethyl acetate into smaller portions and using a method known as ”counter-current liquid-liquid extraction” (this method

can be easily researched online). However, this process is quite labor-intensive, so Bob may not recommend it to Alice.

3. Arguably the best advice would be to use continuous liquid-liquid extraction. If Bob has a special apparatus (this apparatus is relatively simple, the procedure is easy to do), Alice can easily extract almost the whole amount of her compound from the aqueous solution. A description of the continuous liquid-liquid extraction is easy to find, so you can easily read it (or watch a video) by yourself.

10 points:

When Alice entered Bob's office, she noticed him staring at the computer screen, looking embarrassed. "Alice, I know you often use ChatGPT for your work. Is it functioning well?" "Yes, Bob, it's a fantastic tool," Alice responded. "Hmm, I asked it to solve a straightforward problem, but something went wrong. Take a look," he said, gesturing for her to see the screen. The problem stated: "An equimolar mixture of two rubidium halides weighing 4.52 g was treated with an excess of silver nitrate. A precipitate weighing 2.87 g was formed. Determine which halides were present in the mixture." ChatGPT's conclusion was: "The mixture likely contained RbBr (rubidium bromide) and XX" "Did you see that, Alice?" Bob asked. "It seems like it can't solve the problem since 'XX' isn't a meaningful answer. Either ChatGPT isn't as smart as you think, or the problem was poorly formulated." "Unfortunately," Alice replied, "ChatGPT missed a detail. The problem does have a solution: it's important to remember that..."

What did Alice say, and what is the answer?

Hint:

Are all silver halides water-insoluble?

Solution:

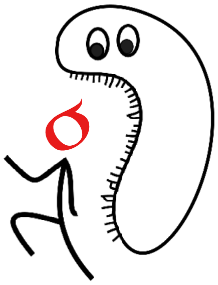
The molar mass of rubidium is 85.46 g/mol, and the molar mass of silver is 107.87 g/mol. Silver is heavier than rubidium, so we would expect the silver precipitate to be heavier than the rubidium salt. However, in this problem, we see that 4.52 grams of RuX produced just 2.87 grams of AgX, which is impossible if these two quantities represent the same number of moles of halide. The key is that not all silver halides are insoluble. AgF is soluble in water and will not precipitate out of solution, while AgCl, AgBr, and AgI are famously insoluble. So, the missing mass is possible if one the two unknown halogens is fluoride. The 2.87 grams of AgX precipitate must be one of AgCl, AgBr, or AgI. We can test these three individually to see which makes the math work.

2.87 g AgCl is 0.0200 mol AgCl → 0.0200 mol RuCl + 0.0200 mol RuF weighs 4.51 g

2.87 g AgBr is 0.0153 mol AgBr → 0.0154 mol RuBr + 0.0154 mol RuF weighs 4.16 g

2.87 g AgI is 0.0122 mol AgI → 0.0122 mol RuI + 0.0122 mol RuF weighs 3.87 g

AgCl is the closest, so the mixture consists of RuCl and RuF.



Biology

5 points:

The thermal stability of proteins indicates that many of them have a denaturation temperature just slightly above 42°C. Why is the denaturation temperature so close to our body temperature, and is there any underlying reason for this?

Hint:

”If you’re seeking stability, visit a graveyard”.

Solution:

Although many proteins appear as dense structures with tightly packed amino acid residues, resembling the arrangement found in organic crystals, it is inaccurate to view proteins as entirely rigid entities. In fact, many enzymes operate under the ”induced fit” mechanism, which means their shape alters upon substrate binding to better accommodate the substrate.

Additionally, some proteins function as ”molecular machines,” with their components moving relative to one another as they carry out their tasks. In this regard, structural changes in many proteins are essential for their proper functioning.

This implies that if proteins are excessively stable and rigid, they may be unable to function effectively; conversely, if they are overly flexible, they may lack thermal stability. The ideal scenario is to achieve a balance, or ”golden mean”; therefore, many proteins in our body have a melting point that is just slightly above our body’s temperature.

10 points:

Proteins are long polymers made up of amino acid residues, each with distinct side groups. The type of side group determines the possible interactions, leading proteins to fold in ways that maximize these interactions. This results in each protein having a unique shape and function. When synthesized in ribosomes, the nascent polypeptide chain is not properly folded; this occurs later in the process. The chain undergoes thermal fluctuations, with most bonds capable of rotating. Fig. 1 shows the bonds in a short peptide that can easily rotate producing different conformations. More stable rotamers are retained, while less stable ones continue to fluctuate. Ideally, a protein should explore all potential conformations (where each bond can rotate at various angles, typically 0, 120, and 270 degrees). However, the number of possible combinations is enormous: if each amino acid residue has three rotatable angles, a dimer has 9 combinations, a trimer has 27, and so forth. The peptide shown in Fig.1 is a very simple example, but even this short peptide has 9 rotatable bonds, each of which can exist in three different rotational states, so the total number of conformations is 3^9. For a protein made of 100 amino acids, the number of combinations is immense (nearly 3^{300}. Taking into account that at least a nanosecond is needed to probe each conformation, the number of possible conformations exceeds the number of nanoseconds that have elapsed since the Big Bang.

This implies that we face a paradox: if proteins fold as a result of random thermal fluctuations, they ought to do so over a timescale similar to the age of the Universe.

How would you address this paradox?

Hint:

Assuming an average protein size of 100 amino acids—a relatively modest estimate—the total number of possible protein sequences is 20^{100}. This is an extraordinarily large number, far exceeding the number of

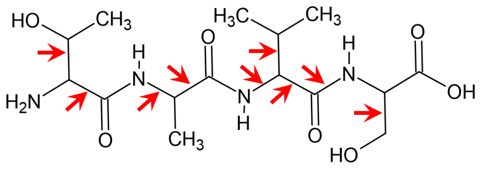


Figure 1: A simple tetrapeptide. Red arrows show rotating bonds.

hadrons (protons and neutrons) in the observable universe. This implies that nature utilizes only a tiny fraction of all potential protein sequences.

Solution:

This problem addresses the well-known Levinthal's paradox, which is well explained in, e.g. the Wikipedia article.

This paradox seems to present a significant challenge because it is generally believed that any system can only achieve its most stable state (i.e., the lowest energy state) by visiting that state multiple times due to thermal motion. However, given that the number of theoretically possible states for any protein is immense, it remains unclear how proteins can ”discover” which state is the most stable.

The general explanations for this paradox involve several key points.

First, if we take a medium-sized or larger protein, heat it until it unfolds (melts), and then gradually cool it down, the protein generally will not revert to its native state. But if proteins folded in the manner described by Levinthal, they would return to their original structure once the temperature decreased (which usually never happens for most proteins, thereby suggesting that Levinthal's assertion was generally wrong). So, how do proteins actually fold? Typically, protein folding occurs during synthesis by ribosomes: the growing protein chain begins folding immediately after it exits the ribosome. Essentially, the N-terminus of the nascent protein starts folding before the complete protein is synthesized, and this folded region serves as a seed for the folding of the remaining parts of the protein molecule.

Second, when Levinthal introduced his paradox, he assumed that any protein sequence could rapidly fold into its lowest energy conformation, which is incorrect. Proteins are made up of 20 amino acids, and if we consider an average protein containing 200 amino acids, the number of possible protein sequences of that length is 20^{200}. This figure is astonishingly large—-greater than the total number of all particles (protons, electrons, photons, etc.) in the observable Universe. Consequently, the proteins present in all living organisms represent only a tiny fraction of the potential protein structures. This provides ample opportunity for natural selection, as only those protein sequences that evolved to fold within a reasonable time frame have emerged. In other words, the proteins all living creatures are composed of belong to the small subset of all possible protein structures that can fold relatively easily. One possible way is described above (*via* gradual folding during peptide synthesis. Another possibility is as follows: some proteins contain specific amino acid sequences that are capable of folding independently of each other due to strong interactions between amino acid side chains. These short segments quickly fold into compact structural motifs separately from each other, and then assemble into a larger structure.

Third, specialized cellular machines known as ”chaperones” assist larger proteins in achieving proper folding.

In summary, regarding Levinthal's formulation of the paradox, the brief answer is as follows: there is no paradox, as most proteins (with the exception of very small ones) do not fold according to the mechanism described by Levinthal.

Linguistics & Applied Sciences

5 points:

In many languages, names of things, such as animals, tend to be based on a combination of more basic words that describe the new concept. These names can describe animals in a way that is useful and easy to use in conversation, even if the name isn't biologically accurate (such as the "seahorse").

Below is a list of 11 sea species and their names in an unknown language:

1. seka mage	A. winged oyster	7. tabui raurau	G. humpback whale
2. seka leuasausau	B. long-snouted coralfish	8. miinana	H. scorpionfish
3. tibi duladula	C. hairy crab	9. diba sonasona	I. partridge tun
4. mosimai	D. horned helmet	10. buburaki seniibi	J. thorny oyster
5. tabui ndisi	E. broad-barred toadfish	11. diba laue	K. spider crab
6. tabui sonasona	F. cameo helmet		

Use the following vocabulary words to match the sea creatures with their names. Provide an explanation for each match:

tibi	the tree *Terminalis catappa*	mage	monkey
sona	to have protrusions	bubu	to sink
dula	to be sewn with needle and thread	mosi	pain
sausau	rough, bony	rau	leaf
nana	poisonous	laue	feather
ndisi	red		

Hint:

No hint this month.

Solution:

Correct Matches:

1. seka mage — C hairy crab
2. seka leuasausau — K spider crab

   By frequency analysis, "seka" can either be crab or oyster. "Mage" correlates to "hairy" because monkies are hairy. Spider crabs look rough and bony, matching "sausau" in "leusausau."

3. tibi duladula — B long-snouted coralfish

   The leaf of the Terminalis catappa is flat and round. Add to it a needle shape, and you get the body and pointed nose of the long-snouted coralfish.

4. mosimai — H scorpionfish
5. tabui ndisi — F cameo helmet
6. tabui sonasona — D horned helmet
7. tabui raurau — I partridge tun

Through frequency analysis, ”tabui” represents ”shell” or something similar. From there, we can easily match up the red stripes of the cameo helmet to ”ndisi,” the white, spiny horned helmet to ”sonasona,” and the smooth partridge tun that has patterns like leaves to ”rau.”

8. miinana — E broad-barred toadfish

   Scorpionfish are spiny and will release painful, but not deadly, toxin if touched, matching the given vocabulary word for pain. Toadfish are poisonous to eat.

9. diba sonasona — J thorny oyster

10. buburaki seniibi — G humpback whale

    The only given vocabulary word in this term is ”to sink.” Whales are the only animal listed who exhibit ”sinking” behavior, such as coming up for air and swimming back down into the ocean.

11. diba laue — A winged oyster

    Finally, the thorny oyster has protrusions, while the winged oyster looks like feathers.

10 points:

(a) The list of English sentences on the right was translated into an unknown Oceanian language and scrambled to produce the list on the right. **Match the English sentences to the Oceanian sentences.** Make sure to **show your work** by explaining the meaning of words and grammatical features you find. Feel free to explain parts of a word in your explanation if you find it necessary. The more you show your work or reasoning, the more partial credit you can get for this problem. Even if you think all your matches are correct — explain how you did it!

1 ee ntook mlaag nok
2 tulkma vte rong late na
3 nee sla aat meen nik
4 va lokp nok
5 o masa oror namee mwermwer
6 nik mdaw lala namee na
7 na vte tek timi nik
8 va we timi
9 na swur nir nir mat
10 slokp nok timi
11 o mwermwer sngor tavool timi
12 na swur lala kmoor
13 na vte la timi aat meen kmoor
14 nee stek timi na
15 nee mtek na a gvoor
16 va wow timi
17 na stek nir
18 o aav va is
19 nee sdaw nok timi
20 na mla aat meen nir
21 na tngor
22 o masa va wow mlitee
23 kmaa vte breeng late nik
24 nik srong tavool timi

A. It’s not good
B. I will see them
C. We won’t be able to help you
D. My child already got married
E. The knife is still dry
F. I gave it to them
G. I’ll be able to curse you both
H. The fire is alive
I. It’s not dry
J. He saw me at home
K. You three won’t be able to hear me
L. I won’t see you
M. She’s no longer doing it
N. The baby didn’t sleep well
O. It’s already wet
P. You could have done it for me
Q. I am sleeping
R. I will curse them so they die
S. You don’t hear well
T. A knife is not a toy for children
U. It’s no longer wet
V. I won’t give it to you both
W. He’ll give it to you
X. I didn’t see her

(b) Translate the following sentences into the unknown language:

i We didn't see her at home
ii I won't be able to marry you
iii I helped them

(c) Translate the following sentences into English:

i nee m'mat nok, le nee va is mlitee?
ii nik sdaw tavool timi
iii ee ntook swe timi

Hint:

No hint this month.

Solution:

Answer:

1 D	2K	3 W	4 O	5 T	6 P	7 L	8 A	9 R	10 U	11 N	12 G
13 V	14 X	15 J	16 I	17 B	18 H	19 M	20 F	21 Q	22 E	23 C	24 S

A 8	B 17	C 23	D 1	E 22	F 20	G 12	H 18	I 16	J 15	K 2	L 7
M 19	N 11	O 4	P 6	Q 21	R 9	S 24	T 5	U 10	V 13	W 3	X 14

we didn't see her at home	kmaa s tek timi nee a gvoor
i won't be able to marry you	na vte laag late nik
i helped them	na m breeng nir
nee m' mat nok, le nee va is mlitee?	did (s)he die already, or is (s)he still alive?
nik s daw tavool timi	you're not doing (it) well / correctly
ee ntook s we timi	my child is not well / my child became sick

Solution:

Pronouns: *na*: I/me *nik*: you *nee*: he/she/her
kmaa: we *kmoor/tulkma*: you two/three *nir*: they/them

Nouns: *(a) gvoor*: (at) home *(o) mwermwer*: (the) child(ren) *masa*: knife
oror: toy *aav*: fire *(ee) ntook*: (my) son / daughter

Verbs: *tek*: see *va*: to be *daw*: to do *la (aat meen)*: to give (it to __) *rong*: to hear *ngor*: to sleep *laag*: get married *breeng*: to help *wur*: to curse *mat*: to die

Adjectives: *wow*: dry *lokp*: wet *we*: good **Adverbs:** *tavool*: well *nok*: already *mlitee*: still

An important distinguishing feature in this language is grammatical tense. Here are four common tenses, all of which contain the verb 'tek' (to see). **Notice:** 's-' actually has two very different meanings!

Past / present: nee m tek na a gvoor	he saw me at home	(15 / J)
Negative past / present: nee s tek timi na	he didn't see me	(14 / X)
Future: na s tek nir	I will see them	(17 / B)
Negative future: na vte tek timi (or late) nik	I won't (be able to) see you	(7 / L)

As you may find below, lala / late refers to whether someone can / cannot or is able / unable to do something. Other important terms include nok, which means 'already', as well as mlitee, which means 'still'. Verbs that express an unchanging state (such as va : to be) tend not to include the prefixes / particles described above, opting simply for timi as a negation term:

va lokp nok	it's already wet	(4 / O)
o masa va wow mlitee	the knife is still dry	(22 / E)
o aav va is	the fire is alive	(18 / H)
va wow timi	it's not dry	(16 / I)
va we timi	it's not good	(8 / A)
o masa oror namee mwermwer***	a knife is not a toy for children	(5 / T)

***Correction: o masa oror namee mwermwer timi. Points will **not** be deducted for errors related to this sentence — apologies for late notice

Here is how the rest of the sentences break down in terms of tense:

Past / present tense:

nik m daw lala namee na	you could have done (it) for me	(6 / P)
ee ntook m laag nok	my child already got married	(1 / D)
na m la aat meen nir	I gave it to them	(20 / F)

Negative past / present tense:

nee s daw nok timi	she's no longer doing (it)	(19 / M)
nik s rong tavool timi	you don't hear well	(24 / S)
s lokp nok timi	it's no longer wet	(10 / U)
o mwermwer s ngor tavool timi	the baby didn't sleep at all	(11 / N)

Future tense:

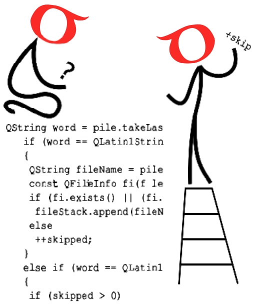
nee s la aat meen nik	he 'll give it to you	(3 / W)
na s wur lala kmoor	I 'll be able to curse you both	(12 / L)
na s wur nir nir mat	I will curse them (so) they die	(9 / R)

Negative future tense:

tulkma vte rong late na	you three won't be able to hear me	(2 / K)
kmaa vte breeng late nik	we won't be able to help you	(23 / C)
na vte la timi aat meen kmoor	I won't give it to you both	(13 / V)

Only exception sentence to these rules is the following, which uses 't-' as a progressive prefix (meaning the action is ongoing) ***na tngor*: I am sleeping (21 Q)** versus *na mngor*: I have slept.

Computer Science

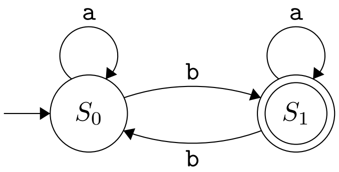


```
QString word = pile.takeLas
   if (word == QLatin1Strin
   {
    QString fileName = pile
    const QFileInfo fi(f le
    if (fi.exists() || (fi.
     fileStack.append(fileN
    else
     ++skipped;
   }
   else if (word == QLatin1
   {
    if (skipped > 0)
```

This month, you will not be submitting code files (`.py` or `.java`) for CS. Instead, you will be submitting a `.pdf` file for each question.

5 points:

Automata theory is the study of *automata*, which are computational models used in many different areas of computer science. In the September 2023 Linguistics/Applied Sciences POM, we explored how automata theory is used for linguistics (we encourage checking out that problem and its solutions). This month, we will explore an application of automata theory to parsers in programming languages.



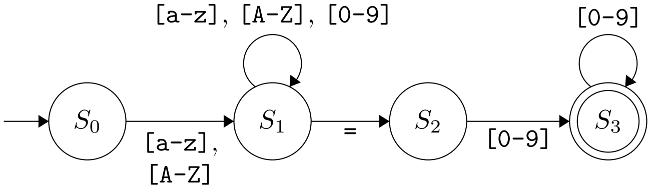
An *automaton* consists of *states* and *transitions* between the states. In the above automaton, the state S_0 is the *initial state*, indicated by the arrow going into it on the left. The state S_1 is a *final state*, indicated by it being double-circled. The transitions are arrows between the states, and are labelled by `a` and `b` in this example. An automaton *reads* a string of characters from left to right and follows the transitions, starting from the initial state. If the automaton finishes reading the string and ends on a final (double-circled) state, we say that the automaton *accepts* that string.

For example, the string "`ab`" is accepted by the above automaton, as the automaton starts at S_0, reads `a` and loops back to S_0, then reads `b` and goes to S_1, which is a final state. Some other strings that are accepted by this automaton are: `bbb`, `baaa`, `bbabbab`. Some strings that are *not* accepted are: `bb`, `bbabab`, `bbababababaab`, and the empty string. After a bit of thought, we can see that the above automaton accepts a binary string if and only if the number of `b`'s in the string is *odd*.

Automata are heavily used in designing parsers for programming languages. Below is an automaton that will accept valid Python declarations of integers of the form

`<variable name>=<digits in 0-9>`

where the variable name can contain letters and numbers but does not start with a digit:



In the above automaton, transitions with the label `[0-9]` allow any digit from 0 to 9. Similarly, `[a-z]` refers to any lowercase letter, and `[A-Z]` refers to any uppercase letter. Some strings accepted by this automaton are (check it for yourself!):

`a=5`   `myVariable=0`   `num123=0401`   `hi12bye34=56565656`

Some strings that are *not* accepted by this automaton are (check it for yourself!):

`1var=3`   `num=`   `=3`   `mystring="hi"`   `var=A5`   `my_num=3`

Note that the strings `mystring="hi"` and `my_num=3` are not accepted since the quotation mark `"` and the underscore `_` are not valid characters for the above automaton.

For this question, you will **design and draw an automaton** that accepts the first line of Python function declarations, which are of the format

`def <name>(<params>):`

The `<name>` field can consist of letters and numbers, but cannot start with a number. The `<params>` field can contain zero, one, or multiple comma-separated parameters, which each can consist of letters and numbers but cannot start with a number.

For example, some strings your automaton should accept are:

- `def myFunc():`
- `def gcd(a, b):`
- `def func1(myParam5):`
- `def funnyFunction123(num1, num2, num3, num4, num5):`

Some strings that are should *not* be accepted are:

- `def 1func():` *(the function name cannot start with a digit)*
- `func(a, b):` *(the function declaration must start with “`def `”)*
- `def thing(` *(the declaration must have parentheses and end with a colon)*
- `def gcd(1a, 1b):` *(parameter names cannot start with digits)*
- `def func(,):` *(the interior of parentheses should be empty if there are no parameters)*

Make sure your automaton includes an initial state and a final state, and test your automaton on the above examples and come up with some other ones to ensure that it is correct. **If desired, your automaton may have more than one final state.**

Hint:

No hint this month.

Solution:

The automaton is shown in Figure 2.

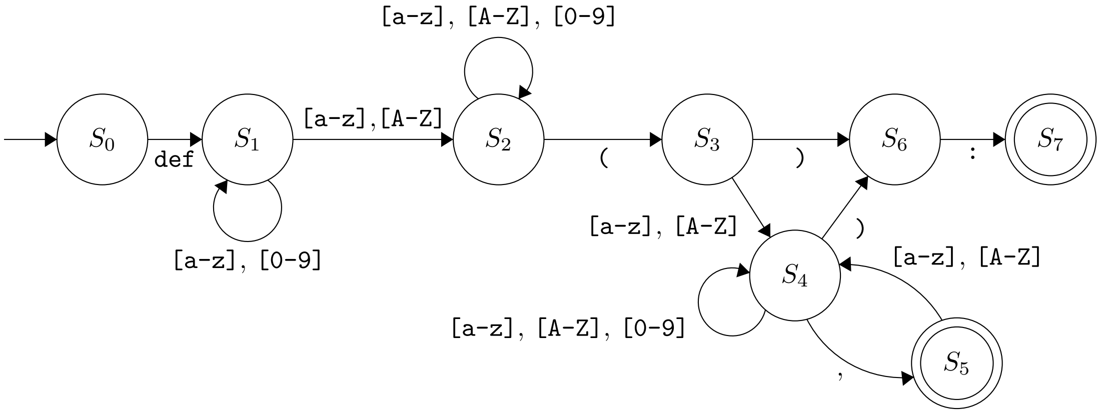


Figure 2: Automaton for accepting Python function definitions.

10 points:

For the following problem, you should submit a `.pdf` with your answers including any code you write and any plots. **Submissions that only submit code will not receive full points!**

Oh no! A dangerous storm is heading for SigmaCamp! Mark instantly runs to the chemistry lab and starts brewing a potion that will protect the camp from the approaching storm. To make the perfect potion, four ingredients need to be combined in precise amounts.

The potency of the potion is determined by the sum of the strengths of the ingredients used in the potion, and Mark's goal is to create a potion whose potency is as close as possible to the required target value T, using any *integer* combination of the four ingredients. Each of the four ingredients has different strengths S_1, S_2, S_3, and S_4 per milligram of the ingredient used. For example, if 3 mg of ingredients 1 and 3 are used and 6 mg of ingredients 2 and 4 are used, then the potency of the potion is $3S_1 + 6S_2 + 3S_3 + 6S_4$.

Your task is to write a Python or Java program to help Mark determine the closest possible potion potency to T that can be created using a combination of the four available ingredients.

Your code should read the input file `input.txt`, which contains a single line containing space-separated values for T, S_1, S_2, S_3, and S_4. It should output to the file `output.txt`, containing a single line of space-separated non-negative integers a, b, c, d that minimize the quantity $|T - (aS_1 + bS_2 + cS_3 + dS_4)|$. If the answer is not unique, output the *lexicographically smallest* list of integers.

Implement your code in the following two ways:

(a) First, implement your program using the naive approach of iterating through all possibilities of sums. Include the code in your writeup, and give a high-level overview of your code.

What is the relationship between your program's runtime and the value of T? Is the time proportional to T, T^2, or something else? Why?

Try out different values of T and record how much time your program takes. Plot the values using your favourite plotting software, and include the plot in your writeup.

(b) Now, implement your program using *dynamic programming*. Include the code in your writeup, and give a high-level overview of your code.

What is the relationship between your program's runtime and the value of T?

Try out different values of T and record how much time your program takes. Plot the values using your favourite plotting software, and include the plot in your writeup.

Timing your code

In Python, you can time your code by using `time.time()` from the `time` package, as done below:

```python
import time
t0 = time.time()
...
t1 = time.time()
print("My code runs in", t1-t0, "seconds")
```

In Java, you can time your code using the `System.currentTimeMillis()` function.

Examples

Sample Input 1:

```
50 3 6 7 2
```

Sample Output 1:

```
0 0 0 25
```

Sample Explanation 1:

Note that $25 \cdot 2 = 50$. While many other combinations of ingredients are possible (e.g. `3 3 3 1`, the one given above is the lexicographically smallest one.

Sample Input 2:

```
54 20 15 40 90
```

Sample Output 2:

```
0 1 1 0
```

Sample Explanation 2:

Note that $1 \cdot 15 + 1 \cdot 40 = 55$ is the closest achievable value to 54.

Hint:

Enumerate through all numbers k from 1 to T and keep track of the closest possible potion potency to k.

Solution:

The solution to parts (a) and (b), as well as the source code for the plot generation, are available on the SigmaCamp GitHub repository here:

https://github.com/SigmaCode/POM-QQ/tree/main/CS/2024-2025/December

The naive part (a) solution runs in time proportional to T^4, since it uses four nested for loops, which each loop $O(T)$ times (proportional to T). The DP (dynamic programming) solution for part (b) runs in time proportional to T, as it loops through all possible potency values from 0 to T. Figure 3 shows a plot of the runtimes of the naive and DP solutions.

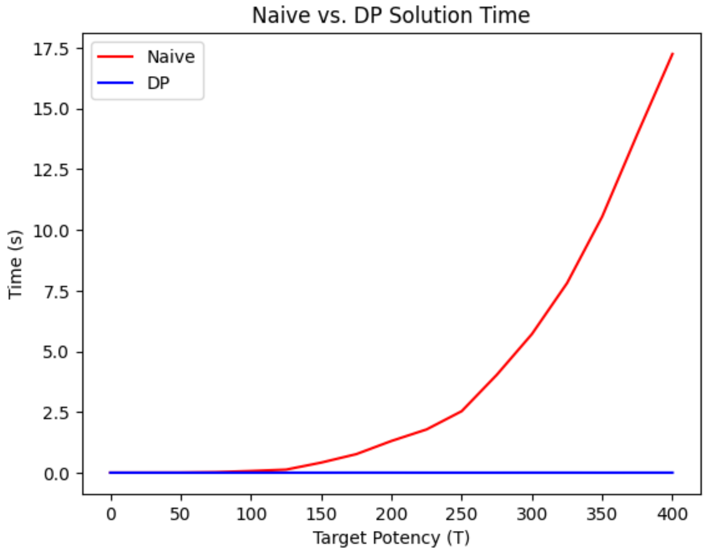


Figure 3: Comparison of runtimes of DP and naive solutions.

Note that the naive solution drastically grows as T increases, while the DP solution stays flat near 0. Figure 4 shows a plot of the DP solution runtime. The DP solution is able to solve the problem in less than a second when $T = 200000$, while the naive solution takes nearly 20 seconds when $T = 400$.

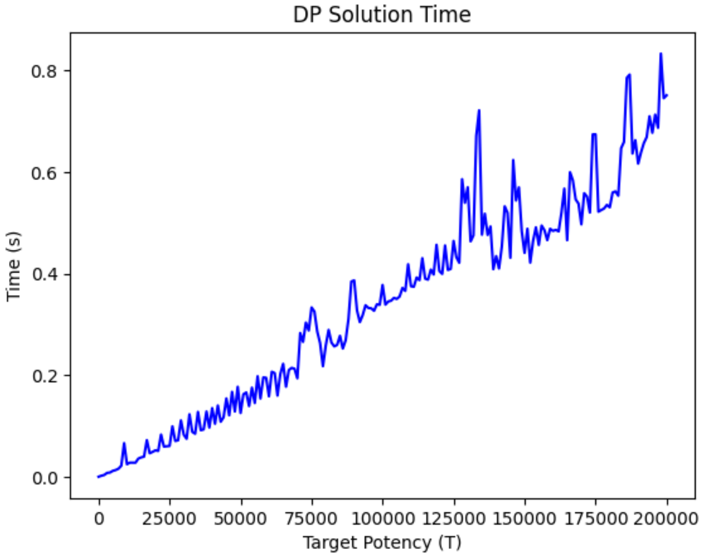


Figure 4: Plot of runtime of DP solution.