There are five (5) problems in this examination.

There should be sufficient room in this booklet for all your work. But if you use other sheets of paper, be sure to mark them clearly and staple them to the booklet.

**Problem 1**

In a game, our goal is to determine exactly an unknown integer, $1, 2, \ldots, 1024 = 2^{10}$, chosen uniformly at random, by asking yes/no questions.

(a) What is the expected number of questions needed if we ask: Is it 1? Is it 2? (asking integers sequentially).

(b) What is the expected number of questions if we eliminate exactly half the possibilities with each question?

Show work. (Hint: Consider cases smaller than 1024 to check for off-by-1 errors.)

**Solution**

Ross 4.26, p. 175.

Put $N = 1024$. In the first case, if the random number is $i < N$, it takes $i$ questions.

Note that, with a careful reading, we never have to ask the $N$’th question, because we know the answer after $N − 1$ nos.

So the expected number is
\[ \frac{1}{N} \sum_{i=1}^{N-1} i + \frac{N-1}{N} = \frac{1}{N} \frac{N+1}{2} - \frac{1}{N} = \frac{N+1}{2} - \frac{1}{N} = \frac{N^2+N-2}{2N}. \]

An alternative, deprecated, reading requires the last question (and gives a prettier answer of) $\frac{N+1}{2}$.

In the second case, we never learn the number exactly until the last question, the tenth, at which point we always learn the number exactly. So the expected number of questions is 10.

**Mathematical concepts:** discrete random variables, expectation

**Problem 2**

Alice and Bob agree to meet at M36 coffee shop, but each arrives independently at random between noon and 1pm. What is the probability that the first to arrive has to wait more than 10 minutes?

**Solution**
Ross, pp. 243, Example 2c.

Let $X$ and $Y$ be the arrival times. By symmetry, we want $2 \Pr(X + 10 < Y)$. This is twice the relative area of a triangle within the 60-by-60 square bounded by $X = 0, Y = 60, Y = X + 10$. This is a 45-45-90 triangle with short side 50 minutes, so twice the relative area is $\left(\frac{50}{60}\right)^2 = \frac{25}{36}$.

Alternatively, the area can be found using an integral.

**Mathematical concepts:** Independent random variables

---

**Problem 3**

Find the number of integers between 1 and 24,000, inclusive, that are not divisible by 4, 5, or 6.

**Solution**

Brualdi 6.7.1, page 198.

Note that 4 and 6 share a factor, and note that 4·5·6 divides 24,000.

Let $A_4, A_5, A_6$ be the sets of multiples of 4, 5, 6, respectively. Then we want:

$$24000 - (|A_4| + |A_5| + |A_6|) + (|A_4 \cap A_5| + |A_4 \cap A_6| + |A_5 \cap A_6|) - |A_4 \cap A_5 \cap A_6|.$$

This is $24000 \left(1 - \left(\frac{1}{4} + \frac{1}{5} + \frac{1}{6}\right) + \left(\frac{1}{20} + \frac{1}{12} + \frac{1}{30}\right) - \frac{1}{36}\right)$.

Here we used that $|A_4 \cap A_6| = 24000 \cdot \frac{1}{12}$, not 24000 · $\frac{1}{48}$, and similarly for $A_4 \cap A_5 \cap A_6$.

**Mathematical concepts:** Inclusion-exclusion principle

---

**Problem 4**

Show that, for any sequence of $n^2 + 1$ distinct real numbers $a_1, a_2, \ldots, a_{n^2+1}$, either there is an increasing subsequence of length $n + 1$ or a decreasing subsequence of length $n + 1$.

Note: A subsequence is gotten by deleting zero or more elements from the original sequence, while retaining the order of the remaining terms. “Increasing” or “decreasing” refers to values. For example, given sequence $(5, 1, 2, 9, 8)$, the sequence $(5, 2)$ is a subsequence, gotten by striking 1, 9, 8 and leaving 5 and 2 in their original order with 5 preceding 2. It is decreasing, since $2 < 5$.

**Solution**

Brualdi, p.76, Application 9

Let $m_k$ denote the length of the longest increasing subsequence that begins at $k$. Assume that, for all $k$, we have $m_k \leq n$. Since there are $n^2 + 1$ such $m_k$’s, the strong pigeonhole principle guarantees at least $n + 1$ equal $m_k$’s, say $m_{k_1} = m_{k_2} = \cdots = m_{k_{n+1}}$, where we take $k_1 < k_2 < \cdots < k_{n+1}$.

Because each $m_{k_i}$ is defined in terms of an increasing sequence of maximal length, we must have $a_{k_1} > a_{k_2} > \cdots > a_{k_{n+1}}$.

**Mathematical concepts:** (Strong) pigeonhole principle

---

**Problem 5**

We are given a sequence of matrix sizes compatible for multiplication (though not the matrices themselves). The matrices will be multiplied in the usual way. Our task is to insert parentheses to make the multiplication fastest.

That is, we are given a sequence $a_0, a_1, a_2, \ldots$, that we interpret as describing a multiplication $A_0A_1A_2\cdots$, where $A_0$ has $a_0$ rows and $a_1$ columns, $A_1$ has $a_1$ rows and $a_2$ columns, etc. If we choose parentheses $(A_0A_1)A_2$, the cost is $a_0a_1a_2$ to multiply $A_0A_1$ and get a $a_0$-by-$a_2$ matrix $A'$, plus $a_0a_2a_3$ to multiply $A'A_2$. On the other hand, parenthesization $A_0(A_1A_2)$ has cost $a_0a_1a_3 + a_1a_2a_3$. 

2
As a more concrete example, on input \((3, 1, 4, 1)\),

\[
\begin{pmatrix}
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\end{pmatrix}
\begin{pmatrix}
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\end{pmatrix}
\]

has cost \(3 \cdot 1 \cdot 4 + 3 \cdot 4 \cdot 1 = 24\) but

\[
\begin{pmatrix}
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\end{pmatrix}
\begin{pmatrix}
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
\end{pmatrix}
\]

has cost \(3 \cdot 1 \cdot 1 + 1 \cdot 4 \cdot 1 = 7 < 24\), so the latter parenthesization is preferable.

Give an algorithm that takes time polynomial in the number \(n\) of matrices and finds the best parenthesization.

**Solution**

We solve the problem and store results for each contiguous subsequence of matrices from the \(i\)’th to the \(j\)’th, in order of increasing \(j - i\). If \(j = i + 1\), the only choice gives cost \(a_i a_{i+1} a_{i+2}\). For larger \(j - i\), try all \(j - i\) highest-level parenthesization into two matrices \((A_i \cdots A_k)(A_{k+1} \cdots A_j)\), and use the precomputed results for the two matrices, in constant time. There are at most \(n\) choices for \(k\) to consider for each of at most \(n^2\) subsequences \(i \ldots j\). This gives runtime \(O(n^3)\).

**Mathematical concepts:** Dynamic programming