## General Instructions

**(i)** This booklet contains 65 questions, each provided with a complete, step-by-step solution.

**(ii)** It comprises 35 single-correct multiple-choice questions, 12 multiple-correct questions and 18 numerical / integer-type questions.

**(iii)** Attempt each question on your own before reviewing the given solution.

**(iv)** For multiple-correct questions, all correct options must be selected to earn full marks.

**(v)** For numerical questions, report the answer rounded exactly as asked.

**1.**

Verbosity : Brevity :: Insolence : _____

Choose the word that best fills the blank.

(A) Innocence

(B) Respect

(C) Solace

(D) Wealth

**Correct Answer:** (B) Respect

**SOLUTION:**

A fast way to solve a word analogy is to name the exact relationship between the first two words, then reuse that same relationship on the second pair. Verbosity and Brevity both describe how much a person talks, one is 'too much', the other is 'too little'. So the relationship is a direct antonym pair on a single scale, the amount of speech.

Insolence sits on a different scale, how a person treats others. It marks the rude, disrespectful end of that scale. The correct answer has to sit at the opposite end of the very same scale, the polite or respectful end, nothing else.

1. **Innocence** sits on a guilt or wrongdoing scale. That is a different scale from manners, so it cannot be the opposite of insolence.
2. **Solace** sits on a comfort or grief scale, again unrelated to how someone treats others.
3. **Wealth** sits on a money scale, which has nothing to do with rude or polite behaviour.
4. **Respect** sits on the exact same manners scale as insolence, at the opposite end. This is the only option that keeps the analogy on the correct scale.

Since the task is to find the antonym of insolence, and respect is the only word that opposes insolence on the same manner-towards-others scale, the blank is filled by Respect, matching option (B).

**Quick Tip:** Find how the first two words are related (they are opposites), then look for the opposite of the second word in the blank pair.

**2.**

The product of the digits of a three-digit number is 70. The sum of the digits of this three-digit number is _____

(A) 12

(B) 14

(C) 16

(D) 18

**Correct Answer:** (B) 14

**SOLUTION:**

**Step 1:** List the possible three-digit factorisations of 70.
We need three whole numbers between 0 and 9 whose product is **70**. Start by picking the hundreds digit and see what is left for the other two digits.

**Step 2:** Try each possible first digit.
If a digit is **1**, the other two digits must multiply to **70**. Checking pairs from **1** to **9**: no pair of single digits multiplies to **70**, since the closest we can get is $9 \times 9 = 81$ and **70** has no factor pair both under **10** (its factor pairs are $1 \times 70$, $2 \times 35$, $5 \times 14$, $7 \times 10$, none with both members under **10**). So no digit set contains a **1**.
If a digit is **2**, the other two must multiply to $70/2 = 35 = 5 \times 7$. Both **5** and **7** are valid digits, so **{2, 5, 7}** works.
If a digit is **5**, the other two must multiply to $70/5 = 14 = 2 \times 7$, again giving the same set **{2, 5, 7}**.
If a digit is **7**, the other two must multiply to $70/7 = 10 = 2 \times 5$, once more giving **{2, 5, 7}**.
No other starting digit (3,4,6,8,9) divides **70** exactly with an integer digit-pair result, so this search confirms only one digit set exists.

**Step 3:** Add the digits.
$2 + 5 + 7 = 14$, and this sum does not change no matter which of the three digits sits in the hundreds place.

**Step 4:** Match with the options.
Out of **12, 14, 16, 18**, only **14** equals this sum.

$\boxed{14}$

**Quick Tip:** Break 70 into three factors that are each a single digit, then add them.

**3.**

The four pieces of a puzzle are shown in the figure below.

Which one of the figures labelled as P, Q, R, and S can be constructed by using each of the four pieces only once without overlaps?

(A)

P

Figure P

(B)

Q

Figure Q

(C)

R

Figure R

(D)

S

Figure S

**Correct Answer:** (A)

P

Figure P

**SOLUTION:**

A reliable way to solve a piece-assembly question like this is to count distinctive features on the four loose pieces first, a smooth curved bite, a matching curved bump, a zig-zag jagged edge, and a twin-peaked spike, and then check each candidate figure for exactly those same features, no more and no less.

The rounded piece and the dome piece share one smooth curve between them (one is concave, one is convex), so together they should produce one clean, unbroken curve somewhere in the final outline, with no extra bumps added there. The zig-zag piece is the only source of jagged, saw-tooth edging in the picture, so the final outline should show exactly one jagged run, not two. The twin-peaked piece is the only source of a beak-like spike, so the final outline must show exactly one spike sticking out, and the smaller of its two peaks has to be absorbed inside the curved bite rather than showing up as a second bump.

1. **Figure Q** fails immediately: it shows no spike at all. Since the twin-peaked piece must contribute a spike somewhere on the boundary, an outline with no spike cannot be built from these four pieces.
2. **Figure R** shows the beak, but it also shows extra jagged bumps crowding the base of the beak, more jagged detail than the single zig-zag piece can supply once its edge is already used for the neck step. That is one jagged feature too many.
3. **Figure S** shows a star-like double spike at the neck in addition to the beak, again asking for jagged or pointed detail beyond what the four pieces contain between them.

4. **Figure P** shows exactly one clean spike (the beak), one smooth curve (head to shoulder), and one jagged step (the neck), which is precisely the feature count carried by the four given pieces: one spike source, one curve pair, one zig-zag source.

Matching feature for feature rules out Q, R and S, leaving Figure P as the only outline whose count and type of edges agrees with the four given pieces.

Let's summarize the feature count check:

- One curve pair (the rounded piece and the dome piece) can only ever produce one smooth curved boundary, never a jagged one.
- One zig-zag piece can only ever produce one jagged run of edge, so an outline with two separate jagged regions, like R, is impossible.
- One twin-peaked piece can only ever produce one visible spike once its smaller peak is tucked away inside the curved bite, so an outline with no spike (Q) or with a double spike (S) cannot be built.

Since Figure P is the only outline whose curve count, jagged-edge count and spike count all match the four given pieces exactly, once, it is the figure that can be assembled from them.

**Quick Tip:** Count the distinct edge types on the four pieces (one curve pair, one jagged edge, one spike) and check which candidate figure shows exactly that many of each, no extra bumps or missing spikes.

• • •

**4.**

Consider two distinct positive real numbers $m, n$, with $m > n$.

Let $x = n^{\log_{10}(m)}$ and $y = m^{\log_{10}(n)}$. The relation between $x$ and $y$ is _____.

(A) $x > y$

(B) $x < y$

(C) $x = y$

(D) $x = \log_{10}(y)$

**Correct Answer:** (C) $x = y$

**SOLUTION:**

**Step 1:** Test the relation with actual numbers first.

Pick two distinct positive numbers with $m > n$, say $m = 100$ and $n = 10$. Then:

$$x = n^{\log_{10} m} = 10^{\log_{10}(100)} = 10^2 = 100$$

$$y = m^{\log_{10} n} = 100^{\log_{10}(10)} = 100^1 = 100$$

Here $x = y = 100$, so for this pair the two quantities are equal.

**Step 2:** Try a second pair to make sure it is not a coincidence.
Take $m = 1000$, $n = 100$:

$$x = 100^{\log_{10}(1000)} = 100^3 = 1{,}000{,}000$$

$$y = 1000^{\log_{10}(100)} = 1000^2 = 1{,}000{,}000$$

Again $x = y$. Two different examples both giving equality is a strong hint that $x = y$ holds in general, not just by chance.

**Step 3:** Prove it holds for every m, n using logs.
Take $\log_{10}$ of both quantities. Using $\log_{10}(a^k) = k\log_{10}(a)$:

$$\log_{10} x = \log_{10}(m) \cdot \log_{10}(n), \qquad \log_{10} y = \log_{10}(n) \cdot \log_{10}(m)$$

These are the same product written in the two possible orders, so $\log_{10} x = \log_{10} y$, which forces $x = y$ since $\log_{10}$ never gives the same output for two different positive inputs.

**Step 4:** Note that the condition m > n never entered the proof.
Nowhere in Step 3 did the size comparison between $m$ and $n$ get used, only that both are positive. This tells us the equality $x = y$ would hold even if we swapped which of $m, n$ is larger, which is a good sign that options (A) and (B), the strict-inequality choices, are traps built around the given condition $m > n$ rather than being genuine consequences of it.

$$\boxed{x = y}$$

**Quick Tip:** Take $\log_{10}$ of both $x$ and $y$ and use the rule $\log_{10}(a^k) = k\log_{10}(a)$.

• • •

**5.**

'If his latest movie had been a commercial success, the actor would have made enough money to sponsor his next movie.'

Based only on the above sentence, which one of the following statements is true?

(A) The actor will certainly sponsor his next movie.

(B) His latest movie was a commercial success.

(C) The actor made enough money from his latest movie.

(D) His latest movie was not commercially successful.

**Correct Answer:** (D) His latest movie was not commercially successful.

**SOLUTION:**

A simple way to handle a conditional sentence like this is to rewrite it in plain words first, without any 'if', and see what it is really claiming. The sentence says: on the assumption that the movie had done well commercially, the actor would have had enough money to fund his next one. The phrase 'would have made' is the giveaway, it is not describing something that happened, it is describing something that was only possible under an assumption.

English marks a past event that genuinely happened with a simple past tense ('the movie was a success' or 'the movie made money'). The moment a sentence switches to 'had been' plus 'would have', it is deliberately stepping outside of what really happened and describing an alternate, unrealised version of events. So whatever sits inside that 'if' clause is being flagged as false in reality.

Applying that to this sentence: the 'if' clause is 'his latest movie had been a commercial success'. Since this clause is marked as an unrealised assumption, reality must be the opposite, the movie was not a commercial success. Nothing in the sentence tells us anything definite about the future (ruling out option A), and nothing in the sentence claims the success or the money actually occurred (ruling out options B and C, both of which restate the false assumption as if it were fact).

The statement that lines up with what the sentence is actually saying, once it is stripped of the hypothetical wrapping, is that the movie did not succeed commercially.

Let's check the other three statements against this same plain-words test. Option A talks about the future ('will certainly sponsor'), but a sentence built entirely around an unrealised past assumption gives no promise about what happens next, so it cannot be picked. Options B and C both restate the assumed condition, that the movie succeeded and that the actor got enough money, as if they were facts, when the whole point of the 'had been / would have' structure is to mark them as not having happened. Only option D avoids repeating the false assumption and instead states its negation, which is the one thing the sentence structure actually guarantees.

**Quick Tip:** 'If + had been, would have' describes an imagined past that did not happen, so the real situation is the opposite of the if-clause.

• • •

**6.**

'My friend and I parted ___ the door ___ the cabin that I had rented ___ the night.'

Choose the option with the correct sequence of words to fill the blanks.

(A) at; of; for

(B) for; at; of

(C) of; for; in

(D) in; of; for

**Correct Answer:** (A) at; of; for

**SOLUTION:**

**Step 1:** Understanding the Concept.
The quickest way to solve this kind of question is to plug each option's three words into the sentence, one full option at a time, and read the whole sentence in your head to see which one sounds like something a native speaker would actually say.

**Step 2:** Key Formula or Approach.
Prepositions are tied to fixed, memorised collocations rather than logical rules, so the fastest check is: does this exact phrase ('parted at', 'door of', 'rented for') sound like a phrase you have heard before, in that order.

**Step 3:** Detailed Explanation.
Option (A) gives: 'parted at the door of the cabin that I had rented for the night.' Each piece is a phrase in common use, 'parted at' a place, 'door of' a building, 'rented for' a stretch of time. The whole sentence flows naturally.
Option (B) gives: 'parted for the door at the cabin that I had rented of the night.' 'Parted for the door' and 'rented of the night' are not phrases anyone uses, this sentence sounds wrong throughout.
Option (C) gives: 'parted of the door for the cabin that I had rented in the night.' 'Parted of' is not standard English, and 'rented in the night' changes the meaning to renting while it was night rather than renting for the duration of the night.
Option (D) gives: 'parted in the door of the cabin that I had rented for the night.' The last two blanks, 'of' and 'for', actually work here, but 'parted in the door' is wrong, a person parts at a doorway, not inside it.

**Step 4:** Final Answer.
Only option (A), 'at; of; for', produces a sentence where every single preposition matches its normal collocation, so that is the correct sequence.

**Quick Tip:** Check each blank against the phrase it sits in: parted 'at' a place, the door 'of' something, rented 'for' a duration.

• • •

**7.**

Five integers are picked from 0 to 20, with possible repetitions, such that their mean is 12, median is 18, and they have a single mode of 20.

Ignoring permutations, the number of ways to pick these five integers is _____

(A) 0

(B) 1

(C) 2

(D) 3

**Correct Answer:** (B) 1

**SOLUTION:**

**Step 1:** Decide how many times 20 must appear.
For 20 to count as 'the single mode' of five numbers, it has to occur more times than any other value in the list, with no other value tying it. Start by asking how many times 20 can realistically appear.

**Step 2:** Rule out 20 appearing three or more times.
If 20 appeared three times, the remaining two numbers would sum to $60 - 60 = 0$, forcing both remaining numbers to be $0$. But then the sorted list would be $0, 0, 20, 20, 20$, and its median (middle value) would be $20$, not $18$. This contradicts the given median, so 20 cannot appear three or more times.

**Step 3:** Rule out 20 appearing once or not at all.
The median is $18$, the middle value of the sorted five numbers, and the fourth and fifth values (both $\geq 18$) are the only positions where a $20$ can sit, since the first two values are $\leq 18 < 20$. If $20$ appears at most once among the five numbers, then no value in the list can beat a frequency of $1$ unless some other number repeats instead, in which case that other number, not $20$, would be the mode. So a lone $20$, or no $20$ at all, cannot give a valid 'single mode of $20$'.

**Step 4:** Conclude 20 appears exactly twice.
Combining Steps 2 and 3, 20 must appear exactly twice, occupying the fourth and fifth positions of the sorted list: $d = e = 20$.

**Step 5:** Solve the remaining two numbers.
The total is $60$, and $c = 18$, $d = e = 20$ account for $58$, leaving $a + b = 2$ for the first two (smallest) numbers, with $a \leq b \leq 18$.
The integer solutions are $(0, 2)$ and $(1, 1)$. But $(1, 1)$ makes $1$ repeat twice as well, tying with $20$'s count of two, which breaks the single-mode requirement. Only $(0, 2)$ survives, since $0$ and $2$ then each appear just once.

**Step 6:** State the unique valid set.
The only five-number multiset meeting every condition is $\{0, 2, 18, 20, 20\}$, and since we count unordered selections, ignoring permutations, this is exactly one valid way to pick the numbers.

$\boxed{1}$

**Quick Tip:** The median fixes the middle number at 18, so 20 can only sit in the fourth and fifth positions, meaning it can appear at most twice.

**8.**

Rishi and Swathi are students of Class 5. Pavan and Tanvi are students of Class 4. Rishi and Pavan are boys. Swathi and Tanvi are girls. The four students played a total of three games of chess. The games were played one after another. A player who lost a game did not participate in any more games. It was observed that:

1. the first game was the only game where two students of the same class played against each other,
2. the students of Class 5 won more games than the students of Class 4, and
3. the boys won two games and the girls won one game.

The student who did not lose any game is _____.

(A) Pavan

(B) Rishi

(C) Swathi

(D) Tanvi

**Correct Answer:** (D) Tanvi

**SOLUTION:**

**Step 1:** Set up win-count variables.
Let $R, S, P, T$ be the number of games won by Rishi, Swathi, Pavan and Tanvi. Three games are played in total, so $R + S + P + T = 3$. Class 5 (Rishi, Swathi) wins more than Class 4 (Pavan, Tanvi), and with only 3 games to split, the only way to keep Class 5 strictly ahead is $R + S = 2$ and $P + T = 1$. Also, boys (Rishi, Pavan) win 2 games total and girls (Swathi, Tanvi) win 1: $R + P = 2$, $S + T = 1$.

**Step 2:** Solve the equations.
From $R + S = 2$ and $R + P = 2$, subtracting gives $S = P$. From $P + T = 1$ and $S + T = 1$, subtracting gives $P = S$ again, so this alone does not fix the values yet. Since $P + T = 1$ with $P, T$ non-negative integers, either $P = 0, T = 1$ or $P = 1, T = 0$.
If $P = 1, T = 0$: then $S = P = 1$, and $R = 2 - S = 1$. This gives $(R, S, P, T) = (1, 1, 1, 0)$.
If $P = 0, T = 1$: then $S = P = 0$, and $R = 2 - S = 2$. This gives $(R, S, P, T) = (2, 0, 0, 1)$.
Both satisfy the arithmetic, so the win totals alone give two candidate patterns; the tournament structure is needed to eliminate one.

**Step 3:** Test (R,S,P,T) = (1,1,1,0) against the structure.
Here Rishi and Swathi both need exactly 1 win each. But in the elimination format, whichever of them loses game 1 (if game 1 is Rishi vs Swathi) is out forever with 0 wins, so they cannot both end up with 1 win, one of them is stuck at 0. This forces game 1 to be Pavan vs Tanvi instead, with Tanvi needing $T = 0$, so Tanvi loses game 1 and Pavan wins it (1 win, matching $P = 1$, so Pavan must lose every later game he plays). Pavan then plays game 2 against a Class 5 student (Rishi or Swathi), and must lose it, so that Class 5 student wins game 2. That same Class 5 student then plays game 3 against the last remaining student, who is the other Class 5 student, since both Rishi and Swathi get used up in games 1 and 2, making game 3 a same-class match. This breaks condition (i), which says only game 1 pairs same-class students. So this whole pattern is structurally impossible.

**Step 4:** Confirm (R,S,P,T) = (2,0,0,1) works.
With $S = 0$ and $P = 0$, both Swathi and Pavan never win a single game, meaning each of them loses the one game they play and is eliminated immediately. Rishi wins 2 games and Tanvi wins 1. Placing Rishi as the winner of games 1 and 2 (beating Swathi, then Pavan) and Tanvi as the winner of game 3 (beating Rishi) satisfies condition (i): game 1 (Rishi vs Swathi) is the only same-class pairing, since games 2 (Rishi vs Pavan) and 3 (Rishi vs Tanvi) are both cross-class. This is consistent, so this is the valid pattern.

**Step 5:** Read off who never lost.
In this valid pattern, Swathi loses game 1, Pavan loses game 2, and Rishi loses game 3 (after having won games 1 and 2). Tanvi only plays game 3 and wins it, so Tanvi is the only student with zero losses.

Tanvi

**Quick Tip:** With four players and three knockout-style games, the win totals split as Class 5 = 2 wins, Class 4 = 1 win; work out who can hold those wins without creating a second same-class game.

• • •

**9.** $P, Q, R, S, X,$ and $Y$ are distinct single-digit whole numbers taking values from 0 to 9.
$PQ$ is a two-digit number with $Q$ in the units place and $P$ in the tens place. Similarly, $RS$ is a two-digit number. It is known that $PQ$ and $RS$ are consecutive numbers and $(PQ)^2 + (RS)^2 = XYP$, with $XYP$ being a three-digit number.
The value of $Y$ is _____.

(A) 4

(B) 5

(C) 6

(D) 7

**Correct Answer:** (C) 6

**SOLUTION:**

This question is a digit-puzzle: we need two consecutive two digit numbers whose squares add up to a three digit number, with a very specific twist, the last digit of that sum has to match the tens digit of the smaller number, and none of the six letters P, Q, R, S, X, Y can repeat a digit.

Start from the size limit. If $a$ and $a + 1$ are the two consecutive numbers, $a^2 + (a + 1)^2$ must stay under 1000 to remain a three digit number. Expanding, $2a^2 + 2a + 1 \leq 999$, so $a^2 + a \leq 499$. Testing $a = 21$: $441 + 21 = 462$, fine. Testing $a = 22$: $484 + 22 = 506$, too big. So only $a = 10$ through $a = 21$ can possibly work.

Now bring in the distinctness rule. Whenever $a$ and $a + 1$ share the same tens digit, for example 14 and 15, the letters P (tens digit of $a$) and R (tens digit of $a + 1$) come out equal, which the question forbids since all six letters must be different digits. Scan the list 10 to 21: every consecutive pair keeps the same tens digit, except the single pair 19 and 20, where the tens digit flips from 1 to 2. That makes $a = 19$, $a + 1 = 20$ the only pair worth checking.

1. **Try PQ = 19, RS = 20:** here $P = 1$, $Q = 9$, $R = 2$, $S = 0$. Squaring and adding: $19^2 = 361$, $20^2 = 400$, sum $= 761$. The units digit of 761 is 1, and that has to equal P. It does, P is 1. So the three digit sum 761 splits as $X = 7$, $Y = 6$, $P = 1$.
2. **Check all six digits:** P=1, Q=9, R=2, S=0, X=7, Y=6. Every one of these six digits, {0,1,2,6,7,9}, is different from the rest, so this assignment satisfies the puzzle completely.
3. **Rule out the flipped order:** if instead PQ=20 and RS=19, then P would be 2, but the sum is still 761 with units digit 1, a contradiction. So the numbers must be read as PQ=19 (smaller) and RS=20 (larger).

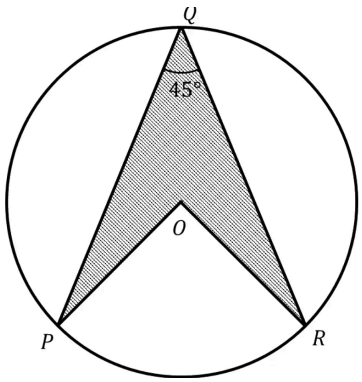
With the full assignment confirmed, the letter Y equals 6, matching option (C).

$$\boxed{Y = 6}$$

**Quick Tip:** Bound the two numbers using $(PQ)^2 + (RS)^2 < 1000$, then notice only one consecutive pair in that range has different tens digits.

• • •

**10.** In the given figure, $P$, $Q$, and $R$ are three points on a circle of radius 10 cm with $O$ as its center, $\overline{PQ} = \overline{RQ}$, and $\angle PQR = 45°$. The figure is representative.
The area of the shaded region $PQRO$ is _____ $cm^2$.




(A) 50
(B) $25\sqrt{2}$
(C) $50\sqrt{2}$
(D) 100

**Correct Answer:** (C) $50\sqrt{2}$

**SOLUTION:**

The shaded patch PQRO is a kite-like quadrilateral made of two triangles glued along the line from the center O to the top point Q. Instead of jumping into a formula, let's build the picture piece by piece.

Because O is the circle's center, OP, OQ and OR are all radii, each 10 cm. We're also told the chords PQ and RQ are equal in length. Two triangles that share a side (OQ) and have their other two pairs of sides equal (OP=OR, QP=QR) must be mirror images of each other, congruent by SSS. That single fact does two things

for us: it tells us OQ splits the quadrilateral exactly down the middle, and it tells us OQ cuts the given 45 degree angle at Q into two 22.5 degree pieces.

1. **Find the apex angle at O:** Look at just the triangle OPQ. Two of its sides, OP and OQ, are both radii of length 10, so the triangle is isosceles, and the base angles at P and Q must match. We already found the angle at Q inside this triangle is 22.5 degrees, so the angle at P is also 22.5 degrees. The angle sitting at O is whatever is left from 180 degrees: $180 - 22.5 - 22.5 = 135$ degrees.
2. **Apply the two-sides-and-included-angle area rule:** For any triangle where two sides $a$ and $b$ meet at a known angle $\theta$, the area is $\frac{1}{2}ab\sin\theta$. Plugging in $a = b = 10$ and $\theta = 135°$: $\frac{1}{2}(10)(10)\sin(135°) = 50 \cdot \frac{\sqrt{2}}{2} = 25\sqrt{2}$ cm$^2$.
3. **Double it for the mirror triangle:** Since triangle OQR is the exact mirror of OPQ, it carries the identical area, $25\sqrt{2}$ cm$^2$. The full shaded region is simply these two matching triangles placed side by side along OQ.

Adding the two pieces: $25\sqrt{2} + 25\sqrt{2} = 50\sqrt{2}$ cm$^2$. This matches option (C), and it also makes sense as a sanity check, since the answer should scale with $r^2 = 100$, and $50\sqrt{2} \approx 70.7$, a value smaller than a full quarter circle of area $25\pi \approx 78.5$, which is reasonable for a thin kite shape cut out of the circle.

$$\boxed{50\sqrt{2}\text{ cm}^2}$$

**Quick Tip:** Split the quadrilateral along $OQ$ into two congruent triangles, each with two 10 cm sides and a 135 degree included angle.

• • •

**11.** For a classification problem, Principal Component Analysis (PCA) has been used to reduce the dimensionality of a feature space from 100 to 10.
Which of the following options is true about the angle $\theta$ between the first and the tenth principal components?

(A) $\theta = 0°$

(B) $\theta = 90°$

(C) $90° < \theta \leq 180°$

(D) $0 < \theta < 90°$

**Correct Answer:** (B) $\theta = 90°$

**SOLUTION:**

This question checks whether we remember what actually makes something a "principal component" in PCA, and the answer comes straight from a linear algebra fact rather than any specific dataset detail.

1. $\theta = 0°$**:** this would mean the 1st and 10th components are the same direction repeated twice. PCA never produces two components in the same direction, each new component is chosen to capture variance that is not already explained by the earlier ones, so this is wrong.
2. $\theta = 90°$**:** PCA finds the eigenvectors of the covariance matrix of the data, and covariance matrices are always symmetric. For a symmetric matrix, eigenvectors tied to different eigenvalues are always at right

angles to each other, no exceptions. The 1st and 10th principal components use different eigenvalues (that is literally why one is ranked higher than the other), so they must be orthogonal. This fits.

3. $90° < \theta \leq 180°$**:** this range describes an obtuse relationship, which is not what orthogonal eigenvectors give us, this is a distractor aimed at people who forget the exact right-angle property.
4. $0 < \theta < 90°$**:** this describes a generic acute angle, again not matching the guaranteed perpendicularity of eigenvectors from a symmetric matrix.

Worth noting: this orthogonality holds regardless of how many dimensions we keep. Going from 100 down to 10 just means we throw away 90 of the eigenvectors (the ones with the smallest eigenvalues), it does not change the geometric relationship among the ones we keep. So whether we compare component 1 and component 2, or component 1 and component 10, the answer is the same right angle.

The correct choice is (B), $\theta = 90°$.

$$\boxed{\theta = 90°}$$

**Quick Tip:** Principal components are eigenvectors of a symmetric covariance matrix, and such eigenvectors are always mutually orthogonal.

• • •

**12.** Consider that you are training a classifier for a 10-class classification problem. Each input is represented as a 512-dimensional vector. There are 1000 samples, out of which the first 100 will be used for testing.
Let Leave-One-Out-Cross-Validation (LOOCV) be used for selection of the classifier model before testing.
Which of the following options is the correct number of validation splits that will be generated?

(A) 10

(B) 512

(C) 900

(D) 1000

**Correct Answer:** (C) 900

**SOLUTION:**

The trick in this question is separating what actually feeds into LOOCV from numbers that are just there to distract, like the class count and the feature dimension.

Start with what LOOCV means. Leave-One-Out-Cross-Validation takes whatever pool of samples it is given, call its size $m$, and builds exactly $m$ train/validation splits, one for every sample in that pool. In split $i$, sample $i$ becomes the single validation point and the remaining $m - 1$ samples form the training set. So the number of splits is always just $m$, the size of the working pool, nothing more complicated than that.

Now figure out what $m$ actually is here. The dataset has 1000 samples total. The question says the first 100 are set aside for testing, and model selection (the step that uses LOOCV) happens before testing even starts. That phrasing matters: it tells us the 100 test samples never enter the LOOCV process at all, they sit untouched until the very end. So the pool LOOCV actually works with is $1000 - 100 = 900$ samples.

Plugging $m = 900$ into the rule from above, LOOCV generates 900 splits.

Let's summarize why the distractors don't fit:

- 10 is the number of output classes, this affects the classifier's output layer, not how cross-validation partitions data.
- 512 is the dimensionality of each input vector, also irrelevant to how many samples get held out one at a time.
- 1000 would only be right if the test samples were included in the LOOCV pool, but the question explicitly excludes them.

So the correct answer is 900, option (C).

$$\boxed{900}$$

**Quick Tip:** LOOCV always generates one split per sample in its working pool; here that pool excludes the 100 test samples, leaving 900.

• • •

**13.** Which of the following algorithms is NOT an example of uninformed search?

(A) Breadth First Search

(B) Depth First Search

(C) A* Search

(D) Depth-limited Search

**Correct Answer:** (C) A* Search

**SOLUTION:**

The question is testing a basic but important AI distinction: blind (uninformed) search versus heuristic-guided (informed) search. The quickest way to answer is to check each option for whether it uses a heuristic.

1. **Breadth First Search:** expands all nodes at the current depth before moving deeper, purely based on the tree structure. No heuristic is used anywhere, so this is uninformed search.
2. **Depth First Search:** follows one path as deep as it can go before backtracking to try another branch. Again, this decision is based only on structure, not on any estimate of closeness to the goal, so it is uninformed search.
3. **A* Search:** combines the actual path cost so far, $g(n)$, with a heuristic estimate of the remaining cost, $h(n)$, into a total score $f(n) = g(n) + h(n)$, and always expands the node with the lowest $f(n)$ first. Because it leans on that heuristic $h(n)$ to make smarter choices, A* is the odd one out, it is informed search, not uninformed.
4. **Depth-limited Search:** this is just Depth First Search capped at a maximum depth so it does not run forever down one branch. The depth cutoff is a structural limit, not a heuristic estimate of goal distance, so it still counts as uninformed search.

Three of the four options, BFS, DFS, and Depth-limited Search, never look at any estimate of how close a state is to the goal, they just follow the shape of the search tree. Only A* Search brings in a heuristic function

to steer the search, which is exactly what separates informed from uninformed methods.

So the one algorithm that is NOT uninformed search is A* Search, option (C).

**A* Search**

**Quick Tip:** Uninformed search uses no heuristic; A* uses $f(n) = g(n) + h(n)$, so it belongs to informed search.

• • •

**14.** Which of the following statements is NOT true? (The names of the predicates are intuitive.)

(A) $\forall x \forall y\ Classmate(x, y) \Rightarrow Classmate(y, x)$

(B) $\forall x\ Likes(x, Icecream) \Rightarrow \neg\exists x\ \neg Likes(x, Icecream)$

(C) "Each king is a person" is equivalent to $\forall x\ IsKing(x) \wedge IsPerson(x)$

(D) "All humans are mortal" is equivalent to $\forall x\ IsHuman(x) \Rightarrow IsMortal(x)$

**Correct Answer:** (C) "Each king is a person" is equivalent to $\forall x\ IsKing(x) \wedge IsPerson(x)$

**SOLUTION:**

This question is really about one recurring mistake in first-order logic translation: mixing up $\wedge$ (and) with $\Rightarrow$ (implies) when turning a universal English sentence into a formula. Let's walk through all four options.

1. **Classmate symmetry:** $\forall x \forall y\ Classmate(x, y) \Rightarrow Classmate(y, x)$. Being classmates is a two-way relationship by definition, if x shares a class with y, y shares that same class with x. This formula states exactly that symmetry correctly, so it is true.
2. **The ice cream tautology:** $\forall x\ Likes(x, Icecream) \Rightarrow \neg\exists x\ \neg Likes(x, Icecream)$. Notice that $\neg\exists x\ \neg Likes(x, Icecream)$ is just another way of writing $\forall x\ Likes(x, Icecream)$ (saying "there is no one who dislikes it" is the same as saying "everyone likes it"). So this statement has the shape $P \Rightarrow P$, which is always true regardless of the actual world, a logical tautology. True.
3. **The king-person conjunction:** the English says "Each king is a person", which is a conditional claim, it only talks about objects that ARE kings. The correct FOL form is $\forall x\ IsKing(x) \Rightarrow IsPerson(x)$. But the option instead writes $\forall x\ IsKing(x) \wedge IsPerson(x)$, using AND. That formula says literally every object in the universe, whether it is a king or not, must be both a king and a person, which is a far stronger and generally false claim, and it does not match the original English sentence at all. This equivalence claim is broken, so this statement is NOT true, this is our answer.
4. **Humans and mortality:** $\forall x\ IsHuman(x) \Rightarrow IsMortal(x)$ is the textbook correct translation of "All humans are mortal", using the right implication structure. True.

Three out of four options hold up under inspection, only the king-person statement swaps the implication for a conjunction, breaking the claimed equivalence. So the answer is option (C).

**Option (C)**

**Quick Tip:** "Every A is a B" translates using implication ($A(x) \Rightarrow B(x)$), never a conjunction ($A(x) \wedge B(x)$).

• • •

**15.** Consider that the quick sort algorithm is used to sort an array of $n$ distinct randomly ordered elements. In every call, the pivot is chosen as the first element of the current subarray.
Let $T(n)$ denote the expected time to sort the array. Assume that the time to partition is linear in the size of the current subarray.
Which of the following recurrence relations correctly represents $T(n)$ in this scenario?

(A) $T(n) = T(1) + T(n-1) + O(n)$

(B) $T(n) = T\left(\frac{n}{4}\right) + T\left(\frac{3n}{4}\right) + O(n)$

(C) $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$

(D) $T(n) = \frac{1}{n}\sum_{k=0}^{n-1}[T(k) + T(n-k-1)] + O(n)$

**Correct Answer:** (D) $T(n) = \frac{1}{n}\sum_{k=0}^{n-1}[T(k) + T(n-k-1)] + O(n)$

**SOLUTION:**

The key word in this question is "expected", not "worst case" or "best case". That single word tells us we need to average over every possible way the pivot could split the array, rather than picking one fixed scenario.

Here is the setup: quicksort always grabs the first element of the current subarray as pivot. If the array were sorted or reverse sorted, that would always give the smallest or largest element, a very lopsided split of size $1$ and $n-1$. But this array is randomly ordered, so nothing forces the first element to be extreme, it is just as likely to be any rank in the array.

Formally, after we partition around the pivot, say $k$ elements land in the left part and $n-1-k$ land in the right part. Because the ordering is random, $k$ can be $0, 1, 2, \ldots, n-1$ with equal probability $\frac{1}{n}$ each, there's no reason for any particular split to be favored over another.

Now build the expected cost. For a fixed $k$, the total work is $T(k)$ to sort the left part, $T(n-k-1)$ to sort the right part, plus the linear partition cost. Since we do not know $k$ in advance and every value is equally likely, we take the average across all $n$ possibilities:

$$T(n) = \frac{1}{n}\sum_{k=0}^{n-1}\Big(T(k) + T(n-k-1)\Big) + O(n)$$

Let's summarize why the other three do not fit:

- $T(n) = T(1) + T(n-1) + O(n)$ is the worst-case recurrence, forcing the smallest possible split every time, which only happens for sorted or adversarial input, not random input.
- $T(n) = T(n/4) + T(3n/4) + O(n)$ and $T(n) = 2T(n/2) + O(n)$ both lock in one specific fixed ratio for every single partition. A random array with a first-element pivot does not guarantee any fixed ratio call after call, the ratio itself is random and must be averaged over, not assumed constant.

Averaging over all $n$ equally likely split points is what genuinely captures "expected time" here, and that is exactly the recurrence in option (D).

$$T(n) = \frac{1}{n}\sum_{k=0}^{n-1}[T(k) + T(n-k-1)] + O(n)$$

**Quick Tip:** Random input means the pivot's rank after partitioning is uniformly distributed over 0 to n-1, so average $T(k) + T(n-1-k)$ over all k.

• • •

**16.** Consider the given Python program.

```
def append_to_lst(val, lst=[]):
    lst.append(val)
    return lst

print(append_to_lst(1))
print(append_to_lst(2))
print(append_to_lst(3, []))
```

Which of the following is the correct output of this program?

(A) [1]
[2]
[3]

(B) [1]
[1, 2]
[3]

(C) [1]
[2]
[1, 2, 3]

(D) [1]
[1, 2]
[1, 3]

**Correct Answer:** (B) [1]
[1, 2]
[3]

**SOLUTION:**

This is the classic Python "mutable default argument" gotcha, and the cleanest way to solve it is to track exactly one thing: which list object each call is actually appending to, the shared default, or a fresh one passed in explicitly.

Python evaluates *lst* = [] exactly once, when the function is defined, not once per call. That single list object then lives on as long as the function exists, and any call that skips the second argument automatically falls back to reusing that same object, carrying forward whatever was appended to it in earlier calls.

1. **print(append_to_lst(1)):** no second argument given, so this call uses the shared default list, which is empty at this point. Appending 1 changes the shared list from **[]** to **[1]**, and **[1]** is printed. From now on, the shared default object itself is **[1]**, not empty.
2. **print(append_to_lst(2)):** again no second argument, so Python reaches for the very same shared list, but that list is no longer empty, it already holds **[1]** from step 1. Appending 2 makes it **[1, 2]**, and that is what prints.
3. **print(append_to_lst(3, [])):** this time an explicit empty list **[]** is passed in. This is a completely different, brand new list object, unrelated to the shared default one. Appending 3 to this fresh list gives **[3]**, which prints. The shared default list quietly stays at **[1, 2]** behind the scenes, but this call never touches it.

Stacking the three printed results in order: **[1]**, then **[1, 2]**, then **[3]**. That sequence corresponds to option (B).

A quick way to remember this for future code review: never use a mutable object like a list or dict as a default argument value unless you actually want it shared and remembered across calls, use ***None*** as the default and create a fresh list inside the function body instead.

$$\boxed{[1],\ [1,2],\ [3]}$$

**Quick Tip:** A mutable default argument like $lst = []$ is created once at function definition and reused across calls that do not pass their own list.

• • •

**17.** Let $R(A, B, C, D, E)$ be a relational schema with functional dependency set $F = \{A \to BC,\ CD \to E,\ E \to A\}$. Which of the following statements is correct?

(A) $AD$, $ED$ and $CD$ are the only candidate keys of $R$.

(B) $AD$ and $ED$ are the only candidate keys of $R$.

(C) $A$, $E$ and $CD$ are the only candidate keys of $R$.

(D) $A$ and $CD$ are the only candidate keys of $R$.

**Correct Answer:** (A) $AD$, $ED$ and $CD$ are the only candidate keys of $R$.

**SOLUTION:**

**Step 1:** Find which attribute must be in every key.
Look at the right-hand side of every FD in $F = \{A \to BC,\ CD \to E,\ E \to A\}$. The attributes that appear on some right side are $B$, $C$ (from $A \to BC$), $E$ (from $CD \to E$), and $A$ (from $E \to A$). $D$ never appears on the right side of any FD.
Since no FD can ever produce $D$ from other attributes, $D$ can only be present in the closure of a set if $D$ was already put into that set to begin with. So $D$ must be part of every single candidate key of $R$.

**Step 2:** Combine $D$ with each other attribute and test the closure.
$D$ alone gives $D^+ = \{D\}$, too small, so we must add one more attribute to $D$ and check if that reaches all of $R$.
$\{A, D\}$: $A \to BC$ gives $\{A, B, C, D\}$, then $CD \to E$ (both $C, D$ present) gives $\{A, B, C, D, E\} = R$. So $AD$ works.
$\{C, D\}$: $CD \to E$ gives $\{C, D, E\}$, then $E \to A$ gives $\{A, C, D, E\}$, then $A \to BC$ gives $\{A, B, C, D, E\} = R$. So $CD$

works.
$\{D, E\}$: $E \rightarrow A$ gives $\{A, D, E\}$, then $A \rightarrow BC$ gives $\{A, B, C, D, E\} = R$. So $ED$ works.
$\{B, D\}$: $B$ never appears on the left of any FD, so this closure just stays $\{B, D\}$, which is not all of $R$. So $BD$ does not work.

**Step 3:** Check minimality.
For $AD$, $CD$, $ED$ to be candidate keys and not just superkeys, no proper subset (that is, $D$ alone, or the other single attribute alone) can already reach $R$. We already know $D^+ = \{D\}$, $A^+ = \{A, B, C\}$, $C^+ = \{C\}$, $E^+ = \{A, B, C, E\}$, none of which equal $R$. So each pair is minimal.

**Step 4:** Conclusion.
Since $D$ must be in every key, and only pairing $D$ with $A$, $C$, or $E$ gives a closure equal to $R$, the complete list of candidate keys is exactly $AD$, $CD$, $ED$.

Option (A): $AD,\ ED,\ CD$

**Quick Tip:** Find which attribute never appears on the right side of any FD, that attribute must be in every candidate key. Then pair it with each other attribute and check whether the closure covers all of $R$.

• • •

**18.** Consider that the visualization of a 3-dimensional data cube is showing Sales Quantity for each combination of the attributes Product Type, Month and Country.
From this, if we want to further visualize the Sales Quantity for each combination of Product Type, Month and State, which of the following OLAP operations should be performed?

(A) Slicing
(B) Dicing
(C) Roll-up
(D) Drill-down

**Correct Answer:** (D) Drill-down

**SOLUTION:**

This question tests the four basic OLAP (Online Analytical Processing) operations used on a multidimensional data cube: slicing, dicing, roll-up and drill-down. Let's check each option against what the question describes.

1. **Slicing**: this operation fixes a single dimension at one value and views the remaining lower-dimensional cube, such as looking only at Country = "USA". The question does not fix any dimension to one value, so this does not fit.
2. **Dicing**: this operation selects a sub-cube by fixing a specific range of values on two or more dimensions at once. The question is not restricting to particular values of any dimension either, so this does not fit.
3. **Roll-up**: this operation climbs up a concept hierarchy, aggregating detailed data into a more summarized form, for example moving from State-wise sales to Country-wise sales. The question moves

in the opposite direction, from Country to State, so roll-up is wrong here.

4. **Drill-down**: this operation is the reverse of roll-up. It moves down a concept hierarchy from a summarized level to a more detailed level, for example Country breaking down into its States. The question exactly describes going from Country-level sales data to State-level sales data, which is a move to finer detail.

Since Country and State belong to the same geography hierarchy, with State being a lower, more detailed level than Country, moving from Country to State is a Drill-down operation.

Let's summarize:

- Slicing and Dicing change which values of a dimension are shown, not the level of detail.
- Roll-up moves to a coarser level, Drill-down moves to a finer level, and here the request is for finer detail.

So the operation needed is Drill-down.

**Quick Tip:** Country and State belong to the same geography hierarchy, with State being more detailed than Country. Moving to a more detailed level of an existing hierarchy is Drill-down.

• • •

**19.** Let $M$ be a randomly chosen non-empty subset of $S = \{1, 2, 3, \ldots, 2026\}$.
Which of the following is the probability that the product of all the elements of $M$ is even?

(A) $\frac{2^{1013}(2^{1013} - 1)}{2^{2026}}$

(B) $\frac{2^{1013}}{2^{2026}}$

(C) $\frac{2^{1013}(2^{1013} - 1)}{2^{2026} - 1}$

(D) $\frac{1}{2^{2026} - 1}$

**Correct Answer:** (C) $\frac{2^{1013}(2^{1013} - 1)}{2^{2026} - 1}$

**SOLUTION:**

**Step 1:** Build the answer directly instead of subtracting.
Instead of finding the complement (product odd) and subtracting, we can directly count the subsets whose product is even, by using the multiplication principle.

**Step 2:** Split into 1013 odd numbers and 1013 even numbers.
$S = \{1, \ldots, 2026\}$ contains $1013$ odd numbers and $1013$ even numbers. A subset $M$'s product is even exactly when $M$ contains at least one even number, no matter what odd numbers it also contains.

**Step 3:** Choose the odd part and the even part independently.
We can build any subset $M$ with product even in two independent choices:
(i) choose any subset of the $1013$ odd numbers, this part is completely free, giving $2^{1013}$ possibilities

(including choosing none of the odd numbers),
(ii) choose a non-empty subset of the **1013** even numbers (it must be non-empty, otherwise there would be no even factor at all), giving $2^{1013} - 1$ possibilities.
Since these two choices are independent and together completely determine $M$, by the multiplication principle the number of subsets with an even product is:

$$2^{1013} \times (2^{1013} - 1)$$

**Step 4:** Divide by the total number of non-empty subsets.
The total number of ways to choose a non-empty $M$ from the **2026** elements is $2^{2026} - 1$. So:

$$P(\text{product even}) = \frac{2^{1013}(2^{1013} - 1)}{2^{2026} - 1}$$

**Step 5:** Sanity check.
This matches exactly what complementary counting would give, which confirms the direct constructive count is consistent. The key idea is that fixing "at least one even number chosen" while letting the odd numbers be completely free avoids double counting or missing any case.

$$\boxed{\frac{2^{1013}(2^{1013} - 1)}{2^{2026} - 1}}$$

**Quick Tip:** The product is odd only when every chosen element is odd. Count non-empty all-odd subsets, subtract that fraction from 1, keeping the denominator as $2^{2026} - 1$ since $M$ must be non-empty.

• • •

**20.** Suppose that a computer program provides a non-negative and integer-valued random solution to the equation $n_1 + n_2 + n_3 + n_4 = 20$.
Which of the following is the probability that all of $n_1, n_2, n_3, n_4$ in the provided solution are positive?

(A) $\binom{19}{3} / \binom{23}{3}$
(B) $\binom{20}{4} / \binom{24}{4}$
(C) $\binom{20}{3} / \binom{23}{3}$
(D) $\binom{19}{4} / \binom{24}{4}$

**Correct Answer:** (A) $\binom{19}{3} / \binom{23}{3}$

**SOLUTION:**

**Step 1:** Picture the balls-and-dividers model directly.
Instead of using the algebraic substitution, we can picture the total non-negative case using a row of **20**

identical balls (representing the total sum) plus 3 dividers, which split the row into 4 groups, representing $n_1, n_2, n_3, n_4$.
In total we are arranging 20 balls and 3 dividers in a row, that is 23 objects, and we just need to choose which 3 of the 23 positions are dividers. This directly gives $\binom{23}{3}$ total arrangements, matching the total sample space.

**Step 2:** Force each group to be non-empty using the gaps method.
For all four parts to be positive (no group empty), think of the 20 balls placed in a row first. Between consecutive balls there are 19 internal gaps (a gap is a place strictly between two balls, not at the two ends). To split the 20 balls into 4 non-empty positive groups, we must place the 3 dividers into 3 of these 19 internal gaps (never at the very ends, and never two dividers in the exact same gap, otherwise some group would be empty).
This gives $\binom{19}{3}$ ways to choose which 3 of the 19 gaps get a divider, which is exactly the count of solutions where every $n_i \geq 1$.

**Step 3:** Form the probability.

$$P(\text{all positive}) = \frac{\text{favourable arrangements}}{\text{total arrangements}} = \frac{\binom{19}{3}}{\binom{23}{3}}$$

**Step 4:** Cross check with the substitution method.
This gap-counting picture gives the same $\binom{19}{3}$ and $\binom{23}{3}$ that the standard $m_i = n_i - 1$ substitution gives, which confirms the answer through a different, purely visual argument.

$$\boxed{\frac{\binom{19}{3}}{\binom{23}{3}}}$$

**Quick Tip:** Total non-negative solutions use $\binom{23}{3}$ (stars and bars with 20 items, 4 bins). For all positive, substitute $m_i = n_i - 1$ and solve $\sum m_i = 16$, giving $\binom{19}{3}$.

• • •

**21.** Let $M = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$ be a $2 \times 2$ matrix, where $\theta = \frac{2\pi}{5}$, and $I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.
Which of the following options is equal to $M^{2026}$?

(A) $M^2$

(B) $M$

(C) $M^{-1}$

(D) $I_2$

**Correct Answer:** (B) $M$

**SOLUTION:**

**Step 1:** Represent the rotation using a complex exponential.
A rotation matrix by angle $\theta$ corresponds exactly to multiplication by the complex number $e^{i\theta} = \cos\theta + i\sin\theta$ on the complex plane, if we think of a vector $(x, y)$ as the complex number $x + iy$. Multiplying by $M$ once is the same as multiplying by $e^{i\theta}$ once.

**Step 2:** Translate $M^{2026}$ into this complex number language.
Applying $M$ a total of $2026$ times corresponds to multiplying by $e^{i\theta}$ a total of $2026$ times, which is:

$$\left(e^{i\theta}\right)^{2026} = e^{i\cdot 2026\theta}$$

**Step 3:** Reduce the exponent modulo $2\pi$.
Since $\theta = \frac{2\pi}{5}$, we get $2026\theta = 2026 \times \frac{2\pi}{5}$. A complex exponential $e^{i\phi}$ repeats with period $2\pi$ in $\phi$, that is, $e^{i(\phi+2\pi k)} = e^{i\phi}$ for any integer $k$.
So we only need $2026\theta \bmod 2\pi$. Because $\theta = \frac{2\pi}{5}$, this is equivalent to reducing $2026 \bmod 5$.
$2026 = 5(405) + 1$, so $2026 \equiv 1 \pmod 5$.

**Step 4:** Simplify.

$$e^{i\cdot 2026\theta} = e^{i(405\cdot 2\pi+\theta)} = e^{i\cdot 405\cdot 2\pi} \cdot e^{i\theta} = 1 \cdot e^{i\theta} = e^{i\theta}$$

Since $e^{i\cdot 405\cdot 2\pi} = 1$ ($405$ full turns bring us back to the start).

**Step 5:** Translate back to matrix form.
$e^{i\cdot 2026\theta} = e^{i\theta}$ means $M^{2026}$ is the same rotation as $M^1$, so:

$$M^{2026} = M$$

$$\boxed{M^{2026} = M}$$

**Quick Tip:** $M$ is a rotation matrix by $\theta$, so $M^k$ is rotation by $k\theta$, which only depends on $k\theta \bmod 2\pi$, i.e., on $k \bmod 5$ here. Find $2026 \bmod 5$.

• • •

**22.** Consider a set $S_1 = \{x = (x_1, x_2, x_3)^T \in \mathbb{R}^3 \mid x^T x \leq 16\}$. Let $S_2$ be another set which is a subspace of $\mathbb{R}^3$ with dimension two.
Which of the following gives the area of $S_1 \cap S_2$?

(A) $16\pi$

(B) $4\pi$

(C) $4\pi^2$

(D) $16\pi^2$

**Correct Answer:** (A) $16\pi$

**SOLUTION:**

**Step 1:** Set up coordinates inside the plane.
Since $S_2$ is a two-dimensional subspace of $\mathbb{R}^3$, we can pick an orthonormal basis $u_1, u_2$ for $S_2$ (two mutually perpendicular unit vectors that span the plane). Any point $x$ in $S_2$ can then be written as:

$$x = a\,u_1 + b\,u_2, \quad a, b \in \mathbb{R}$$

**Step 2:** Rewrite the ball condition in these coordinates.
Because $u_1, u_2$ are orthonormal ($u_1 \cdot u_1 = 1$, $u_2 \cdot u_2 = 1$, $u_1 \cdot u_2 = 0$), the squared length of $x$ works out cleanly:

$$x^T x = (au_1 + bu_2)^T(au_1 + bu_2) = a^2(u_1^T u_1) + 2ab(u_1^T u_2) + b^2(u_2^T u_2) = a^2 + b^2$$

So the condition $x^T x \leq 16$ becomes, purely in terms of the plane's own coordinates $(a, b)$:

$$a^2 + b^2 \leq 16$$

**Step 3:** Recognize the shape in the $(a, b)$ plane.
This is precisely the equation of a filled disk of radius $4$ centered at the origin of the $(a, b)$ coordinate system. Since $(a, b)$ are just orthonormal coordinates within $S_2$, distances measured in $(a, b)$ match distances measured in $\mathbb{R}^3$, so this disk is a true, undistorted disk of radius $4$.

**Step 4:** Compute the area.

$$\text{Area} = \pi r^2 = \pi(4)^2 = 16\pi$$

**Step 5:** Conclusion.
This coordinate-based derivation confirms, without relying on intuition alone, that intersecting a solid ball of radius 4 through its center with any plane through that center always produces a disk of the same radius 4.

$$\boxed{16\pi}$$

**Quick Tip:** $S_1$ is a solid ball of radius 4 centered at the origin, $S_2$ is a plane through the origin. Their intersection is a disk of radius 4, so use area $= \pi r^2$.

• • •

**23.** In the following table, the Task column lists a few tasks related to machine learning. The Algorithm column lists a few algorithms.
Each entry "t" from the Task column is to be matched with an appropriate entry "a" from the Algorithm column such that the task "t" can be solved using the algorithm "a". Denote such a match as t:a.

| Task | Algorithm |
|---|---|
| T1 - Clustering | A1 - Markov Chain Monte Carlo |
| T2 - Classification | A2 - K-Medoid |
| T3 - Sampling | A3 - Linear Discriminant Analysis |
| T4 - Feature Extraction | A4 - Naive Bayes |

Which of the following options is/are the correct matching(s)?

(A) T1:A4, T2:A3, T3:A1, T4:A2

(B) T1:A2, T2:A4, T3:A1, T4:A3

(C) T1:A3, T2:A4, T3:A1, T4:A2

(D) T1:A4, T2:A2, T3:A1, T4:A3

**Correct Answer:** (B) T1:A2, T2:A4, T3:A1, T4:A3

**SOLUTION:**

This question checks whether each algorithm listed is correctly paired with the machine learning task it is actually built to solve. Let's look at what each algorithm is used for and eliminate the wrong pairings.

1. **A1 - Markov Chain Monte Carlo**: this is a sampling technique used to draw samples from probability distributions that are hard to sample from directly, so A1 belongs with T3 (Sampling), not with any other task.
2. **A2 - K-Medoid**: this algorithm groups data points into clusters around representative medoid points, so A2 belongs with T1 (Clustering), it has no role in classification or feature extraction.
3. **A3 - Linear Discriminant Analysis**: this technique finds directions that best separate classes and is mainly used to reduce the number of features while keeping class-separating information, so A3 belongs with T4 (Feature Extraction).
4. **A4 - Naive Bayes**: this is a probabilistic classifier that predicts a class label using Bayes' theorem, so A4 belongs with T2 (Classification).

Putting these correct pairs together gives T1:A2, T2:A4, T3:A1, T4:A3, which is exactly option (B). Every other option swaps at least one algorithm into the wrong task, for instance putting Naive Bayes against Clustering or K-Medoid against Classification, both of which contradict what these algorithms are designed to do.

Let's summarize:

- Clustering pairs with K-Medoid, Classification pairs with Naive Bayes.
- Sampling pairs with MCMC, Feature Extraction pairs with LDA.

So the only fully correct matching is option (B).

**Quick Tip:** K-Medoid clusters, Naive Bayes classifies, MCMC samples, and LDA extracts/reduces features. Match each task to the algorithm built for that exact purpose.

• • •

**24.** Sentence $X$ is said to entail Sentence $Y$ if whenever $X$ is $TRUE$, $Y$ also must hold $TRUE$.
Which of the following statements is/are correct if $X$ entails $Y$?

(A) $X \Rightarrow Y$

(B) $X \wedge \neg Y$ is $FALSE$

(C) if $X$ then $Y$

(D) if $Y$ then $X$

**Correct Answer:** (A) $X \Rightarrow Y$ ; (B) $X \wedge \neg Y$ is $FALSE$ ; (C) if $X$ then $Y$

**SOLUTION:**

**Step 1:** Build a truth table for the four possible cases.
$X$ and $Y$ can each be $TRUE$ or $FALSE$, giving four combinations. "$X$ entails $Y$" rules out exactly one of these four: the case $X = TRUE, Y = FALSE$. The remaining three combinations ($TT$, $FT$, $FF$) are all allowed under entailment.

**Step 2:** Evaluate each option in all three allowed combinations.
Case $X = T, Y = T$: $X \Rightarrow Y$ is $T$; $X \wedge \neg Y = T \wedge F = F$, so "is FALSE" holds; "if X then Y" holds since both are true; "if Y then X" also happens to hold here.
Case $X = F, Y = T$: $X \Rightarrow Y$ is $T$ (false antecedent makes implication true); $X \wedge \neg Y = F \wedge F = F$, so "is FALSE" holds; "if X then Y" holds vacuously; "if Y then X" FAILS here, because $Y$ is true but $X$ is false.
Case $X = F, Y = F$: $X \Rightarrow Y$ is $T$; $X \wedge \neg Y = F \wedge T = F$, so "is FALSE" holds; "if X then Y" holds vacuously; "if Y then X" holds vacuously here (Y is false).

**Step 3:** Read off which statements hold in every allowed case.
$X \Rightarrow Y$: true in all three allowed cases, always holds under entailment.
$X \wedge \neg Y$ is FALSE: true in all three allowed cases, always holds under entailment.
"if X then Y": true in all three allowed cases (this phrase means the same thing as $X \Rightarrow Y$), always holds.
"if Y then X": this FAILS in the case $X = F, Y = T$, which is a case entailment does allow, so this statement is not guaranteed.

**Step 4:** Conclusion.
Only the case $X = F, Y = T$ is needed as a single counterexample to rule out option (D), since entailment permits that case but "if Y then X" breaks there. Options (A), (B), (C) hold in all cases entailment allows.

**Options (A), (B) and (C) are correct, (D) is not**

**Quick Tip:** Entailment rules out only $X = TRUE, Y = FALSE$. Check each statement against this and note that "if Y then X" is the converse, which is not guaranteed.

• • •

**25.** You are given the following Pre-order and In-order traversals of a Binary Tree ***T*** with nodes E, F, G, P, Q, R, S.

Pre-order: P Q S E R F G
In-order: S Q E P F R G

Which of the following statements is/are true about the Binary Tree ***T***?

(A) Node P is the root of T

(B) The Post-order traversal of T is: S E Q F G R P

(C) Node Q has only one child

(D) The left subtree of node R contains the node G

**Correct Answer:** (A) Node P is the root of T ; (B) The Post-order traversal of T is: S E Q F G R P

**SOLUTION:**

The trick with pre-order and in-order pairs is that the root always sits at the front of the pre-order list, and it splits the in-order list into a left part and a right part. Apply that rule repeatedly to draw the tree, then check each statement against the drawing.

Pre-order: P Q S E R F G. In-order: S Q E P F R G.

The root is ***P***, the first pre-order entry. In the in-order list, ***P*** sits between "S Q E" (left side) and "F R G" (right side), so the left subtree holds nodes {S, Q, E} and the right subtree holds {F, R, G}.

Matching the pre-order list to these two groups, the left group takes the next 3 letters after P, which is "Q S E", and the right group takes what remains, "R F G".

For the left piece (pre-order Q S E, in-order S Q E): the root is ***Q***, with S to its left and E to its right in the in-order list, so ***Q*** has two children, left = S and right = E.

For the right piece (pre-order R F G, in-order F R G): the root is ***R***, with F to its left and G to its right, so ***R*** has two children, left = F and right = G.

So the tree looks like this: P is the root; the left child of P is Q, with left child S and right child E; the right child of P is R, with left child F and right child G.

Now check each statement:

1. **(A) P is the root:** confirmed directly above, so this is correct.
2. **(B) Post-order is S E Q F G R P:** post-order visits left child, right child, then the node itself. For the Q-subtree that gives S, E, Q. For the R-subtree that gives F, G, R. Putting the root P last gives S E Q F G R P, which matches, so this is correct.
3. **(C) Q has only one child:** Q has both S and E as children, so this is wrong.
4. **(D) G lies in the left subtree of R:** R's left subtree is just F; G is R's right child, so this is wrong.

The correct statements are (A) and (B).

(A) and (B)

**Quick Tip:** Use the first element of the pre-order list as the root, then split the in-order list around it to find the left and right subtrees.

• • •

**26.** Consider two relations $r$ and $s$ defined on the relational schemas $R(A, B)$ and $S(E, C)$, respectively. $A$ is the primary key of $R$ and $E$ is a foreign key of $S$ referencing $A$ in $R$.

Which of the following operations will NEVER violate the foreign key constraint?

(A) Inserting records into relation r

(B) Deleting records from relation s

(C) Deleting records from relation r

(D) Inserting records into relation s

**Correct Answer:** (A) Inserting records into relation r ; (B) Deleting records from relation s

**SOLUTION:**

Think of $R$ as the "parent" table and $S$ as the "child" table, since $S.E$ points back to $R.A$. A foreign key constraint only cares about one thing: every $E$ value stored in $S$ must have a matching $A$ value sitting in $R$. Go through each operation with that single rule in mind.

1. **Inserting records into $r$:** this only grows the pool of valid $A$ values in the parent table. No existing $S$ row loses its match, so nothing can break. Always safe.
2. **Deleting records from $s$:** this only removes rows from the child table. Fewer rows in $S$ can never cause a reference to go missing, since you are not creating any new $E$ values that need to be checked. Always safe.
3. **Deleting records from $r$:** if the row you delete from the parent table has the same $A$ value that some child row's $E$ points to, that child row is now referencing a value that no longer exists. That is exactly what the constraint forbids, so this can break things.
4. **Inserting records into $s$:** a brand new child row might carry an $E$ value that was never a valid $A$ value in the parent table to begin with. That new row would violate the constraint the moment it is inserted. So this can also break things.

Only the two "growing the parent" and "shrinking the child" operations, inserting into $r$ and deleting from $s$, are guaranteed never to violate the foreign key constraint.

(A) and (B)

**Quick Tip:** Think about which side of the reference, the referenced primary key or the referencing foreign key, each operation changes.

• • •

**27.** Let $f(x) = x^3 - 3x^2 + 2$ be a function defined on $(-1, 3]$.

Which of the following statements is/are correct?

(A) $f(x)$ has exactly two roots in $[-0.9, 0]$.

(B) $f(x)$ has a minimum at 2 only.

(C) $f(x)$ has maximum at 0 only.

(D) $f(x)$ has a root at 1.

**Correct Answer:** (B) $f(x)$ has a minimum at 2 only. ; (D) $f(x)$ has a root at 1.

**SOLUTION:**

Work with a value table for $f(x) = x^3 - 3x^2 + 2$ rather than pure algebra, it makes the true/false checks easier to see.

First locate where the slope is zero: $f'(x) = 3x^2 - 6x = 3x(x - 2)$, zero at $x = 0$ and $x = 2$, both inside the domain $(-1, 3]$.

Build a small table of values around the interesting points:

| x | -1 (excluded) | -0.9 | -0.732 | 0 | 1 | 2 | 2.732 | 3 |
|---|---|---|---|---|---|---|---|---|
| f(x) | -2 (limit only) | -1.159 | 0 | 2 | 0 | -2 | 0 | 2 |

Reading the sign changes: $f$ crosses zero once between $-0.9$ and $0$ (at $x \approx -0.732$), again at $x = 1$, and again at $x \approx 2.732$. Only the first crossing sits inside $[-0.9, 0]$, so there is exactly one root there, not two. Statement (A) is false.

The table also shows $f$ rising from $-1.159$ up to $f(0) = 2$ (a local peak), falling back down to $f(2) = -2$ (a local trough), then climbing again to $f(3) = 2$.

Since $x = -1$ is not part of the domain, the value $-2$ that $f$ would take there is never actually reached at that end. The only point where $f$ genuinely equals $-2$ in this domain is $x = 2$. So statement (B), "minimum at 2 only", holds.

The peak value 2 appears twice in the table though, once at $x = 0$ and again at $x = 3$, since $x = 3$ is included in the domain. So the maximum is not exclusive to $x = 0$, which makes statement (C) false.

The table confirms $f(1) = 0$ directly, so statement (D) is true.

So the correct statements are (B) and (D).

$$\boxed{\text{(B) and (D)}}$$

**Quick Tip:** Find the critical points with the first derivative, classify them with the second derivative, and do not forget to check the domain endpoints.

• • •

**28.** Suppose a random variable $Z$ follows $Normal(\mu = 0, \sigma^2 = 1)$ distribution with probability density function $g(z)$ and cumulative distribution function $G(z)$. Another random variable $Y$ follows $t_1$ distribution with probability density function $h(y)$ and cumulative distribution function $H(y)$. Let $c$ be the positive real number for which $g(c) = h(c)$.

Which of the following statements is/are correct?

(A) $G(0) = H(0)$

(B) $G(c) < H(c)$

(C) $G(-c) < H(-c)$

(D) $g(0) = h(0)$

**Correct Answer:** (A) $G(0) = H(0)$ ; (C) $G(-c) < H(-c)$

**SOLUTION:**

The key fact to hold onto here is that a standard normal curve and a $t_1$ (Cauchy) curve are both symmetric, single-peaked bell shapes centered at $0$, but they do not have the same "spread". The normal curve is tall and thin near the centre, while the Cauchy curve is shorter at the peak but drags out into much heavier tails.

Peak heights: $g(0) = \frac{1}{\sqrt{2\pi}} \approx 0.399$ against $h(0) = \frac{1}{\pi} \approx 0.318$. The normal peak is clearly taller, so $g(0) \neq h(0)$, statement (D) fails right away.

Because both curves are mirror images of themselves about $0$, exactly half of each distribution's probability sits on the left of $0$ and half on the right, no matter how the shapes differ elsewhere. That gives $G(0) = H(0) = 0.5$ automatically, so statement (A) is true.

Now picture sliding along the positive $x$-axis. Near $0$, the normal curve sits above the Cauchy curve since it is taller there. But because the Cauchy tail decays only like $1/y^2$ while the normal tail decays like $e^{-y^2/2}$, far away from $0$ the Cauchy curve ends up above the normal curve. The two curves must cross exactly once for positive $y$: that crossing point is $c$, where $g(c) = h(c)$.

So on the strip from $0$ to $c$, the normal curve is on top, meaning the normal distribution picks up more probability in that strip than the Cauchy does. That extra probability pushes $G(c)$ above $H(c)$, i.e. $G(c) > H(c)$, the opposite of what statement (B) claims, so (B) is false.

Mirror that strip to the negative side using symmetry. If the normal has more mass than the Cauchy in $(0, c)$, then by the mirror image, the normal has less mass than the Cauchy in $(-c, 0)$, since both distributions must still total to 1 on each side. Less mass to the left of $-c$ under the normal than under the Cauchy means $G(-c) < H(-c)$, so statement (C) holds.

The correct statements are (A) and (C).

$$\boxed{\text{(A) and (C)}}$$

**Quick Tip:** Remember both densities are symmetric about 0, so their CDFs must equal 0.5 there regardless of the shape of the tails.

• • •

**29.** Consider that for a supervised learning task, the objective function being minimized is $f_w(x) = wx$, where $x \in \mathbb{R}$ is the input and $w \in \mathbb{R}$ is the parameter. Stochastic Gradient Descent with a learning rate of 0.10 is used for parameter updates.

Suppose that at the end of iteration $i$, the value of $w$ becomes 10.00.

Let $x = 10.00$ be the input for iteration $(i + 1)$.

The value of $w$ at the end of iteration $(i + 1)$ is _____. (Rounded off to two decimal places)

**Correct Answer:** 9.00

**SOLUTION:**

This question is testing the basic SGD update formula, applied to a toy objective that happens to have a very simple gradient.

The general one-parameter SGD step is

$$w \leftarrow w - \eta g$$

where $g$ is the gradient of the objective with respect to $w$, evaluated at the current data point, and $\eta$ is the learning rate.

Here the objective is $f_w(x) = wx$. Treating $x$ as a constant during differentiation with respect to $w$ gives

$$g = \frac{\partial}{\partial w}(wx) = x$$

So the gradient at any step is simply equal to whatever input $x$ was fed in that step, nothing more elaborate.

Now plug in the numbers given for iteration $(i + 1)$: the parameter going into this step is $w = 10.00$ (carried over from the end of iteration $i$), the input is $x = 10.00$, so $g = 10.00$, and $\eta = 0.10$.

The update becomes

$$w_{new} = 10.00 - (0.10)(10.00) = 10.00 - 1.00 = 9.00$$

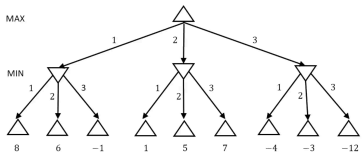
So after this single SGD step, $w$ drops from 10.00 to 9.00.

$$\boxed{w = 9.00}$$

**Quick Tip:** The gradient of wx with respect to w is just x, so the update rule reduces to w minus the learning rate times x.

• • •

**30.** Consider the game tree for a two-player turn-taking minimax game as shown in the figure. The value of a terminal node represents the utility of the game state if the game ends there. The numbers written next to the edges denote the strategies.




There are two players MAX and MIN. At any particular state of the game, MAX prefers to move to a state of maximum value. On the other hand, MIN prefers to move to a state of minimum value.

Suppose MAX starts the game at the root and has three strategies: 1, 2 and 3. Next, MIN plays and also has three strategies: 1, 2 and 3. The game ends there. Both players always take optimal strategies throughout the game.

At the root, the best strategy for MAX is _____. (Answer in integer)

**Correct Answer:** 2

**SOLUTION:**

Minimax works from the leaves upward: figure out what each MIN node will do first, since MIN moves right after MAX, then let MAX pick the best of those outcomes.

List the three MIN nodes with the leaf values under each:

| MAX's move | Leaf values under that MIN node | MIN picks (smallest) |
|---|---|---|
| Strategy 1 | 8, 6, -1 | -1 |
| Strategy 2 | 1, 5, 7 | 1 |
| Strategy 3 | -4, -3, -12 | -12 |

MIN always wants to steer the game toward the lowest possible utility for MAX, so each MIN node's value is just the smallest of its three children. That gives **−1**, **1**, and **−12** for MAX's three options.

Now step up one more level. MAX gets to choose which of these three backed-up values to walk into, and MAX always wants the largest one. Comparing **−1**, **1**, and **−12**, the largest is **1**, coming from strategy 2.

So playing optimally, MAX should choose strategy 2 at the root, since every other strategy leads to a worse (smaller) guaranteed outcome once MIN responds optimally.

Strategy 2

**Quick Tip:** Evaluate each MIN node first by taking the minimum of its children, then let MAX pick the largest of those results.

• • •

**31.** Let $A$ be a sorted array containing 1000 distinct integers. You perform a recursive binary search on $A$ to find an element $y$. Suppose each comparison checks whether the middle element computed during the current recursive step is equal to, less than, or greater than $y$.

The maximum number of comparisons that may have to be performed if $y$ is not an element of $A$ is _____. (Answer in integer)

**Correct Answer:** 10

**SOLUTION:**

A neat way to see this without memorising a formula is to picture binary search as a decision tree. Every internal node of this tree represents one comparison against a middle element, has a "target is smaller" branch and a "target is larger" branch, and every leaf represents "search space is now empty, target is absent".

A tree where each internal node has 2 branches and needs to represent every possible "gap" outcome for a missing element needs enough leaves to cover all $n + 1$ gaps (before the first element, between each pair of elements, and after the last element). With $n = 1000$ elements, there are $1001$ such gaps.

A binary tree with depth $d$ can have at most $2^d$ leaves. We need

$$2^d \geq 1001$$

Check successive powers of 2: $2^9 = 512$, which is too small, and $2^{10} = 1024$, which is enough to cover all 1001 gaps. So the smallest depth that works is $d = 10$.

Each level of this decision tree corresponds to exactly one comparison in the actual search, so the worst-case unsuccessful search needs 10 comparisons before it can conclude the element is not present.

This also lines up with just repeatedly halving 1000: $1000 \to 500 \to 250 \to 125 \to 62 \to 31 \to 15 \to 7 \to 3 \to 1 \to 0$, which is 10 halving steps to reach an empty range.

$$\boxed{10}$$

**Quick Tip:** Each comparison roughly halves the search space, so count how many times 1000 can be halved before nothing is left.

• • •

**32.** In a relational database, a B+ Tree Index is to be constructed for a relation on a key field. In a B+ Tree, a Node Pointer points to a sub-tree and a Data Record Pointer points to a block of database records.

Let, Node size = 4096 bytes, Node Pointer size = 10 bytes, Search Key Field size = 11 bytes and Data Record Pointer size = 12 bytes.

The maximum number of Node Pointers that can be present in a non-leaf node of the B+ Tree is _____. (Answer in integer)

**Correct Answer:** 195

**SOLUTION:**

A B+ Tree non-leaf node is built like this: pointer, key, pointer, key, and so on, ending in a pointer. If there are $n$ pointers, there must be exactly $n-1$ keys sitting between them to guide the search. The Data Record Pointer size given in the question is a distractor here, since that field only matters inside leaf nodes, not non-leaf nodes.

Let $n$ be the number of Node Pointers we are trying to find. The total bytes used by the node is

$$\underbrace{10n}_{\text{pointers}} + \underbrace{11(n-1)}_{\text{keys}}$$

and this has to stay within the 4096-byte node size:

$$10n + 11(n-1) \leq 4096$$

Expand and collect terms:

$$10n + 11n - 11 \leq 4096$$

$$21n \leq 4107$$

$$n \leq \frac{4107}{21} = 195.571\ldots$$

Since $n$ has to be a whole number of pointers, round down to $n = 195$.

Double check: with **195** pointers there are **194** keys, using $195(10) + 194(11) = 1950 + 2134 = 4084$ bytes, safely under the 4096 limit. Bumping up to **196** pointers would need $196(10) + 195(11) = 1960 + 2145 = 4105$ bytes, which overshoots the node size by 9 bytes. So 195 is indeed the maximum.

$$\boxed{n = 195}$$

**Quick Tip:** A non-leaf node with n pointers holds n-1 keys between them; set up the space inequality and solve for the largest integer n.

• • •

**33.** The number of bijections $f(\cdot)$ from the set $S = \{1, 2, 3, 4\}$ to itself such that $f(f(n)) = n$, for all $n \in S$, is _____. (Answer in integer)

**Correct Answer:** 10

**SOLUTION:**

**Step 1:** Set up a recursion for the count of involutions.
Let $I(n)$ be the number of involutions (permutations with $f(f(n)) = n$) on a set of $n$ labeled elements. Look at where the last element, call it element $n$, is sent by $f$.

**Step 2:** Split on the image of element $n$.

Either $f(n) = n$, so element $n$ is fixed and the remaining $n-1$ elements form an involution on their own: this gives $I(n-1)$ possibilities.
Or $f(n) = k$ for some other element $k$. Since $f$ is an involution, $f(k)$ must equal $n$ as well, so $n$ and $k$ form a swapped pair. There are $n-1$ choices for $k$, and the remaining $n-2$ elements form an involution among themselves, giving $(n-1) \cdot I(n-2)$ possibilities.
So the recursion is $I(n) = I(n-1) + (n-1)I(n-2)$, with $I(0) = 1$ and $I(1) = 1$.

**Step 3:** Build up the values for $n = 0$ to 4.
$I(0) = 1$
$I(1) = 1$
$I(2) = I(1) + 1 \cdot I(0) = 1 + 1 = 2$
$I(3) = I(2) + 2 \cdot I(1) = 2 + 2 = 4$
$I(4) = I(3) + 3 \cdot I(2) = 4 + 3 \cdot 2 = 4 + 6 = 10$

**Final Answer:**
The number of bijections on $S = \{1, 2, 3, 4\}$ with $f(f(n)) = n$ for all $n$ is 10.

$$\boxed{10}$$

**Quick Tip:** f(f(n)) = n means f is an involution: count fixed points plus swapped pairs among 4 elements.

• • •

**34.** Let $X$ be an exponentially distributed random variable with mean $\lambda(> 0)$. If $P(X > 5) = 0.35$, then the conditional probability $P(X > 10 \mid X > 5)$ is _____. (Rounded off to two decimal places)

**Correct Answer:** 0.35

**SOLUTION:**

**Step 1:** Write the survival function of the exponential distribution.
If $X$ is exponential with mean $\lambda$, its survival function is $P(X > x) = e^{-x/\lambda}$ for $x \geq 0$. This directly gives the chance $X$ exceeds any value $x$.

**Step 2:** Write the conditional probability as a ratio.
Since the event $X > 10$ is a subset of the event $X > 5$ (if $X$ is bigger than 10 it is automatically bigger than 5),

$$P(X > 10 \mid X > 5) = \frac{P(X > 10 \text{ and } X > 5)}{P(X > 5)} = \frac{P(X > 10)}{P(X > 5)}$$

**Step 3:** Express both survival probabilities using the exponential formula.
$P(X > 5) = e^{-5/\lambda}$ and $P(X > 10) = e^{-10/\lambda} = \left(e^{-5/\lambda}\right)^2$. So the ratio becomes

$$\frac{P(X > 10)}{P(X > 5)} = \frac{\left(e^{-5/\lambda}\right)^2}{e^{-5/\lambda}} = e^{-5/\lambda} = P(X > 5)$$

This shows the ratio always collapses back to $P(X > 5)$ for this exponential setup, which is really just the memoryless property appearing algebraically.

**Step 4:** Plug in the given value.
We are given $P(X > 5) = 0.35$, so

$$P(X > 10 \mid X > 5) = 0.35$$

**Final Answer:**

$$\boxed{0.35}$$

**Quick Tip:** Use the memoryless property of the exponential distribution: P(X>10|X>5) reduces to P(X>5).

• • •

**35.** The value of $\sum_{i=0}^{\infty}\sum_{j=1}^{\infty} 2^{-i}\,3^{-j}$ is _____. (Answer in integer)

**Correct Answer:** 1

**SOLUTION:**

**Step 1:** Fix $i$ and sum over $j$ first.
For a fixed value of $i$, the inner sum is $\sum_{j=1}^{\infty} 2^{-i}3^{-j} = 2^{-i}\sum_{j=1}^{\infty} 3^{-j}$, since $2^{-i}$ does not depend on $j$ and can be pulled out.

**Step 2:** Evaluate the inner geometric series.
$\sum_{j=1}^{\infty} 3^{-j} = \frac{1}{3} + \frac{1}{9} + \frac{1}{27} + \cdots$ is geometric with first term $1/3$ and ratio $1/3$, so it sums to $\frac{1/3}{1-1/3} = \frac{1}{2}$. So the inner sum for fixed $i$ becomes $2^{-i} \cdot \frac{1}{2}$.

**Step 3:** Sum the result over all $i$ from 0 to infinity.

$$\sum_{i=0}^{\infty} 2^{-i} \cdot \frac{1}{2} = \frac{1}{2}\sum_{i=0}^{\infty} 2^{-i}$$

The remaining series $\sum_{i=0}^{\infty} 2^{-i} = 1 + \frac{1}{2} + \frac{1}{4} + \cdots$ is geometric with first term 1 and ratio $1/2$, giving $\frac{1}{1-1/2} = 2$.

**Step 4:** Combine.

$$\frac{1}{2} \times 2 = 1$$

**Final Answer:**

$$\boxed{1}$$

**Quick Tip:** Split the double sum into a product of two independent geometric series, one in i and one in j.

• • •

**36.** Let four points in three-dimensional space be:
$P1 : [2, 3, -1]$, $P2 : [3, 1, 1]$, $P3 : [5, -2, 3]$ and $P4 : [3, 3, 3]$.
Hierarchical Agglomerative Clustering is used to cluster the above points. If Manhattan Distance is used as the distance metric during clustering, which of the following options indicates the two points that will be merged first?

(A) $P1, P2$

(B) $P2, P3$

(C) $P3, P4$

(D) $P2, P4$

**Correct Answer:** (D) $P2, P4$

**SOLUTION:**

**Step 1:** List the coordinates clearly.
$P1 = (2, 3, -1)$, $P2 = (3, 1, 1)$, $P3 = (5, -2, 3)$, $P4 = (3, 3, 3)$. Manhattan distance between two points just adds up the absolute differences along each of the 3 coordinate axes, so we build a small distance table for all pairs.

**Step 2:** Work out the coordinate-wise differences for each pair.
$P1$ to $P2$: differences are $1, 2, 2 \Rightarrow$ sum $= 5$
$P1$ to $P3$: differences are $3, 5, 4 \Rightarrow$ sum $= 12$
$P1$ to $P4$: differences are $1, 0, 4 \Rightarrow$ sum $= 5$
$P2$ to $P3$: differences are $2, 3, 2 \Rightarrow$ sum $= 7$
$P2$ to $P4$: differences are $0, 2, 2 \Rightarrow$ sum $= 4$
$P3$ to $P4$: differences are $2, 5, 0 \Rightarrow$ sum $= 7$

**Step 3:** Scan the table for the smallest entry.
Sorting the six distances: $4 < 5 = 5 < 7 = 7 < 12$. The smallest value, 4, belongs to the pair $(P2, P4)$, since their $x$-coordinates already match (difference 0) and their $y$ and $z$ coordinates are each only 2 apart.

**Step 4:** Apply the clustering rule.
In agglomerative clustering the very first merge always joins the closest pair of individual points. Since $(P2, P4)$ has the smallest Manhattan distance of all six pairs, this is the pair that merges first.

**Final Answer:**
The pair $(P2, P4)$ merges first, which is option (D).

**Quick Tip:** Compute all 6 pairwise Manhattan distances between the points and pick the smallest one.

• • •

**37.** Which of the following statements is true for Ridge Regression?

(A) The regularizer in the objective function of Ridge Regression is used to guard against scenarios where the model works well for the test data, but poorly for the training data.

(B) The regularizer of Ridge Regression uses $L_1$ norm.

(C) Ridge Regression aims to reduce the number of parameters that have negative values.

(D) The regularizer of Ridge Regression may increase the bias of the model, but it helps in reducing the variance in predictions.

**Correct Answer:** (D) The regularizer of Ridge Regression may increase the bias of the model, but it helps in reducing the variance in predictions.

**SOLUTION:**

Ridge Regression modifies ordinary least squares by adding a penalty on the size of the weights: $\text{Cost} = \text{RSS} + \lambda \sum w_i^2$. We check each statement against this definition.

1. **The regularizer guards against the model working well on test data but poorly on training data**: This gets overfitting backwards. Overfitting means good performance on training data and poor performance on test data, not the reverse, so this statement is false.
2. **The regularizer uses $L_1$ norm**: Ridge's penalty term is $\sum w_i^2$, the squared $L_2$ norm. The $L_1$ norm penalty ($\sum |w_i|$) is what Lasso Regression uses instead, so this is false for Ridge.
3. **Ridge Regression reduces the number of parameters with negative values**: Ridge shrinks all weights toward zero by roughly the same proportion, but it does not target the sign of the weights or force any of them to zero, so this statement misrepresents what the penalty does.
4. **The regularizer may increase bias but reduces variance**: Since the penalty shrinks weights away from their unconstrained least-squares fit, it introduces a small, controlled bias. In exchange, the model becomes far less sensitive to noise in the training set, so its variance across different samples drops. This trade-off is precisely the mechanism Ridge Regression uses to reduce overfitting, which makes this statement true.

The correct option is the fourth one: Ridge Regression's $L_2$ penalty increases bias slightly while reducing variance, which is the standard bias-variance trade-off behind regularization.

Let's summarize:

- Ridge Regression uses an $L_2$ (squared-weight) penalty, not $L_1$.
- It shrinks weights but does not zero them out or specifically target negative values.
- Its main effect is trading a bit of bias for a large drop in variance, which reduces overfitting.

So the statement in option (D) is the one that correctly describes Ridge Regression.

**Quick Tip:** Ridge Regression's penalty is the squared L2 norm; think about what shrinking weights does to bias and variance.

• • •

**38.** Assume that a Creative ($C$) person will Succeed ($S$) if the person is also Disciplined ($D$), but will not succeed otherwise. Now, consider the following statements:
(i) $C \wedge S \Leftrightarrow D$
(ii) $C \Rightarrow (S \Leftrightarrow D)$
(iii) $C \Leftrightarrow ((D \Rightarrow S) \vee \neg S)$
Which of the following options is correct?

(A) Both (i) and (ii) are TRUE

(B) Only (ii) is TRUE

(C) Both (ii) and (iii) are TRUE

(D) Only (iii) is TRUE

**Correct Answer:** (B) Only (ii) is TRUE

**SOLUTION:**

**Step 1:** Write the given rule as a formula.
"A Creative person succeeds if and only if disciplined" (with succeeding being impossible for creative-but-undisciplined people) translates to the axiom $A : C \Rightarrow (S \Leftrightarrow D)$. Any world we consider must make $A$ true, since it is given as a fact about how these three properties relate.

**Step 2:** Go through all 8 combinations of $C, S, D$ and mark which satisfy $A$.
$C$=0: $A$ is automatically true no matter what $S, D$ are, since an implication with a false antecedent is always true. That covers all 4 rows with $C$=0.
$C$=1: $A$ needs $S \Leftrightarrow D$, so only $(S$=0$, D$=0$)$ and $(S$=1$, D$=1$)$ satisfy it; the rows $(S$=0$, D$=1$)$ and $(S$=1$, D$=0$)$ with $C$ =1 violate $A$ and are excluded.
So the 6 surviving rows are: $(0,0,0), (0,0,1), (0,1,0), (0,1,1), (1,0,0), (1,1,1)$, written as $(C, S, D)$.

**Step 3:** Check statement (ii) is just $A$ itself.
Statement (ii) is $C \Rightarrow (S \Leftrightarrow D)$, which is identical to $A$. Since we only ever consider worlds where $A$ holds, statement (ii) is true in every one of the 6 surviving rows, so it is always TRUE.

**Step 4:** Check statement (i) on the row $(0,0,1)$.
Here $C \wedge S = 0 \wedge 0 = 0$, while $D = 1$. So $C \wedge S \Leftrightarrow D$ becomes $0 \Leftrightarrow 1$, which is false. Statement (i) fails on this valid row, so it is not always true.

**Step 5:** Check statement (iii) on the row $(0,0,0)$.
Here $D \Rightarrow S = 0 \Rightarrow 0 = 1$ and $\neg S = 1$, so $(D \Rightarrow S) \vee \neg S = 1$. But $C = 0$, so $C \Leftrightarrow 1$ is false. Statement (iii) fails on

this valid row too, so it is not always true.

**Final Answer:**
Only statement (ii) is guaranteed true across every world consistent with the rule, so the answer is option (B).

**Quick Tip:** Rewrite the given rule as C => (S <=> D), then test each statement against the valid (C,S,D) combinations.

• • •

**39.** A recursive function in Python is given.

```
def mystery(n):
    if n <= 0:
        return 1
    else:
        return mystery(n-1) + mystery(n-2)
```

Now, consider the following function call:
`mystery(4)`
Assume that a typical runtime stack is used to manage function calls. Each function call is pushed onto the stack and removed only after it finishes execution.
Which of the following options denotes the total number of function calls (i.e., the total number of stack activations), including the initial call, to compute `mystery(4)`?

(A) 5

(B) 9

(C) 15

(D) 17

**Correct Answer:** (C) 15

**SOLUTION:**

**Step 1:** Expand the call tree of $mystery(4)$ level by level.
$mystery(4)$ is 1 call. It spawns $mystery(3)$ and $mystery(2)$.
$mystery(3)$ spawns $mystery(2)$ and $mystery(1)$.
$mystery(2)$ (the one under $mystery(4)$ directly) spawns $mystery(1)$ and $mystery(0)$.

**Step 2:** Keep expanding every non-base-case call until only base cases remain.
Every call with argument $\leq 0$ stops immediately, costing 1 activation with no children. Every call with argument 1 spawns $mystery(0)$ and $mystery(-1)$, both base cases, so a call to $mystery(1)$ always costs $1 + 1 + 1 = 3$ activations total. A call to $mystery(2)$ spawns $mystery(1)$ (cost 3) and $mystery(0)$ (cost 1), so it costs $1 + 3 + 1 = 5$ activations total.

**Step 3:** Roll this up to $mystery(3)$ and then $mystery(4)$.

*mystery*(3) spawns *mystery*(2) (cost 5) and *mystery*(1) (cost 3), so it costs $1 + 5 + 3 = 9$ activations total.
*mystery*(4) spawns *mystery*(3) (cost 9) and *mystery*(2) (cost 5), so it costs $1 + 9 + 5 = 15$ activations total.

**Step 4:** Sanity check against the stack description.
Since each call stays on the stack only while it is active and is popped once it returns, the "total number of function calls" being asked for is the count of every push that ever happens, not the maximum stack depth at any one time. Adding up every push in the tree built above gives 15.

**Final Answer:**
There are 15 total function-call activations to compute *mystery*(4), matching option (C).

**Quick Tip:** Set up T(n) = 1 + T(n-1) + T(n-2) with T(n)=1 for n<=0, then build up to T(4).

• • •

**40.** Consider a directed graph $G = (V, E)$, where $V$ is the finite set of vertices and $E$ is the set of directed edges between the vertices. $G$ may contain cycles but there is no self-loop. Further, $G$ may not be strongly connected. Let $G^R$ be the graph obtained by reversing the directions of all the edges in $G$ without changing the set of vertices. Assume that Breadth First Search (BFS) or Depth First Search (DFS) from any given vertex $v$ of a graph visits only the reachable vertices from $v$ in that graph.
Which of the following statements must always be true, regardless of the structure of $G$?

(A) If $u$ is a reachable vertex in the BFS of $G^R$ from $v$, then $u$ is also a reachable vertex in the DFS of $G$ from $v$.

(B) In $G^R$, the BFS traversal from $v$ will visit exactly the same set of vertices as the DFS from $v$ in $G$.

(C) The order of vertices visited in the BFS of $G^R$ from $v$ is the reverse of the order of vertices visited in the DFS of $G$ from $v$.

(D) If $u$ is a reachable vertex in the DFS of $G$ from $v$, then $v$ is also a reachable vertex in the BFS of $G^R$ from $u$.

**Correct Answer:** (D) If $u$ is a reachable vertex in the DFS of $G$ from $v$, then $v$ is also a reachable vertex in the BFS of $G^R$ from $u$.

**SOLUTION:**

**Step 1:** State the key lemma about reversed graphs.
For any directed graph $G$ and its edge-reversal $G^R$, and any two vertices $x, y$: $y$ is reachable from $x$ in $G$ exactly when $x$ is reachable from $y$ in $G^R$. This holds because every directed edge $p \to q$ in $G$ becomes $q \to p$ in $G^R$, so any path $x \to \cdots \to y$ in $G$ becomes the path $y \to \cdots \to x$ in $G^R$ when every edge on it is flipped, and the vertices used are identical, just traversed in the opposite direction.

**Step 2:** Apply the lemma with $x = v$ and $y = u$.
If $u$ is reachable from $v$ in $G$, the lemma with $x = v, y = u$ says $v$ is reachable from $u$ in $G^R$. Since the question tells us that BFS (or DFS) from any vertex visits exactly its reachable set, "$v$ reachable from $u$ in $G^R$" is the same fact as "$v$ is visited by BFS of $G^R$ starting at $u$".

**Step 3:** Match this to each option.
Option (D) states precisely this: if $u$ is reachable in DFS of $G$ from $v$, that is, $u$ reachable from $v$ in $G$, then $v$ is

reachable in BFS of $G^R$ from $u$, that is, $v$ reachable from $u$ in $G^R$. This is the lemma itself, so it holds for every possible $G$, cyclic or not, strongly connected or not.

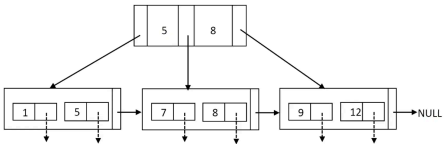
**Step 4:** Show the other three do not follow from the lemma.
Options (A), (B), and (C) all compare the reachable set from the same vertex $v$ in $G$ against the reachable set from that same $v$ in $G^R$. The lemma says nothing about these two sets being related, equal, or reversed in visiting order. They describe reachability in opposite directions of the edges and can be completely disjoint except for $v$ itself, so none of (A), (B), (C) is guaranteed by the lemma or by anything else in the problem statement.

**Final Answer:**
Only option (D) is guaranteed true for every directed graph $G$.

**Quick Tip:** Reversing every edge on a path from v to u in G turns it into a path from u to v in G^R.

• • •

**41.** Consider a B+ Tree where the maximum number of key values in each leaf node is 2 and the maximum number of pointers in each non-leaf node is 3. Let the content of the B+ Tree be as shown in the figure.




Which of the following options denotes the key value(s) stored in the root node after inserting a key value 3 in the given B+ Tree?

(A) 5

(B) 8

(C) 3 and 5

(D) 3, 5 and 8

**Correct Answer:** (A) 5

**SOLUTION:**

**Step 1:** Restate the split rules before touching the tree.
A leaf here can hold at most 2 keys, so a 3rd key forces a leaf split. An internal node can hold at most 2 keys (since it can carry at most 3 child pointers), so a 3rd key at that level forces the node to split too, with its middle key promoted upward instead of being copied.

**Step 2:** Locate the target leaf for key 3.
The root separators are 5 and 8. Using the ordering already visible from the leaves, the leftmost child [1, 5] is

the one whose largest stored key is 5, so any key at or below 5, including 3, lands there.

**Step 3:** Grow the leaf and split it.
[1, 5] plus the new key 3 gives the ordered list [1, 3, 5]. Splitting an odd list of 3 into two leaves of size 2 and 1 gives [1, 3] on the left and [5] on the right.
The separator carried up to the parent for this new pair is the boundary value 3, the last key of the left half.

**Step 4:** Count pointers at the root before deciding if it overflows.
The root started with exactly 3 children (its allowed maximum). Inserting the new separator 3 would raise the child count to 4 pointers with keys [3, 5, 8], one pointer past the limit, so the root must be split next.

**Step 5:** Promote the middle key of the overflowed root.
List the three root keys in order: 3, 5, 8. The middle entry is 5. In an internal-node split this middle entry alone becomes the new root, while 3 stays with the left half and 8 stays with the right half; neither child keeps a copy of 5.

**Conclusion:**
The tree now has a new single-key root: 5. That is choice (A).

$\boxed{5}$

**Quick Tip:** First find which leaf 3 falls into using the $\leq$ rule on the root separators, then check the leaf split, and finally check if the root itself now has too many pointers and must split too.

**42.** Consider the given relations $X$, $Y$ and $Z$. The relation $X$ has three columns P, Q and R. The relation $Y$ has three columns P, Q and S. The relation $Z$ has two columns P and T.

Relation X:

| P | Q | R |
|---|---|---|
| P1 | Q1 | R1 |
| P2 | Q2 | R2 |
| P3 | Q3 | R2 |

Relation Y:

| P | Q | S |
|---|---|---|
| P1 | Q1 | 2 |
| P1 | Q2 | 5 |
| P2 | Q1 | 6 |
| P3 | Q3 | 1 |

Relation Z:

| P | T |
|---|---|
| P1 | T1 |
| P3 | T2 |
| P4 | T3 |
| P4 | NULL |

Consider the relational algebra expression

$$\Pi_{P,R,S}\left[\left(\sigma_{(Q=Q3 \vee R=R2)}[X \bowtie Y]\right) \bowtie \left(\sigma_{(S>1)}[Y \bowtie Z]\right)\right]$$

where $\bowtie$ denotes natural join operation.

Which of the following options is the correct output for the given expression?

(A) Two rows (P1, R1, 2) and (P1, R1, 5)

(B) Three rows (P1, R1, 2), (P1, R1, 5) and (P2, R2, 6)

(C) One row (P1, R1, 2)

(D) Zero rows

**Correct Answer:** (D) Zero rows

**SOLUTION:**

**Step 1:** Work out each branch's key value first, before computing every column.
The final operation is a natural join of two intermediate relations on their common columns P, Q, S. A natural join can only produce rows where the P values actually agree, so it helps to track just the P values on each side before doing the full column-by-column join.

**Step 2:** Track the P value on the left branch.
$X \bowtie Y$ only keeps rows where P and Q match between X and Y. Checking each X row against Y gives two survivors: (P1,Q1,R1,2) and (P3,Q3,R2,1).
Applying $\sigma_{(Q=Q3 \vee R=R2)}$: the P1 row has Q=Q1 and R=R1, neither condition holds, so it is removed. The P3 row has Q=Q3, so it survives.
So the left branch collapses to a single row with P=P3.

**Step 3:** Track the P value on the right branch.
$Y \bowtie Z$ matches on P alone. P1 appears twice in Y and once in Z, giving two joined rows with P=P1: (P1,Q1,2,T1) and (P1,Q2,5,T1). P2 has no partner in Z. P3 gives (P3,Q3,1,T2).
Applying $\sigma_{(S>1)}$: the two P1 rows have S=2 and S=5, both greater than 1, so both survive. The P3 row has S=1, which fails $S > 1$, so it is removed.
So the right branch collapses to two rows, and both of them have P=P1 (the P3 row was filtered out by the S>1 condition).

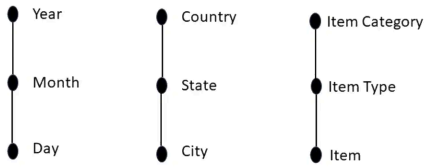
**Step 4:** Compare the P values across the two branches.
The left branch only has P=P3. The right branch only has P=P1. Since a natural join requires equal P values to produce any tuple, and P3 never equals P1, there is no way for any pair of rows to combine.

**Step 5:** Conclude without needing to check Q or S at all.
Because the P values already disagree completely between the two sides, the join is empty regardless of what the Q and S columns contain, and projecting an empty relation onto P, R, S still gives zero rows. This matches option (D).

Zero rows

**Quick Tip:** Compute $X \bowtie Y$ and $Y \bowtie Z$ separately, apply each selection, then check whether the surviving P values on the two sides can ever match before joining them.

**43.** Consider the concept hierarchies as shown in the figure.




Which of the following options denotes the total number of possible data cuboids from these concept hierarchies?

(A) $4^3$

(B) $2^3$

(C) $2!$

(D) $4!$

**Correct Answer:** (A) $4^3$

**SOLUTION:**

**Step 1:** List the actual set of levels for one dimension.
Take the time dimension as an example. Its hierarchy is Year, then Month, then Day. The set of ways to group data along this one dimension is {all, Year, Year-Month, Year-Month-Day}, which has 4 members: the fully generalized "all" plus the 3 named levels.

**Step 2:** Repeat for the other two dimensions.
The location dimension (Country, State, City) gives the same count of 4 possible grouping levels: {all, Country, Country-State, Country-State-City}. The product dimension (Item Category, Item Type, Item) also gives 4: {all, Item Category, Item Category-Item Type, Item Category-Item Type-Item}.

**Step 3:** Build the cube as a Cartesian product of level choices.
A data cuboid corresponds to picking exactly one entry from each dimension's set of levels, independently of the other dimensions, since the dimensions are orthogonal axes of the cube.
The number of ways to make 3 independent choices from sets of size 4, 4 and 4 is $4 \times 4 \times 4$.

**Step 4:** Compute the count and check the distractors.

$$4 \times 4 \times 4 = 64 = 4^3$$

$2^3$ undercounts because it wrongly assumes only 2 grouping levels per dimension. $2!$ and $4!$ count orderings of items, which is a different combinatorial question and does not apply to independently choosing a level from each dimension.

**Conclusion:**
The number of data cuboids is $4^3$, so option (A) is correct.

$4^3$

**Quick Tip:** Each dimension has 3 hierarchy levels plus an implicit "all" level, giving 4 choices per dimension; multiply these across the 3 independent dimensions.

• • •

**44.** Let $X$ and $Y$ be two independent random variables. $X$ follows $Bernoulli(p = 0.3)$ distribution and $Y$ follows $Normal(\mu = 0, \sigma^2 = 100)$ distribution.

Which of the following options is the variance of $(2X - 1)Y$?

(A) 100

(B) 90

(C) 49

(D) 21

**Correct Answer:** (A) 100

**SOLUTION:**

**Step 1:** Use the general variance formula for a product of two independent variables.
For independent $U$ and $V$: $\text{Var}(UV) = \text{Var}(U)\text{Var}(V) + \text{Var}(U)(E[V])^2 + \text{Var}(V)(E[U])^2$.
Here take $U = 2X - 1$ and $V = Y$.

**Step 2:** Find the moments of $U = 2X - 1$.
$X$ is Bernoulli(0.3), so $\text{Var}(X) = p(1 - p) = 0.3 \times 0.7 = 0.21$ and $E[X] = 0.3$.
For a linear transform, $\text{Var}(2X - 1) = 4\,\text{Var}(X) = 4(0.21) = 0.84$, and $E[2X - 1] = 2(0.3) - 1 = -0.4$.

**Step 3:** Find the moments of $V = Y$.
$Y$ is $Normal(0, 100)$, so $\text{Var}(Y) = 100$ and $E[Y] = 0$.

**Step 4:** Substitute into the product-variance formula.

$$\text{Var}(UV) = \text{Var}(U)\text{Var}(V) + \text{Var}(U)(E[V])^2 + \text{Var}(V)(E[U])^2$$

$$= (0.84)(100) + (0.84)(0)^2 + (100)(-0.4)^2$$

$$= 84 + 0 + 100(0.16)$$

$$= 84 + 16 = 100$$

**Step 5:** Cross-check against the direct computation.
This matches the direct route of noting $(2X - 1)^2 = 1$ always, so $E[(2X - 1)^2 Y^2] = E[Y^2] = 100$ and, since the mean of the product is 0, the variance equals this second moment, 100. Both routes agree.

**Conclusion:**
The variance of $(2X-1)Y$ is 100, so option (A) is correct. Getting 90, 49 or 21 instead means one of the pieces, such as $\text{Var}(X)$, $E[X]$, or $\text{Var}(Y)$, was plugged into the product-variance formula incorrectly.

$$\boxed{100}$$

**Quick Tip:** Notice that $2X-1$ only takes values $-1$ or $1$, so its square is always 1; use this with independence to find $E[W]$ and $E[W^2]$ for $W=(2X-1)Y$.

• • •

**45.** Let

$$L=\lim_{n\to\infty}\sum_{k=0}^{n}\frac{e^{-n}n^k}{k!}$$

Which of the following is the value of $L$?

(A) 0.5

(B) 1.0

(C) 0

(D) $e^{-1}$

**Correct Answer:** (A) 0.5

**SOLUTION:**

**Step 1:** Reframe the sum using the skewness of the Poisson distribution.
$L$ is $P(N\le n)$ for $N\sim Poisson(n)$, the probability that the count is at most equal to its own mean $n$. The skewness of a Poisson($n$) distribution is $1/\sqrt{n}$, which shrinks to 0 as $n$ grows, so the distribution becomes more and more symmetric about its mean for large $n$.

**Step 2:** Use symmetry directly instead of invoking the CLT limit formula.
A distribution that is (in the limit) perfectly symmetric about its mean $n$ splits its total probability mass of 1 into two equal halves on either side of the mean: $P(N<n)\to P(N>n)$ as $n\to\infty$.
Since $P(N<n)+P(N=n)+P(N>n)=1$ and the individual term $P(N=n)=\frac{e^{-n}n^n}{n!}$ shrinks to 0 as $n\to\infty$ (by Stirling's approximation, it behaves like $\frac{1}{\sqrt{2\pi n}}\to 0$), the two tails must each approach $1/2$.

**Step 3:** Match this to $L$.
$L=P(N\le n)=P(N<n)+P(N=n)$. Since $P(N<n)\to 1/2$ and $P(N=n)\to 0$,

$$L\to\frac{1}{2}+0=0.5$$

**Step 4:** Sanity-check with the boundary options.
$L = 1.0$ would require the sum to go on forever past $k = n$, capturing the whole distribution, which is not what the given sum does. $L = 0$ contradicts the fact that roughly half the mass sits at or below the mean by symmetry. $e^{-1}$ is just the isolated $k = 0$ term, not the cumulative sum.

**Conclusion:**
This confirms $L = 0.5$, option (A), matching a well known asymptotic result for the Poisson CDF evaluated at its own mean.

$$\boxed{L = 0.5}$$

**Quick Tip:** The sum is the Poisson(n) CDF evaluated at its own mean n; as n grows, the Poisson distribution becomes approximately Normal and symmetric about its mean, so roughly half the probability lies at or below it.

• • •

**46.** Let $\gamma_1, \gamma_2, \gamma_3$ be the eigenvalues of the matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos t & \sin t \\ 0 & -\sin t & \cos t \end{bmatrix}$$

where $t \in [-\pi, \pi]$ is in radians.

Which one of the following options lists all the possible values of $t$ satisfying $\gamma_1 + \gamma_2 + \gamma_3 = 1 + \sqrt{2}$?

(A) $\{\pi/3, -\pi/4\}$

(B) $\{\pi/4, -\pi/3\}$

(C) $\{\pi/4, -\pi/4\}$

(D) $\{\pi/3, -\pi/3\}$

**Correct Answer:** (C) $\{\pi/4, -\pi/4\}$

**SOLUTION:**

**Step 1:** Split the matrix into its block structure.
The matrix is block diagonal: a $1 \times 1$ block $[1]$ in the top left, and a $2 \times 2$ rotation-style block $\begin{bmatrix} \cos t & \sin t \\ -\sin t & \cos t \end{bmatrix}$ in the bottom right. Eigenvalues of a block-diagonal matrix are just the eigenvalues of each block, combined.

**Step 2:** Find the eigenvalue of the $1 \times 1$ block.
The top-left block is simply the number 1, so it contributes the eigenvalue $\gamma_1 = 1$ directly.

**Step 3:** Find the eigenvalues of the $2 \times 2$ block using its characteristic equation.
For $\begin{bmatrix} \cos t & \sin t \\ -\sin t & \cos t \end{bmatrix}$, the characteristic polynomial is

$$\lambda^2 - (2\cos t)\lambda + (\cos^2 t + \sin^2 t) = 0$$

$$\lambda^2 - (2\cos t)\lambda + 1 = 0$$

since $\cos^2 t + \sin^2 t = 1$. Solving this quadratic with the quadratic formula gives $\lambda = \cos t \pm i \sin t$, which by Euler's formula are $e^{it}$ and $e^{-it}$.

**Step 4:** Add up all three eigenvalues.

$$\gamma_1 + \gamma_2 + \gamma_3 = 1 + e^{it} + e^{-it} = 1 + 2\cos t$$

using the identity $e^{it} + e^{-it} = 2\cos t$. This confirms the same trace-based result reached from a different route.

**Step 5:** Solve for $t$.

$$1 + 2\cos t = 1 + \sqrt{2} \implies \cos t = \frac{1}{\sqrt{2}}$$

Within $[-\pi, \pi]$, the solutions are $t = \pi/4$ and $t = -\pi/4$, since these are exactly the angles whose cosine is $1/\sqrt{2}$.

**Conclusion:**
This matches option (C), $\{\pi/4, -\pi/4\}$; the pair involving $\pi/3$ would need $\cos t = 1/2$, which is a different value entirely.

$$\boxed{t \in \{\pi/4, -\pi/4\}}$$

**Quick Tip:** The sum of eigenvalues equals the matrix trace; set $1 + 2\cos t = 1 + \sqrt{2}$ and solve for $\cos t$, then find all matching angles in $[-\pi, \pi]$.

• • •

**47.** Consider that 20 stories of Author X and 10 stories of Author Y were kept together without mentioning the names of the authors. A classifier was then asked to predict the author (X or Y) of each of these stories. Let, out of X's stories, 6 were classified as that of Y. On the other hand, out of Y's stories, 2 were classified as that of X.

Considering X and Y as two classes, which of the following statements is/are true?

(A) Accuracy of the classifier is 11/15.

(B) Precision of Class X is higher than the Precision of Class Y.

(C) Recall of Class X is higher than the Recall of Class Y.

(D) Accuracy of the classifier is 14/15.

**Correct Answer:** (A) Accuracy of the classifier is 11/15. ; (B) Precision of Class X is higher than the Precision of Class Y.

**SOLUTION:**

The classifier makes two kinds of errors: 6 of X's 20 stories are wrongly tagged Y, and 2 of Y's 10 stories are wrongly tagged X. That means, out of 30 stories in total, 14 X-stories and 8 Y-stories are labeled correctly, and the remaining 8 stories (6 + 2) are labeled wrong. Let's check each statement in turn.

1. **Accuracy is 11/15**: Accuracy counts every correct call over every story: $(14 + 8)/30 = 22/30$. Dividing top and bottom by 2 gives $11/15$. This statement is true.
2. **Precision of X is higher than Precision of Y**: Precision asks, out of everything the classifier called a given class, how much of it was actually that class. The classifier called 16 stories "X" in total (14 real X plus 2 real Y mislabeled), and 14 of those really were X, giving $14/16 = 0.875$. It called 14 stories "Y" in total (8 real Y plus 6 real X mislabeled), and 8 of those really were Y, giving $8/14 \approx 0.571$. Since $0.875 > 0.571$, this statement is true.
3. **Recall of X is higher than Recall of Y**: Recall asks, out of every story that truly belongs to a class, how much did the classifier catch. For X, 14 out of the 20 real X stories were caught: $14/20 = 0.7$. For Y, 8 out of the 10 real Y stories were caught: $8/10 = 0.8$. Since $0.7 < 0.8$, X's recall is lower, not higher, so this statement is false.
4. **Accuracy is 14/15**: We already found the true accuracy is $11/15$, which is about $0.733$, not $14/15 \approx 0.933$. This statement is false.

So the true statements are the first two: the accuracy is 11/15, and X's precision beats Y's precision. The apparent asymmetry, X has better precision but worse recall than Y, happens because Y has fewer stories overall (10 vs 20), so the 2 misclassified Y stories hurt Y's precision more than they hurt X's recall.

Let's summarize:

- Accuracy = 22/30 = 11/15, matching statement (A).
- $\text{Precision}_X$ = 14/16 = 0.875 is greater than $\text{Precision}_Y$ = 8/14 ≈ 0.571, matching statement (B).
- $\text{Recall}_X$ = 0.7 is less than $\text{Recall}_Y$ = 0.8, so statement (C) fails.

The correct choices are (A) and (B).

**Quick Tip:** Build the 2x2 confusion matrix from the story counts first (14, 6, 2, 8), then compute accuracy, precision per class, and recall per class from it.

• • •

**48.** Let $P(x)$ be a predicate.

Which of the following statements is/are NOT valid in first-order logic?

(A) $\forall x\, P(x) \Rightarrow \exists x\, P(x)$

(B) $\exists x\, P(x) \Rightarrow \forall x\, P(x)$

(C) $\exists x\, P(x) \Leftrightarrow \forall x\, P(x)$

(D) $\forall x\, P(x) \Rightarrow \exists x\, \neg P(x)$

**Correct Answer:** (B) $\exists x\, P(x) \Rightarrow \forall x\, P(x)$ ; (C) $\exists x\, P(x) \Leftrightarrow \forall x\, P(x)$ ; (D) $\forall x\, P(x) \Rightarrow \exists x\, \neg P(x)$

**SOLUTION:**

To test validity of these predicate-logic statements, use a small two-element domain $\{a, b\}$ and try every combination of truth values for $P(a)$ and $P(b)$. If any single combination makes a statement false, the statement is not valid; if every combination makes it true, it is valid.

1. $\forall x\, P(x) \Rightarrow \exists x\, P(x)$: check all four combinations of $P(a), P(b) \in \{T, F\}$. The antecedent $\forall x\, P(x)$ is only true in the single case $P(a) = T, P(b) = T$, and in that exact case the consequent $\exists x\, P(x)$ is also true. In every other case the antecedent is false, which makes the implication true automatically. So it is true in all four cases: this statement is valid.
2. $\exists x\, P(x) \Rightarrow \forall x\, P(x)$: take $P(a) = T, P(b) = F$. The antecedent $\exists x\, P(x)$ is true (because of $a$), but the consequent $\forall x\, P(x)$ is false (because of $b$). True implies false is false, so this combination breaks the statement: not valid.
3. $\exists x\, P(x) \Leftrightarrow \forall x\, P(x)$: with the same $P(a) = T, P(b) = F$, the left side is true and the right side is false, so the biconditional is false: not valid.
4. $\forall x\, P(x) \Rightarrow \exists x\, \neg P(x)$: take $P(a) = T, P(b) = T$. The antecedent $\forall x\, P(x)$ is true, but the consequent $\exists x\, \neg P(x)$ asks for some element where $P$ fails, and there is none, so it is false. True implies false is false: not valid.

Running through all four truth-value combinations on a small domain shows the same answer as picking targeted counterexamples: option (A) survives every combination, while (B), (C) and (D) each fail for at least one assignment of $P(a)$ and $P(b)$.

Let's summarize:

- (A) holds for every truth assignment, so it is valid.
- (B) fails when $P(a) = T, P(b) = F$.
- (C) fails under the same assignment as (B).
- (D) fails when $P(a) = T, P(b) = T$.

The statements that are NOT valid are (B), (C) and (D).

**Quick Tip:** Try a small two-element domain with mixed truth values for P; a formula is valid only if it stays true under every such assignment.

• • •

**49.** Consider the problem of sorting the given array in ascending order:

$P = [1, 2, 3, 5, 4]$

Consider two sorting algorithms Bubble Sort (BS) and Insertion Sort (IS).

Let $N_1$ be the total number of comparisons done by BS on the elements of $P$ and $N_2$ be the total number of comparisons done by IS on the elements of $P$.

Which of the following options is/are correct?

(A) $N_1 = 10, N_2 = 4$

(B) $N_1 > N_2$

(C) IS on P will perform only one swap

(D) Both BS and IS on P will make at least one unnecessary comparison (i.e., comparing elements that are already in correct order)

**Correct Answer:** (B) $N_1 > N_2$ ; (C) IS on P will perform only one swap ; (D) Both BS and IS on P will make at least one unnecessary comparison (i.e., comparing elements that are already in correct order)

**SOLUTION:**

**Step 1:** Set up the problem.
We have $P = [1, 2, 3, 5, 4]$, an array of 5 elements that is sorted except for the last two entries, which are swapped.

**Step 2:** Bubble sort comparison count.
A textbook bubble sort with no early-exit check always compares every adjacent pair in every pass, no matter how sorted the array already is. For $n = 5$ elements it runs 4 passes with $4, 3, 2, 1$ comparisons respectively. $N_1 = 4 + 3 + 2 + 1 = 10$, which is just the sum of the first $(n - 1)$ natural numbers, $\binom{n}{2} = \binom{5}{2} = 10$.

**Step 3:** Insertion sort comparison count.
Insertion sort only compares an element with its immediate left neighbor until it finds the correct slot, so its comparison count depends on how out of place the elements are.
Position 2 (value 2): 1 comparison against 1, no shift.
Position 3 (value 3): 1 comparison against 2, no shift.
Position 4 (value 5): 1 comparison against 3, no shift.
Position 5 (value 4): compares against 5 first (comparison 1, shift happens), then against 3 (comparison 2, stop).
$N_2 = 1 + 1 + 1 + 2 = 5$

**Step 4:** Evaluate each option against these counts.
(A) says $N_1 = 10, N_2 = 4$: the $N_1$ part is right but $N_2$ should be 5, so this option fails.
(B) says $N_1 > N_2$: $10 > 5$ holds, so this option is true.
(C) says insertion sort makes only one swap: tracing the run above, the only actual element movement is the single shift of 5 at position 5, so this is true.
(D) says both algorithms make at least one unnecessary comparison: bubble sort keeps comparing already-ordered pairs like (1,2) in every pass, and insertion sort compares 1 with 2, 2 with 3, and 3 with 5, all already-ordered pairs, so this is also true.

**Step 5:** A quick sanity check on why the counts differ.
Bubble sort's inner loop shrinks by exactly one comparison every pass regardless of how the data looks, since it always walks the full unsorted portion of the array, so its total is fixed by the array length alone at $\binom{5}{2} = 10$. Insertion sort, on the other hand, only compares as far left as it needs to place the current key, so a nearly-sorted array like this one lets it finish with far fewer comparisons. That is exactly why $N_1 = 10$ is much larger than $N_2 = 5$ here, even though both algorithms sort the same five-element array.

**Final Answer:**
Options (B), (C) and (D) are correct.

$$\boxed{\text{(B), (C), (D)}}$$

**Quick Tip:** Trace both algorithms step by step on P = [1, 2, 3, 5, 4] and count every comparison made, including the ones between elements already in the right order.

• • •

**50.** Consider the given Python program.

```
def outer():
    x = []
    def inner(val):
        x.append(val)
        return x
    return inner

f1 = outer()
f2 = outer()
print(f1(10))   # Line P
print(f1(20))   # Line Q
print(f2(30))   # Line R
print(f1(40))   # Line S
```

Which of the following options is/are correct?

(A) f1 and f2 share the same list x

(B) Output of Line Q is [10, 20]

(C) Output of Line R is [10, 20, 30]

(D) Output of Line S is [10, 20, 40]

**Correct Answer:** (B) Output of Line Q is [10, 20] ; (D) Output of Line S is [10, 20, 40]

**SOLUTION:**

**Step 1:** Recall the closure rule.
A Python closure remembers the variables from its enclosing scope by reference, not by value, and a fresh copy of that scope is created on every call to the outer function. Since outer() is called twice, two independent scopes exist, each with its own x.

**Step 2:** Track state changes call by call.
f1 controls list $x_1$, which starts as [ ]. f2 controls list $x_2$, which starts as [ ] and is totally separate from $x_1$.

Call f1(10): acts on $x_1$, which goes from [ ] to [10]. Printed: [10].
Call f1(20): acts on $x_1$ again, which goes from [10] to [10, 20]. Printed: [10, 20].
Call f2(30): acts on $x_2$, which goes from [ ] to [30]. Printed: [30].
Call f1(40): acts on $x_1$ again, which goes from [10, 20] to [10, 20, 40]. Printed: [10, 20, 40].

**Step 3:** Match this to the options.
(A) is false because $x_1$ and $x_2$ never mix.
(B) matches the trace exactly ([10, 20] for Line Q), so it is true.
(C) does not match the trace (Line R gives [30], not [10, 20, 30]), so it is false.
(D) matches the trace exactly ([10, 20, 40] for Line S), so it is true.

**Step 4:** Why this happens under the hood.
Every call to outer() executes its body fresh, which means a new x = [] object is created and a new inner function object is built that closes over that specific x, not over any shared global slot. f1 and f2 hold references to two different inner function objects, each carrying its own closure cell pointing at its own list, so calling f1 repeatedly keeps mutating x1 while f2 keeps mutating a totally separate x2.

**Final Answer:**
The correct options are (B) and (D).

$$\boxed{\text{(B), (D)}}$$

**Quick Tip:** Remember that each call to outer() creates a brand new local list; f1 and f2 do not share state, but repeated calls to the same function do.

**51.** Consider a table Employee(EmpID, TeamID), where the column EmpID (ID of an employee) is the primary key. The column TeamID denotes the team ID of the team of which the employee is a member. TeamID is a NOT NULL column.

We want to display the size of the team (denoted as TeamSize) in which each employee is a member by using SQL. As an example, the desired output for the given Employee table is also shown in tabular form.

Which of the following is/are correct?

Employee table:

| EmpID | TeamID |
|---|---|
| 1 | 8 |
| 2 | 8 |
| 3 | 8 |
| 4 | 7 |
| 5 | 7 |
| 6 | 9 |

Output table:

| EmpID | TeamSize |
|---|---|
| 1 | 3 |
| 2 | 3 |
| 3 | 3 |
| 4 | 2 |
| 5 | 2 |
| 6 | 1 |

(A)

```
SELECT E.EmpID, B.TeamSize
FROM Employee AS E, (SELECT TeamID, COUNT(TeamID) AS TeamSize FROM Employee GROUP BY TeamID) AS B
WHERE E.TeamID = B.TeamID
```

(B)

```
SELECT A.EmpID, COUNT(B.TeamID) AS TeamSize
FROM Employee AS A, Employee AS B
WHERE A.TeamID = B.TeamID AND A.EmpID = B.EmpID
GROUP BY A.EmpID
```

(C)

```
SELECT B.EmpID, B.TeamSize
FROM (SELECT EmpID, COUNT(TeamID) AS TeamSize
FROM Employee GROUP BY EmpID) AS B
```

(D)

```
SELECT A.EmpID, B.TeamSize
FROM Employee AS A, (SELECT COUNT(TeamID) AS TeamSize FROM Employee GROUP BY TeamID) AS B
WHERE A.TeamID = B.TeamID
```

**Correct Answer:** (A)

```
SELECT E.EmpID, B.TeamSize
FROM Employee AS E, (SELECT TeamID, COUNT(TeamID) AS TeamSize FROM Employee GROUP BY TeamID) AS B
WHERE E.TeamID = B.TeamID
```

**SOLUTION:**

This question checks whether a SQL query correctly computes, for every employee, the number of employees who belong to the same team. The Employee table has EmpID as the primary key and a NOT NULL TeamID, and the target output attaches each employee's team size next to their EmpID. Let's test each candidate query against the sample data directly.

1. **Option (A)**: The inner query "SELECT TeamID, COUNT(TeamID) AS TeamSize FROM Employee GROUP BY TeamID" groups the six rows into three teams: TeamID 8 with 3 members, TeamID 7 with 2 members, TeamID 9 with 1 member. Joining this back onto Employee by TeamID gives EmpID 1, 2, 3 a TeamSize of 3, EmpID 4, 5 a TeamSize of 2, and EmpID 6 a TeamSize of 1, exactly matching the target output. This option is correct.
2. **Option (B)**: This query joins Employee to itself and restricts the match to rows where both TeamID and EmpID agree. Because EmpID alone already identifies a unique row, adding A.EmpID = B.EmpID collapses every match down to the single row itself, so COUNT(B.TeamID) is always 1 no matter which team the employee belongs to. This does not reproduce the target output, so this option is wrong.

3. **Option (C)**: Grouping by EmpID instead of TeamID is the key mistake here. Since every EmpID value appears exactly once, each group contains one row, so the count is always 1 for every employee. This never gives 3, 2, or 1 the way the target output needs, so this option is wrong.
4. **Option (D)**: The subquery computes COUNT(TeamID) AS TeamSize but drops TeamID from its own SELECT list. The outer WHERE clause then tries to use B.TeamID, a column the subquery never produced, so the statement is not valid SQL and cannot even run. This option is wrong.

Only option (A) both runs correctly and produces the exact TeamSize values shown in the target output table.

Let's summarize:

- Grouping by TeamID (not EmpID) is what gives the real per-team headcount.
- A self-join that also matches on the primary key EmpID collapses to one row per match, which defeats the purpose of counting teammates.
- A subquery cannot be referenced by a column it never selected.

So the correct query is option (A).

**Quick Tip:** Work out what each subquery actually returns on the sample data before joining it back to Employee; watch for grouping by the wrong column or a self-join that inadvertently matches only identical rows.

• • •

**52.** Let $M = \left(I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)$ be a matrix, where $\mathbf{1} = (1, 1, 1, \ldots, 1)^T \in \mathbb{R}^n$ and $I_n$ is the identity matrix of order $n$.

Which of the following options is/are correct?

(A) $M^T = M$

(B) $M^2 = I_n$

(C) $\text{Trace}(M) = n$

(D) $M$ is a projection matrix

**Correct Answer:** (A) $M^T = M$ ; (D) $M$ is a projection matrix

**SOLUTION:**

**Step 1:** Try a small concrete case to build intuition.

Take $n = 2$. Then $\mathbf{1} = (1, 1)^T$ and $\mathbf{1}\mathbf{1}^T = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$.

$$M = I_2 - \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} - \begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix} = \begin{pmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{pmatrix}$$

**Step 2:** Test each property on this concrete M.

$M^T = \begin{pmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{pmatrix} = M$, so $M$ is symmetric, consistent with option (A).

$M^2 = \begin{pmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{pmatrix}\begin{pmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{pmatrix} = \begin{pmatrix} 0.5 & -0.5 \\ -0.5 & 0.5 \end{pmatrix} = M$, not $I_2$, so option (B) fails here.

$\text{Trace}(M) = 0.5 + 0.5 = 1 = n - 1$ (since $n = 2$), not $n = 2$, so option (C) fails here too.
Since $M^T = M$ and $M^2 = M$, $M$ satisfies the definition of a projection matrix, supporting option (D).

**Step 3:** Confirm the pattern holds for general n algebraically.
For any $n$, $\mathbf{1}^T\mathbf{1} = n$ because it sums $n$ ones. Using this:

$$M^2 = I_n - \frac{2}{n}\mathbf{1}\mathbf{1}^T + \frac{1}{n^2}\mathbf{1}(\mathbf{1}^T\mathbf{1})\mathbf{1}^T = I_n - \frac{2}{n}\mathbf{1}\mathbf{1}^T + \frac{1}{n}\mathbf{1}\mathbf{1}^T = I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T = M$$

so $M$ is always idempotent, matching what the $n = 2$ case showed. Also $\text{Trace}(\mathbf{1}\mathbf{1}^T) = \mathbf{1}^T\mathbf{1} = n$, so $\text{Trace}(M) = n - \frac{1}{n} \cdot n = n - 1$ for every $n$, confirming option (C) is always false and option (B) is always false for $n > 1$.

**Step 4:** Conclude.
$M$ is symmetric and idempotent for every $n$, which by definition makes it a projection matrix (specifically, projection onto the hyperplane orthogonal to $\mathbf{1}$, the classic mean-centering operator in statistics). Options (A) and (D) hold in general, while (B) and (C) do not.

(A) and (D) are correct

**Quick Tip:** Use the identity $\mathbf{1}^T\mathbf{1} = n$ to expand $M^2$ and $\text{Trace}(M)$; a symmetric idempotent matrix is by definition a projection matrix.

• • •

**53.** Let $X_1, X_2, \ldots, X_n$ be $n$ independent random variables. Each of the random variables follows $Normal(\mu = 0, \sigma^2 = 1)$ distribution. Define $\bar{X} = \frac{1}{n}\sum_{i=1}^{n} X_i$.

Which of the following statements is/are correct?

(A) $\sum_{i=1}^{n} X_i^2$ follows Chi-square distribution with $n$ degrees of freedom.
(B) $\sum_{i=1}^{n} (X_i - \bar{X})^2$ follows Chi-square distribution with $(n-1)$ degrees of freedom.
(C) $X_1^2 + X_n^2$ follows exponential distribution with mean 2.
(D) $(\sqrt{n}\bar{X})^2$ follows Chi-square distribution with 2 degrees of freedom.

**Correct Answer:** (A) $\sum_{i=1}^{n} X_i^2$ follows Chi-square distribution with $n$ degrees of freedom. ; (B) $\sum_{i=1}^{n} (X_i - \bar{X})^2$ follows Chi-square distribution with $(n-1)$ degrees of freedom. ; (C) $X_1^2 + X_n^2$ follows exponential distribution with mean 2.

**SOLUTION:**

**Step 1:** List the two facts this question leans on.
Fact 1: if $Z \sim N(0,1)$ then $Z^2 \sim \chi_1^2$ (chi-square with 1 df).
Fact 2 (additivity): if $U \sim \chi_{k_1}^2$ and $V \sim \chi_{k_2}^2$ are independent, then $U + V \sim \chi_{k_1+k_2}^2$.
Both facts follow directly from how the chi-square distribution is built as a sum of squared independent standard normals.

**Step 2:** Option (A) by direct application of Fact 1 and Fact 2.
Each $X_i \sim N(0,1)$, so each $X_i^2 \sim \chi_1^2$ by Fact 1. Since the $X_i$ are independent, repeated use of Fact 2 gives $\sum_{i=1}^{n} X_i^2 \sim \chi_n^2$. This matches option (A), so it is true.

**Step 3:** Option (B) via the standard sample variance result.
For iid normal data, $\bar{X}$ and the deviations $(X_i - \bar{X})$ are independent, and $\sum (X_i - \bar{X})^2/\sigma^2$ has one fewer degree of freedom than $\sum X_i^2/\sigma^2$ because estimating $\bar{X}$ removes one degree of freedom. With $\sigma^2 = 1$, this gives $\sum_{i=1}^{n} (X_i - \bar{X})^2 \sim \chi_{n-1}^2$ directly, so option (B) is true.

**Step 4:** Option (C) using the Gamma-Exponential link.
By independence, $X_1^2 \sim \chi_1^2$ and $X_n^2 \sim \chi_1^2$ independently, so $X_1^2 + X_n^2 \sim \chi_2^2$ by Fact 2. A $\chi_2^2$ random variable has probability density $f(x) = \frac{1}{2}e^{-x/2}$ for $x \geq 0$, which is exactly the density of an Exponential distribution with rate $\frac{1}{2}$, i.e. mean $\frac{1}{1/2} = 2$. So option (C) is true.

**Step 5:** Option (D) by computing the distribution of $\bar{X}$ directly.
$\bar{X}$ is a linear combination of independent normals, so it is itself normal, with mean $E[\bar{X}] = 0$ and variance $Var(\bar{X}) = \frac{1}{n^2} \sum Var(X_i) = \frac{1}{n^2} \cdot n = \frac{1}{n}$.
So $\sqrt{n}\bar{X}$ has mean 0 and variance $n \cdot \frac{1}{n} = 1$, meaning $\sqrt{n}\bar{X} \sim N(0,1)$.
Squaring a single $N(0,1)$ variable gives $\chi_1^2$, not $\chi_2^2$, so option (D) is false.

**Final Answer:**
Statements (A), (B) and (C) are correct; (D) is not.

$$\boxed{\text{(A), (B), (C)}}$$

**Quick Tip:** Remember that a sum of k independent squared standard normals is Chi-square with k degrees of freedom, and Chi-square with 2 degrees of freedom is the same as Exponential with mean 2.

• • •

**54.** Let $X$ be a discrete valued random variable with cumulative distribution function $F(x)$.

Which of the following statements is/are correct?

(A) $F(x)$ is always a positive function.

(B) $F(x)$ is a non-decreasing function.

(C) $F(x)$ has jump discontinuity.

(D) $F(x)$ is a left continuous function.

**Correct Answer:** (B) $F(x)$ is a non-decreasing function. ; (C) $F(x)$ has jump discontinuity.

**SOLUTION:**

A cumulative distribution function (CDF) is defined as $F(x) = P(X \leq x)$. For a discrete random variable, this function is built entirely out of the probability masses at the individual points $X$ can take, which gives it a

very specific step-function shape. Let's check each statement about $F(x)$ against this definition.

1. **$F(x)$ is always a positive function**: A CDF is a probability, so it can never be negative, but it is not always strictly positive either. For any $x$ below the smallest value $X$ can take, $F(x) = 0$ exactly, since $P(X \leq x) = 0$ there. So calling it "always positive" is too strong; the correct statement would be "always non-negative". This option is wrong.
2. **$F(x)$ is a non-decreasing function**: Take any $x_1 < x_2$. The event $X \leq x_1$ is entirely contained inside the event $X \leq x_2$, so its probability cannot exceed the probability of the larger event. That means $F(x_1) \leq F(x_2)$ always, for any distribution, discrete or continuous. This option is correct.
3. **$F(x)$ has jump discontinuity**: Because $X$ is discrete, all of its probability is concentrated at a countable set of points. Right at each such point, $F(x)$ suddenly increases by that point's probability mass, then stays flat until the next point. This staircase behavior is precisely a jump discontinuity at every value $X$ can take. This option is correct.
4. **$F(x)$ is a left continuous function**: The standard definition $F(x) = P(X \leq x)$ already includes the value $x$ itself, so nudging $x$ slightly to the right does not change what is included, making $F$ right continuous. Nudging $x$ slightly to the left, though, excludes the probability mass sitting exactly at $x$, so the left-hand limit falls short of $F(x)$ at any jump point. So $F(x)$ is right continuous, not left continuous. This option is wrong.

Only the non-decreasing property and the jump discontinuity property hold for a discrete random variable's CDF.

Let's summarize:

- A CDF is non-negative, not strictly positive everywhere.
- Monotonic non-decreasing behavior is a universal CDF property, for any type of random variable.
- Discreteness of $X$ is exactly what produces jump discontinuities in $F(x)$.
- A CDF is right continuous by convention, never left continuous at a jump.

So the correct statements are (B) and (C).

**Quick Tip:** Check the CDF definition $F(x) = P(X \leq x)$ directly against each claim: it is non-negative and non-decreasing always, and being right continuous with jumps is specific to discrete random variables.

• • •

**55.** Consider that Linear Ridge Regression is being used to learn a prediction function $y_{pred} = w^T x$, where $w, x \in \mathbb{R}^2$ and Mean Absolute Error (MAE) is used to measure the prediction error. A weight of 0.20 is associated with the regularizer.

At an intermediate step of the training process, assume that the parameter $w = [-3.00,\ 4.00]^T$. In the next step, for the input $x = [1.00,\ 2.00]^T$, the predicted value of $y$ is noted. Let the relation between $x = [x_1,\ x_2]^T$ and the true value of $y$ be $y_{true} = x_1 + x_2$.

The value of the overall regularized loss function for this instance is _____ . (Rounded off to two decimal places)

**Correct Answer:** 7.00

**SOLUTION:**

**Step 1:** Write down the Ridge-regularized MAE loss for one sample.

$$L(w) = \underbrace{|w^T x - y_{true}|}_{\text{MAE term}} + \underbrace{\lambda(w_1^2 + w_2^2)}_{\text{L2 penalty}}$$

with $\lambda = 0.20$ given as the regularizer weight.

**Step 2:** Plug in the numbers for the prediction error piece first.
$w^T x = (-3.00)(1.00) + (4.00)(2.00) = 5.00$, and separately $y_{true} = x_1 + x_2 = 1.00 + 2.00 = 3.00$.
The absolute error is $|5.00 - 3.00| = 2.00$.

**Step 3:** Work out the penalty piece independently.
The squared norm of the weight vector is $w_1^2 + w_2^2 = (-3.00)^2 + (4.00)^2 = 9.00 + 16.00 = 25.00$.
Scaling by the regularizer weight: $0.20 \times 25.00 = 5.00$.

**Step 4:** Combine the two independent pieces.

$$L = 2.00 + 5.00 = 7.00$$

Since the MAE term and the L2 penalty term are computed from completely separate quantities (prediction error vs. weight magnitude), they simply add.

**Step 5:** Sanity check the magnitude.
$w = [-3, 4]$ is a fairly large weight vector (norm 5), so a regularizer weight of 0.20 on its squared norm contributing 5.00 to a total loss of 7.00 makes sense, the penalty dominates the raw prediction error here.

$$\boxed{L = 7.00}$$

**Quick Tip:** Compute the MAE term |w^T x - y_true| and the L2 penalty lambda times ||w||^2 separately, then add them together.

• • •

**56.** Consider a fully-connected feed-forward multi-layer perceptron. It has 30 neurons in the input layer, followed by two hidden layers and an output layer. The first hidden layer has 4 neurons and the second hidden layer has 3 neurons. The output layer has only one neuron. Assume that no bias parameters are used.

The number of learnable parameters in the multi-layer perceptron is _____ . (Answer in integer)

**Correct Answer:** 135

**SOLUTION:**

**Step 1:** Model each layer transition as a weight matrix.
A fully-connected layer that maps from a layer with $p$ neurons to a layer with $q$ neurons (no bias) has a weight matrix of shape $q \times p$, contributing exactly $p \times q$ learnable numbers.

**Step 2:** List the layer sizes in order.

$$30 \to 4 \to 3 \to 1$$

This network has three transitions: input to hidden-1, hidden-1 to hidden-2, and hidden-2 to output.

**Step 3:** Compute the parameter count for each transition as a matrix size.
Transition 1 (input to hidden-1): weight matrix of shape $4 \times 30$, giving $4 \times 30 = 120$ entries.
Transition 2 (hidden-1 to hidden-2): weight matrix of shape $3 \times 4$, giving $3 \times 4 = 12$ entries.
Transition 3 (hidden-2 to output): weight matrix of shape $1 \times 3$, giving $1 \times 3 = 3$ entries.

**Step 4:** Sum the sizes of all weight matrices.

$$\text{Total} = 120 + 12 + 3 = 135$$

Since no bias vectors exist in this network, this sum of matrix entries is the entire parameter count, nothing else to add.

**Step 5:** Confirm nothing was double counted.
Each pair of consecutive layers contributes exactly one weight matrix, and there are exactly 3 consecutive layer pairs for 4 layers total (input, hidden-1, hidden-2, output), so all three products were needed and none overlap.

$$\boxed{135}$$

**Quick Tip:** With no bias terms, count only the weight connections between each pair of consecutive layers (input to output) and add them up: 30x4 + 4x3 + 3x1.

• • •

**57.** A clinic specializes in testing for a disease $D$. The result of the test can be either positive or negative.

A study revealed that if a person suffers from the disease $D$, the test result in that clinic comes out positive 80% of the time, and negative 20% of the time. If a person is not suffering from the disease $D$, the test comes out positive 10% of the time and negative 90% of the time. It is also known that among the general population, the disease $D$ occurs in 30% of the individuals.

If a person tests positive for $D$ in that clinic, the probability that he/she actually suffers from the disease $D$ is _____. (Rounded off to two decimal places)

**Correct Answer:** 0.77

**SOLUTION:**

Take an imaginary group of 100000 people from the general population and track them through the test, since working with actual counts instead of fractions makes the logic easier to follow.

**Step 1: Split the population by disease status.** Since 30% of people have the disease $D$, out of 100000 people:

$$\text{People with } D = 100000 \times 0.30 = 30000$$

$$\text{People without } D = 100000 \times 0.70 = 70000$$

**Step 2: Apply the test outcome rates to each group.** Among the 30000 people with the disease, the test is positive 80% of the time:

$$\text{True positives} = 30000 \times 0.80 = 24000$$

Among the 70000 people without the disease, the test is positive 10% of the time (a false alarm):

$$\text{False positives} = 70000 \times 0.10 = 7000$$

**Step 3: Count everyone who tests positive.**

$$\text{Total positives} = 24000 + 7000 = 31000$$

This group of 31000 people is everyone the clinic reports as positive, whether or not they are actually sick.

**Step 4: Find what fraction of the positives are actually sick.** Out of the 31000 positive results, only 24000 come from people who genuinely have the disease. So the chance a positive person is truly sick is:

$$P(D \mid \text{positive}) = \frac{24000}{31000} = 0.7742$$

Rounded to two decimal places, this gives 0.77.

$$\boxed{0.77}$$

**Quick Tip:** Use Bayes' theorem: $P(D|\text{positive}) = \dfrac{P(\text{positive}|D)P(D)}{P(\text{positive}|D)P(D) + P(\text{positive}|D^c)P(D^c)}$.

• • •

**58.** Consider the given Python program.

```
def fun(L, i=0):
    if i >= len(L)-1:
        return 0
    if L[i] > L[i+1]:
        L[i+1], L[i] = L[i], L[i+1]
        return 1+fun(L, i+1)
    else:
        return fun(L, i+1)

data = [5, 3, 4, 1, 2]
count = 0
for _ in range(len(data)):
    count += fun(data)
print(count)
```

The output of the program is _____. (Answer in integer)

**Correct Answer:** 8

**SOLUTION:**

Instead of simulating every pass step by step, use a shortcut: each call to fun does one left-to-right bubble pass, and every time it swaps two adjacent elements, it fixes exactly one "inversion" (a pair of positions where the earlier element is bigger than a later element). Running enough passes to fully sort a 5-element list means every inversion gets fixed exactly once across all the passes, so the total count printed at the end equals the number of inversions present in the original array.

First count the inversions in the starting array **[5, 3, 4, 1, 2]**. An inversion is any pair of positions **(*i*, *j*)** with ***i*** **<** ***j*** where the value at ***i*** is bigger than the value at ***j***. Check every pair:

1. (5, 3): **5 > 3**, inversion.
2. (5, 4): **5 > 4**, inversion.
3. (5, 1): **5 > 1**, inversion.
4. (5, 2): **5 > 2**, inversion.
5. (3, 4): **3 < 4**, not an inversion.
6. (3, 1): **3 > 1**, inversion.
7. (3, 2): **3 > 2**, inversion.
8. (4, 1): **4 > 1**, inversion.
9. (4, 2): **4 > 2**, inversion.
10. (1, 2): **1 < 2**, not an inversion.

That gives 8 inversions out of the 10 possible pairs.

Now check that 5 passes is really enough to clear all of them. A single left-to-right bubble pass is guaranteed to move the largest remaining unsorted element all the way to its correct position at the far right. With 5 elements, at most 4 passes are ever needed to fully sort the list, and the loop here runs 5 passes, one more than needed. So by the time all 5 calls to fun finish, the list is completely sorted and every one of the 8 original inversions has been resolved by exactly one adjacent swap.

Since each swap adds exactly 1 to count and there are 8 swaps in total across the 5 passes, count ends at 8, and that is what print(count) outputs.

8

**Quick Tip:** Each call to fun performs one bubble-sort pass and counts its swaps; running it 5 times over a 5-element list is enough to fully sort it, so the total equals the number of inversions in the original list.

• • •

**59.** Let there be two relations $X$ and $Y$ as shown. $X$ has three columns $P$, $Q$ and $R$. $Y$ has two columns $P$ and $S$.

**Relation X:**

| P | Q | R |
|---|---|---|
| P1 | Q1 | R1 |
| P2 | Q2 | R2 |
| P3 | Q3 | R2 |

**Relation Y:**

| P | S |
|---|---|
| P1 | 10 |
| P1 | 15 |
| P2 | 20 |
| P3 | 1 |

Consider that the following tuple relational calculus expression is evaluated.

$$\{t \mid t \in X \wedge \exists z \in X(t[P] = z[P]) \wedge \exists m \in Y(m[P] = t[P] \wedge m[S] > 1)\}$$

The number of tuples that will be returned is _____. (Answer in integer)

**Correct Answer:** 2

**SOLUTION:**

A clean way to read this tuple relational calculus expression is to translate it into an equivalent SQL query, since that makes the logic more familiar. The condition $\exists z \in X(t[P] = z[P])$ is a self-reference that is always

satisfied (just pick $z = t$), so it contributes nothing to the filtering and can be dropped. What remains is: keep a tuple $t$ from $X$ only if there is some row in $Y$ with the same $P$ value and $S > 1$. In SQL terms, this is:

SELECT * FROM X WHERE P IN (SELECT P FROM Y WHERE S > 1)

1. **Find which P values in Y have S > 1.** Y's rows are (P1, 10), (P1, 15), (P2, 20), (P3, 1). Filtering for S > 1 keeps (P1, 10), (P1, 15), and (P2, 20), while (P3, 1) is dropped since 1 is not greater than 1. So the surviving P values are P1 and P2.
2. **Match these P values against X.** X's rows are (P1, Q1, R1), (P2, Q2, R2), (P3, Q3, R2). The row with P1 matches, the row with P2 matches, but the row with P3 does not, since P3 was excluded from the surviving P values in step 1.

So exactly two rows of X, the ones with P1 and P2, satisfy the original expression, while the P3 row is excluded because its only matching row in Y has S = 1, which fails the strict S > 1 test.

2

**Quick Tip:** The self-reference on X is always true; the real filter checks whether each tuple's P value appears in Y with an S value strictly greater than 1.

• • •

**60.** Let Account be a relation as shown.

**Table: Account**

| AccNo | Balance |
|---|---|
| A1 | 5000 |
| A2 | 5000 |
| A3 | 10000 |
| A4 | 15000 |
| A5 | 18000 |

Consider the given SQL query.

```
SELECT AccNo FROM Account AS A
WHERE (SELECT COUNT(*) FROM Account AS B
WHERE A.Balance < B.Balance) >= (SELECT COUNT(*)
FROM Account AS C WHERE A.Balance > C.Balance)
```

The number of rows returned by the SQL query is _____. (Answer in integer)

**Correct Answer:** 3

**SOLUTION:**

A quicker way to see this is to first sort the balances and think in terms of rank. Sorted in increasing order, the balances are 5000, 5000, 10000, 15000, 18000, held by A1/A2, A1/A2, A3, A4, A5 respectively (A1 and A2 tie at 5000).

For any account A, countGreater is simply how many of the 5 balances are strictly above A's balance, and countLess is how many are strictly below it. The condition countGreater $\geq$ countLess is really asking whether an account sits at or below the "middle" of the distribution: values on the high end will always have more accounts below them than above, so they fail the test, while values on the low or middle end pass.

1. **A1 and A2 (5000, the smallest value, tied):** Nothing is smaller, so countLess $= 0$. Everything except the other tied account is bigger, so countGreater $= 3$. Since $3 \geq 0$, both pass.
2. **A3 (10000, the middle value):** Two balances (5000, 5000) are smaller and two (15000, 18000) are bigger, so countLess $= 2$ and countGreater $= 2$. Since $2 \geq 2$, it passes.
3. **A4 (15000):** Three balances are smaller (5000, 5000, 10000) and only one is bigger (18000), so countLess $= 3$ and countGreater $= 1$. Since $1 \geq 3$ is false, it fails.
4. **A5 (18000, the largest value):** All four other balances are smaller, so countLess $= 4$ and countGreater $= 0$. Since $0 \geq 4$ is false, it fails.

So the accounts that pass the filter are exactly those at or below the median balance: A1, A2, and A3. That gives 3 rows in the result.

3

**Quick Tip:** For each account, compare how many accounts have a strictly higher balance against how many have a strictly lower balance; only rows at or below the median pass.

• • •

**61.** Consider an ER model with the entities E1(A11, A12, A13) and E2(A21, A22, A23), where A11, A12, A13 are the attributes of E1, and A21, A22, A23 are the attributes of E2. Let A22 be a multi-valued attribute. A11 and A21 are the primary keys of E1 and E2, respectively.

Let R12 be a many-to-many relationship between E1 and E2. Participation of both E1 and E2 in R12 is total.

The minimum number of relations required to convert the ER model into relational model (assuming there is no other functional dependency) where each relation is in third normal form (3NF) is _____. (Answer in integer)

**Correct Answer:** 4

**SOLUTION:**

Instead of deriving the relations one by one, count by category: how many strong entities are there, how many multi-valued attributes, and how many many-to-many relationships, since each of these always needs a dedicated relation of its own under the standard mapping rules.

1. **Strong entities: E1 and E2.** Both are regular strong entities with their own primary keys (A11 and A21). Each strong entity always becomes exactly one relation on its own, contributing 2 relations so far: E1(A11, A12, A13) and E2(A21, A23), since A22 does not stay with E2.
2. **Multi-valued attributes: just A22.** A multi-valued attribute always gets pulled out into its own relation keyed on (owner's primary key, the attribute itself), because a relational column can only hold one atomic value per row. This adds 1 relation: A22_table(A21, A22). Total so far: 3.
3. **Many-to-many relationships: just R12.** A relationship with M:N cardinality can never be folded into either side's entity table, since that side would then need to store a variable number of foreign keys. It always becomes its own relation holding the primary keys of the two entities it connects, regardless of whether participation is total or partial. This adds 1 more relation: R12(A11, A21). Total: 4.

Total participation would only save a relation if the relationship were one-to-many, since then the "many" side's table could absorb the foreign key of the "one" side. R12 is many-to-many, so that shortcut does not apply, and it must stay as its own bridge table.

Adding up: 2 entity relations + 1 multi-valued attribute relation + 1 relationship relation gives 4 relations in total, and since there is no other functional dependency, each is already in 3NF without further splitting.

4

**Quick Tip:** Count separately: one relation per strong entity, one relation for the multi-valued attribute, and one relation for the many-to-many relationship, since M:N relationships always need their own table regardless of total participation.

• • •

**62.** For a given data set $\{x_1, x_2, \ldots, x_n\}$, where $n = 100$, it is known that

$$\frac{1}{2000}\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i - x_j)^2 = 99$$

Let us denote $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

The value of $\frac{1}{99}\sum_{i=1}^{n}(x_i - \bar{x})^2$ is _____. (Answer in integer)

**Correct Answer:** 10

**SOLUTION:**

Think of picking an index $i$ uniformly at random from 1 to $n$ and independently picking another index $j$ the same way, then looking at $x_i$ and $x_j$ as two independent draws $X$ and $Y$ from the same empirical distribution (the list of data values, each with probability $1/n$). Averaging $(x_i - x_j)^2$ over all $n^2$ ordered pairs $(i, j)$ is exactly the expected value $E[(X - Y)^2]$ for these two independent copies.

For any two independent random variables $X$ and $Y$ with the same distribution and mean $\mu$, a standard probability identity gives:

$$E[(X-Y)^2] = E[X^2] - 2E[X]E[Y] + E[Y^2] = 2E[X^2] - 2\mu^2 = 2\,\mathrm{Var}(X)$$

where $\mathrm{Var}(X) = E[X^2] - \mu^2$ is the population variance of the data, namely $\mathrm{Var}(X) = \frac{1}{n}\sum_i (x_i - \bar{x})^2$.

So we have the relation:

$$\frac{1}{n^2}\sum_{i=1}^{n}\sum_{j=1}^{n}(x_i - x_j)^2 = 2\,\mathrm{Var}(X) = \frac{2}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Now plug in the given information. We are told $\frac{1}{2000}\sum_i\sum_j (x_i - x_j)^2 = 99$, so $\sum_i\sum_j (x_i - x_j)^2 = 198000$. With $n = 100$, $n^2 = 10000$, so:

$$\frac{198000}{10000} = 19.8 = \frac{2}{100}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Solve for the sum of squared deviations:

$$\sum_{i=1}^{n}(x_i - \bar{x})^2 = 19.8 \times \frac{100}{2} = 19.8 \times 50 = 990$$

Finally, divide by 99 as the question asks:

$$\frac{1}{99}\sum_{i=1}^{n}(x_i - \bar{x})^2 = \frac{990}{99} = 10$$

$$\boxed{10}$$

**Quick Tip:** Use the identity $\sum_i\sum_j (x_i - x_j)^2 = 2n\sum_i (x_i - \bar{x})^2$ to convert the given double sum into the sum of squared deviations from the mean.

• • •

**63.** Let $X$ be a random variable that follows $Uniform(-1, 1)$ distribution. The conditional distribution of the random variable $Y$ given $X = x$ is the $Uniform(x^2 - 0.1, x^2 + 0.1)$ distribution.

The value of $correlation(X, Y)$ is _____. (Answer in integer)

**Correct Answer:** 0

**SOLUTION:**

A cleaner way to see this is with a pure symmetry argument, without directly computing any integral for $E[X^3]$.

Since $X$ is $Uniform(-1, 1)$, its distribution is perfectly symmetric around 0: the random variable $-X$ has exactly the same distribution as $X$. Now look at how $Y$ is generated: given $X = x$, $Y$ is drawn uniformly from an interval centered at $x^2$. Since $(-x)^2 = x^2$, the conditional distribution of $Y$ given $X = x$ is identical to the conditional distribution of $Y$ given $X = -x$. In other words, $Y$'s distribution only depends on $x$ through $x^2$, so $Y$ is completely blind to the sign of $X$.

Now consider the pair $(X, Y)$ versus the pair $(-X, Y)$. Because of the symmetry just described, these two pairs have exactly the same joint distribution: for every value of $X$, the conditional law of $Y$ is unchanged if we flip

the sign of $X$, and $X$ itself is equally likely to be positive or negative. This means:

$$E[XY] = E[(-X)Y] = -E[XY]$$

The only number that equals its own negative is 0, so $E[XY] = 0$ follows immediately without any integration.

Also, $E[X] = 0$ since $Uniform(-1, 1)$ is symmetric about the origin. So the covariance is:

$$Cov(X, Y) = E[XY] - E[X]E[Y] = 0 - 0 \cdot E[Y] = 0$$

Both $X$ and $Y$ have finite positive variance ($X$ is a proper uniform variable, and $Y$'s conditional spread is a fixed constant plus the extra variation coming from $X^2$, so $\sigma_Y$ is finite and nonzero). With the numerator of the correlation formula equal to 0, the correlation itself must be 0:

$$correlation(X, Y) = \frac{Cov(X, Y)}{\sigma_X \sigma_Y} = \frac{0}{\sigma_X \sigma_Y} = 0$$

$$\boxed{0}$$

**Quick Tip:** Use the tower property: $E[XY] = E[X \cdot E[Y|X]] = E[X^3]$, and note that $X^3$ is an odd function of a symmetric variable, so its expectation is 0, making the covariance and hence the correlation 0.

• • •

**64.** Let $A_{5\times5}$ be a matrix such that each of its elements follows $Bernoulli(p = 0.50)$ distribution independently.

The probability that the row-sum of the second row and the column-sum of the third column are both equal to 3 is _____. (Rounded off to two decimal places)

**Correct Answer:** 0.10

**SOLUTION:**

**Step 1:** Understanding the Concept:
Since the matrix has 25 independent Bernoulli(0.5) entries, every one of the 25 cells is an independent fair coin flip. Only 9 of those cells actually affect the event we care about: the 4 remaining cells of row 2, the 4 remaining cells of column 3, and the single cell $A_{23}$ shared by both. The other 16 cells can be anything and do not change the probability.

**Step 2:** Key Formula or Approach:
Treat the 9 relevant cells as 9 independent fair coins, giving $2^9 = 512$ equally likely outcomes. Count how many of these 512 outcomes make the row-2 sum equal to 3 and the column-3 sum equal to 3 at the same time, then divide by 512.

**Step 3:** Detailed Explanation:
Split the count by the value of the shared cell $A_{23}$.
If $A_{23} = 0$: the other 4 cells of row 2 must add to 3, which can happen in $\binom{4}{3} = 4$ ways, and the other 4 cells of column 3 must also add to 3, again $\binom{4}{3} = 4$ ways. That gives $4 \times 4 = 16$ favorable outcomes for this branch.

If $A_{23} = 1$: the other 4 cells of row 2 must now add to 2 (since $2 + 1 = 3$), which happens in $\binom{4}{2} = 6$ ways, and the other 4 cells of column 3 must also add to 2, again $\binom{4}{2} = 6$ ways. That gives $6 \times 6 = 36$ favorable outcomes for this branch.
Total favorable outcomes $= 16 + 36 = 52$.
Probability $= \frac{52}{512} = 0.1015625$.

**Step 4:** Final Answer:
Rounding 0.1015625 to two decimal places gives 0.10, which matches the value found from splitting on the shared cell's value directly.

$$\boxed{0.10}$$

**Quick Tip:** Split the row-2 sum and column-3 sum around the one cell they share, $A_{23}$. Condition on $A_{23}$ being 0 or 1, then use $Binomial(4, 0.5)$ for the remaining four cells in each.

• • •

**65.** Let $A = \left(I_n - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right)$ be a matrix, where $\mathbf{1} = (1, 1, 1, \ldots, 1)^T \in \mathbb{R}^n$ and $I_n$ is the identity matrix of order $n$.

The value of $\max_S x^T Ax$, where $S = \{x \in \mathbb{R}^n \mid x^T x = 1\}$, is _____. (Answer in integer)

**Correct Answer:** 1

**SOLUTION:**

This question asks for the maximum of the quadratic form $x^T Ax$ over all unit vectors, a classic constrained optimization problem. Instead of jumping straight to eigenvalue facts, let's set it up with Lagrange multipliers first and then work out the actual numbers for this specific matrix $A$.

**Setting up the optimization:** We want to maximize $f(x) = x^T Ax$ subject to $g(x) = x^T x - 1 = 0$. At the maximum, the gradients must be parallel: $\nabla f = \lambda \nabla g$ for some scalar $\lambda$. Since $A$ is symmetric, $\nabla f = 2Ax$ and $\nabla g = 2x$, so this condition reduces to $Ax = \lambda x$. In other words, the maximizer $x$ must be a unit eigenvector of $A$, and the maximum value of $x^T Ax$ equals the corresponding eigenvalue $\lambda$, since at an eigenvector $x^T Ax = x^T(\lambda x) = \lambda(x^T x) = \lambda$.

**Finding the eigenvalues by testing vectors directly:** Rather than proving the general idempotent-matrix fact first, plug in two natural test vectors.

1. **Test $x = \frac{1}{\sqrt{n}}\mathbf{1}$ (unit vector along the all-ones direction):** $\mathbf{1}^T x = \frac{1}{\sqrt{n}}(\mathbf{1}^T\mathbf{1}) = \frac{n}{\sqrt{n}} = \sqrt{n}$. So $Ax = x - \frac{1}{n}\mathbf{1}(\mathbf{1}^T x) = x - \frac{1}{n}\mathbf{1}\sqrt{n} = x - \frac{1}{\sqrt{n}}\mathbf{1} = x - x = 0$. This direction gives eigenvalue 0, so $x^T Ax = 0$ here, not the maximum.
2. **Test any unit $x$ with $\mathbf{1}^T x = 0$ (entries summing to zero, for example $x = \frac{1}{\sqrt{2}}(1, -1, 0, \ldots, 0)^T$):** $Ax = x - \frac{1}{n}\mathbf{1}(\mathbf{1}^T x) = x - \frac{1}{n}\mathbf{1} \cdot 0 = x$. So $x^T Ax = x^T x = 1$. This direction gives eigenvalue 1.

Squaring $A$ shows $A^2 = A$, so for any eigenvalue $\lambda$ we get $\lambda^2 = \lambda$, forcing $\lambda \in \{0, 1\}$ only. We already found a direction giving 0 and a direction giving 1, so no other value is possible, and the largest achievable value of $x^T Ax$ on the unit sphere is 1.

Let's summarize:

- The maximizer of $x^T Ax$ on the unit sphere is always a top eigenvector of $A$, by Lagrange multipliers.
- Testing the all-ones direction gives eigenvalue 0, and any zero-sum direction gives eigenvalue 1.
- Squaring $A$ confirms these are the only two possible eigenvalues.

So $\max_S x^T Ax = 1$.

$$\boxed{1}$$

**Quick Tip:** Check that $A$ is symmetric and idempotent ($A^2 = A$), so its eigenvalues can only be 0 or 1. The maximum of $x^T Ax$ on the unit sphere equals the largest eigenvalue.