Problem 1. In our class we used the Blum-Floyd-Rivest-Pratt-Tarjan select algorithm using $n/5$ blocks of size 5, each. Now, consider the generalization of this algorithm in which you would partition the input array into n/b blocks of size b, where b is a constant. You may assume b to be odd. Answer the following questions:

1. Find the number of comparisons for each step of the algorithm.

Solution:
Sorting each group of $\frac{n}{b}$ elements: $\frac{b(b-1)}{2} \cdot \frac{n}{b} = \frac{n(b-1)}{2}$
Finding the median of medians: $T\left(\frac{n}{b}\right)$
Number of comparisons to partition the input sequence: $n-1$
Proper partition size: $n - \left(\frac{b+1}{2} \cdot \frac{1}{2} \cdot \frac{n}{b}\right) = \frac{n}{4}\left(3 - \frac{1}{b}\right)$
Recurrence over the proper partition: $T\left(\frac{n}{4}\left(3 - \frac{1}{b}\right)\right)$

2. Obtain the overall recurrence equation for the upper bound on the number of comparisons and solve it.

Solution:

$$
\begin{aligned}
T(n) &\leq \frac{n(b-1)}{2} + T\left(\frac{n}{b}\right) + n - 1 + T\left(\frac{n}{4}\left(3 - \frac{1}{b}\right)\right) \\
&= T\left(\frac{n}{b}\right) + T\left(\frac{n}{4}\left(3 - \frac{1}{b}\right)\right) + \frac{n(1+b)}{2} - 1
\end{aligned}
$$

Solving the recurrence using constructive induction.
Guess: $T(n) \leq an$

$$
\begin{aligned}
T(n) &\leq \frac{an}{b} + \frac{an}{4}\left(3 - \frac{1}{b}\right) + \frac{n(1+b)}{2} - 1 \\
&= \left(\frac{a}{b} + \frac{a}{4}\left(3 - \frac{1}{b}\right) + \frac{1+b}{2}\right)n - 1 \\
&= \left(\frac{3a}{4} + \frac{3a}{4b} + \frac{1+b}{2}\right)n - 1 \\
&\leq an
\end{aligned}
$$

Thus,

$$\begin{aligned}
\frac{3a}{4} + \frac{3a}{4b} + \frac{1+b}{2} &= a \\
\frac{3a}{4} + \frac{3a}{4b} - a &= -\frac{1+b}{2} \\
\left(\frac{3}{b} - 1\right)\frac{a}{4} &= -\frac{1+b}{2} \\
\left(1 - \frac{3}{b}\right)\frac{a}{2} &= 1+b \\
a &= \frac{2b(1+b)}{b-3}
\end{aligned}$$

Therefore,

$$T(n) \leq \left(\frac{3a}{4} + \frac{3a}{4b} + \frac{1+b}{2}\right)n - 1$$

Substituting $a = \frac{2b(1+b)}{b-3}$, we get

$$\begin{aligned}
T(n) &\leq \left(\frac{3 \cdot 2b(1+b)}{4(b-3)} + \frac{3 \cdot 2b(1+b)}{4b(b-3)} + \frac{1+b}{2}\right)n - 1 \\
&= \left(\frac{2b(1+b)}{b-3}\right)n - 1
\end{aligned}$$

3. Verify your result when $b = 5$.

Solution: when $b = 5$,

$$\begin{aligned}
T(n) &\leq \left(\frac{2b(1+b)}{b-3}\right)n - 1 \\
&= 30n - 1
\end{aligned}$$

4. Find the upper bound on the number of comparisons for a block size of 7.

Solution: when $b = 7$,

$$\begin{aligned}
T(n) &\leq \left(\frac{2b(1+b)}{b-3}\right)n - 1 \\
&= 28n - 1
\end{aligned}$$

Problem 2. In our analysis of the select algorithm for a block size of 5, we compared the *median of the medians* with every other element in the input array to partition it. However, that led to more comparisons than we should have done. We could reduce it since we know that at least $3n/10$ elements are less than or equal to the *median of the medians* and similar number of elements is at least greater than or equal to the *median of the medians*. Obtain a new upper bound on the worst-case number of comparisons using this piece of information. What is the value of the constant?

Solution:

Number of comparisons to sorting each group of 5 elements: $\frac{5(5-1)}{2} \cdot \frac{n}{5} = 2n$
Finding the median of medians: $T\left(\frac{n}{5}\right)$
Since we know $3n/10$ elements are smaller than the median and $3n/10$ are larger than the median. We don't need to compare with those elements to partion the input. Therefore, the number of elements left to compare with would be: $n - \frac{3n}{2} - \frac{3n}{2} = \frac{2n}{5}$
Number of comparisons to partition: $\frac{2n}{5} - 1$
Recurrence over the proper partition: $T\left(\frac{7n}{10}\right)$
Therefore, the upper bound on runtime is,

$$\begin{aligned} T(n) &\leq 2n + T\left(\frac{n}{5}\right) + \frac{2n}{5} - 1 + T\left(\frac{7n}{10}\right) \\ &= T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + \frac{12n}{5} - 1 \end{aligned}$$

Solving the recurrence using constructive induction.
Guess: $T(n) \leq an$

$$\begin{aligned} T(n) &\leq \frac{an}{5} + \frac{7an}{10} + \frac{12n}{5} - 1 \\ &= \frac{9an}{10} + \frac{12n}{5} - 1 \\ &= \left(\frac{9a}{10} + \frac{12}{5}\right)n - 1 \\ &\leq an \end{aligned}$$

Thus,

$$\begin{aligned} \frac{9a}{10} + \frac{12}{5} &= a \\ \frac{12}{5} &= a - \frac{9a}{10} \\ a &= 24 \end{aligned}$$

Therefore,

$$T(n) \le \left(\frac{9a}{10} + \frac{12}{5}\right)n - 1$$

$$\frac{9a}{10} + \frac{12}{5} = a$$

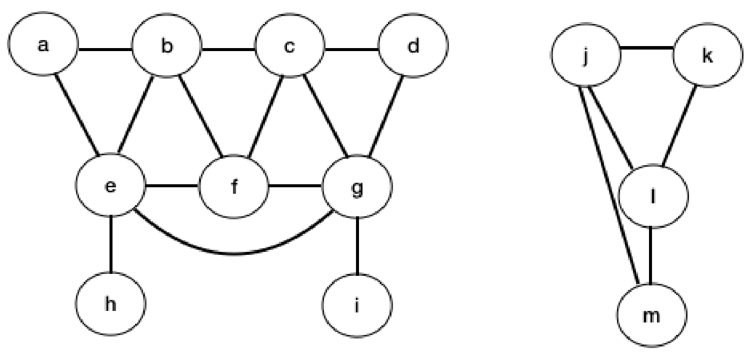
$$\frac{12}{5} = a - \frac{9a}{10}$$

$$a = 24$$

Subsituting, $a = 24$

$$\begin{aligned} T(n) &\le \left(\frac{9a}{10} + \frac{12}{5}\right)n - 1 \\ &= \left(\frac{9}{10} \cdot 24 + \frac{12}{5}\right)n - 1 \\ &= 24n - 1 \end{aligned}$$

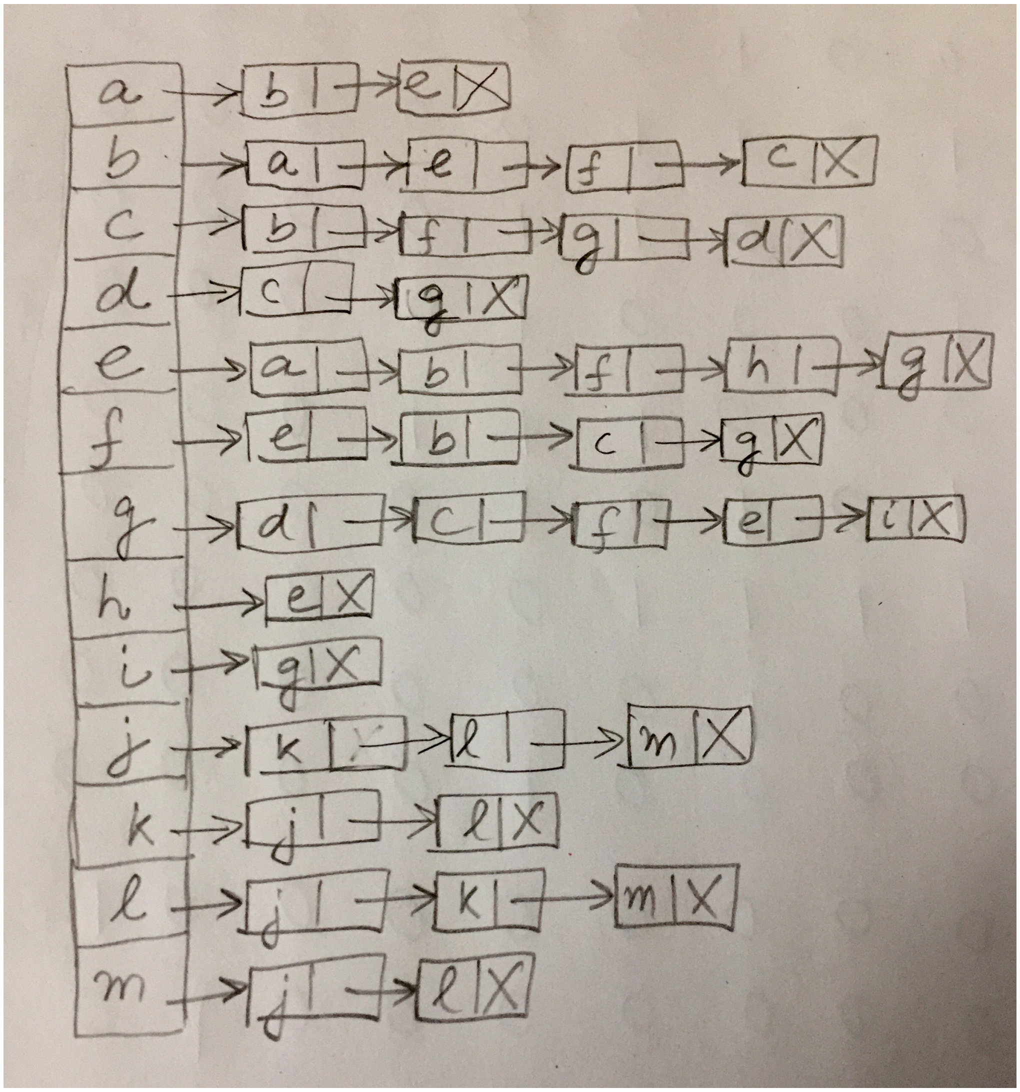
Problem 3. Consider the following graph, $G = (V, E)$, with $|V|$ vertices and $|E|$ edges.



If we represent graphs in a matrix form, it takes constant time, $O(1)$, to determine whether there is an edge between two vertices, however, the space complexity is $O(|V|^2)$. Representing a graph using an adjacency list has a space complexity of $O(|V|+|E|)$, however, determining an edge, $u, v \in |E|$ can have a runtime complexity of $O(deg(u))$. Answer the following questions:

1. Show an adjacency list representation of the graph.

Solution:



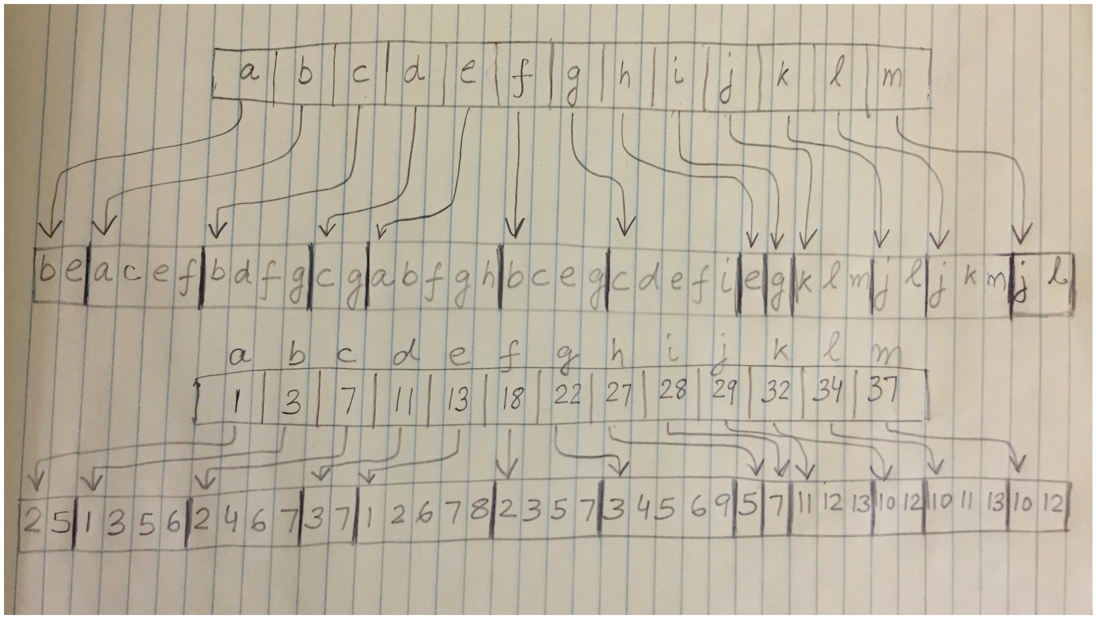
2. Show an adjacency matrix representation of the graph.

Solution:

	a	b	c	d	e	f	g	h	i	j	k	l	m
a	0	1	0	0	1	0	0	0	0	0	0	0	0
b	1	0	1	0	1	1	0	0	0	0	0	0	0
c	0	1	0	1	0	1	1	0	0	0	0	0	0
d	0	0	1	0	0	0	1	0	0	0	0	0	0
e	1	1	0	0	0	1	1	1	0	0	0	0	0
f	0	1	1	0	1	0	1	0	0	0	0	0	0
g	0	0	1	1	1	1	0	0	0	0	0	0	0
h	0	0	0	0	1	0	0	0	0	0	0	0	0
i	0	0	0	0	0	0	1	0	0	0	0	0	0
j	0	0	0	0	0	0	0	0	0	0	1	1	1
k	0	0	0	0	0	0	0	0	0	1	0	1	0
l	0	0	0	0	0	0	0	0	0	1	1	0	1
m	0	0	0	0	0	0	0	0	0	1	0	1	0

3. Suppose we want to take advantage of space complexity of adjacency lists and still have a faster look up of an edge, give a graph representation so that we can determine whether vertex v is adjacent to vertex u in $O(lg(deg(u)))$.

Solution:
We can use something like an adjacency array. It will use a single array to store all edges, with the edges incident to each vertex in a contiguous interval. A separate array stores the index of the first edge incident to each vertex. The intervals for each vertex are in the sorted order. This would help us check whether two vertices u and v are adjacent to each other in $O(lg(deg(u)))$ runtime. This respresentation is shown below:


a b c d e f g h i j k l m
b e a c e f b d f g c g a b f g h b c e g c d e f i e g k l m j l j k m j l
a b c d e f g h i j k l m
1 3 7 11 13 18 22 27 28 29 32 34 37
2 5 1 3 5 6 2 4 6 7 3 7 1 2 6 7 8 2 3 5 7 3 4 5 6 9 5 7 11 12 13 10 12 10 11 13 10 12