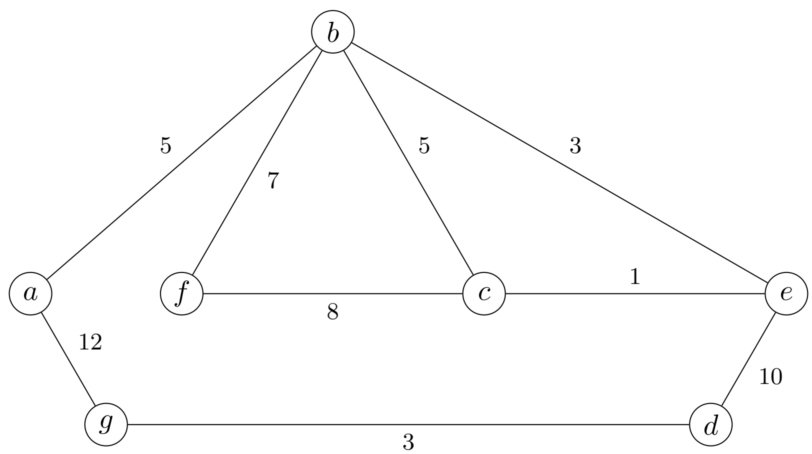
3 Standard 3 – Dijkstra's Algorithm

3.1 Problem2

Problem 2. Consider the undirected weighted graph $G(V, E, w)$ pictured below. Work through Dijkstra's algorithm on the following graph, using the source vertex a. **Note:** In order to get full credits consider the following:

- Clearly include the contents of the priority queue, the distance from a and the parent of each vertex at each iteration.
- If you use a table to store the distances, clearly label the keys according to the vertex names rather than numeric indices (i.e., `dist['B']` is more descriptive than `dist['1']`).
- You do **not** need to draw the graph at each iteration, though you are welcome to do so. [This may be helpful scratch work, which you do not need to include.]
- Finally represent the shortest path graph.



Answer. We proceed as follows.

(a) Our source node is a. We begin by setting the found distances for each node other than a to ∞. We also set a as processed.

(b) We begin by examining a's neighbors, updating their found distances and placing into the priority queue, ordering by found distance. So:

$$Q = [(b, 5), (g, 12)].$$

$nodes$	a	b	c	d	e	f	g
$distances$	∞	5	∞	∞	∞	∞	12
$parent$	NA	a	NA	NA	NA	NA	a

(c) We begin by examining b's neighbors, updating their found distances and placing into the priority queue, ordering by found distance. So:

$$Q = [(e, 8), (c, 10), (g, 12), (f, 12)].$$

$nodes$	a	b	c	d	e	f	g
$distances$	∞	5	10	∞	8	12	12
$parent$	NA	a	b	NA	b	b	a

(d) We begin by examining e's neighbors, updating their found distances and placing into the priority queue, ordering by found distance. So:

$$Q = [(c, 9), (g, 12), (f, 12), (d, 18)].$$

$nodes$	a	b	c	d	e	f	g
$distances$	∞	5	9	18	8	12	12
$parent$	NA	a	e	e	b	b	a

(e) We begin by examining c's neighbors, updating their found distances and placing into the priority queue, ordering by found distance. So:

$$Q = [(g, 12), (f, 12), (d, 18)].$$

$nodes$	a	b	c	d	e	f	g
$distances$	∞	5	9	18	8	12	12
$parent$	NA	a	e	e	b	b	a

(f) We begin by examining g's neighbors, updating their found distances and placing into the priority queue, ordering by found distance. So:

$$Q = [(f, 12), (d, 15)].$$

$nodes$	a	b	c	d	e	f	g
$distances$	∞	5	9	15	8	12	12
$parent$	NA	a	e	g	b	b	a

(g) We begin by examining f's neighbors, updating their found distances and placing into the priority queue, ordering by found distance. So:

$$Q = [(d, 15)].$$

$nodes$	a	b	c	d	e	f	g
$distances$	∞	5	9	15	8	12	12
$parent$	NA	a	e	g	b	b	a

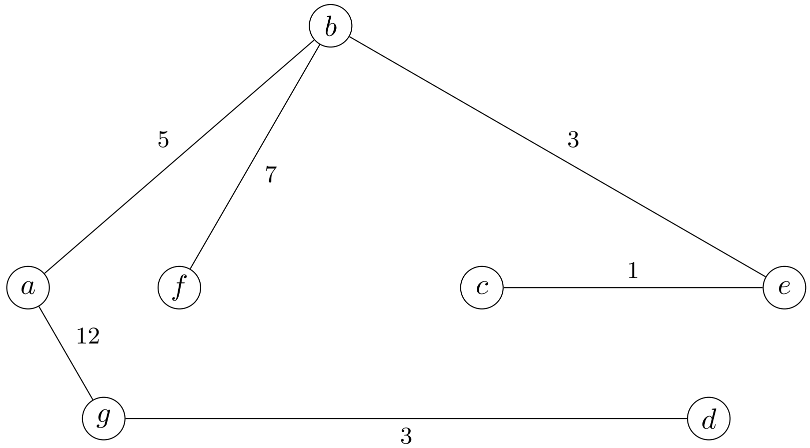
(h) We begin by examining d's neighbors, updating their found distances and placing into the priority queue, ordering by found distance. So:

$$Q = [].$$

$nodes$	a	b	c	d	e	f	g
$distances$	∞	5	9	15	8	12	12
$parent$	NA	a	e	g	b	b	a

(i) Finally, we see that the Priority Queue is empty, which is when we exit the algorithm.

The final Dijkstra's shorted path graph:



$\square$

15 Standard 15- Analyzing Code II: (Dependent nested loops)

Problem 15. Analyze the *worst-case* runtime of the following algorithms. Clearly derive the runtime complexity function $T(n)$ for this algorithm, and then find a tight asymptotic bound for $T(n)$ (that is, find a function $f(n)$ such that $T(n) = \Theta(f(n))$). Avoid heuristic arguments from 2270/2824 such as multiplying the complexities of nested loops.

Algorithm 2 Nested Algorithm 2

procedure Boo1(Integer n)
    for $i \leftarrow 1; i \leq n; i \leftarrow i + 1$ **do**
        for $j \leftarrow i; j \leq n; j \leftarrow j + 1$ **do**
            print “Hi”

Answer. □

2.5 Problem 5

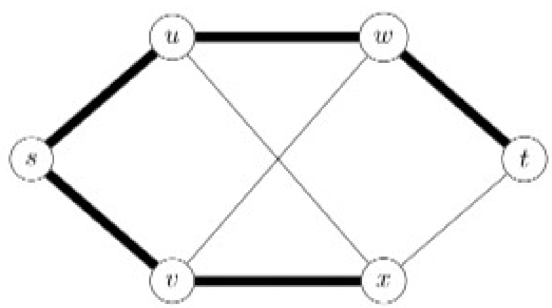
Problem 5. Give an example of a simple, undirected, and unweighted graph $G(V, E)$ that has a single source shortest path tree which a **breadth-first traversal** will not return for any ordering of its vertices. Your answer must

(a) Provide a drawing of the graph G. [**Note:** We have provided TikZ code below if you wish to use LaTeX to draw the graph. Alternatively, you may hand-draw G and embed it as an image below, provided that (i) your drawing is legible and (ii) we do not have to rotate our screens to grade your work.]

(b) Specify the single source shortest path tree $T = (V, E_T)$ by specifying E_T and also specifying the root $s \in V$. [**Note:** You may again hand-draw this tree. If you wish, you may clearly mark the edges of T on your drawing of G. Please make it easy on the graders to identify the edges of T.]

(c) Include a clear explanation of why the breadth-first search algorithm we discussed in class will never produce T for any orderings of the vertices.

Answer.



We go with the example graph with some modifications to make wide edges. So $V = \{s, t, u, v, w, x\}$ and:

$$E = \{\{s, u\}, \{s, v\}, \{u, w\}, \{u, x\}, \{v, w\}, \{v, x\}, \{w, t\}, \{x, t\}\}.$$

We have chosen $E_T = \{\{s, u\}, \{s, v\}, \{u, w\}, \{v, x\}, \{w, t\}\}$ indicated with bold edges. We assert that the given T is a spanning tree from s as all the vertices can be reached by a path originating from s. This is because T is connected and includes all of the vertices of G. Moreover, T is a single-source shortest-path spanning tree with source s: it is clear that there are no shorter paths to u and v since they are already at distance 1 from s in T. Likewise for w and x, they are not adjacent to s and so the shortest paths must have length at least 2. Finally, we see that there is no path of length 2 from s to t.

E_T can never be given by BFS starting from s. There are two cases.

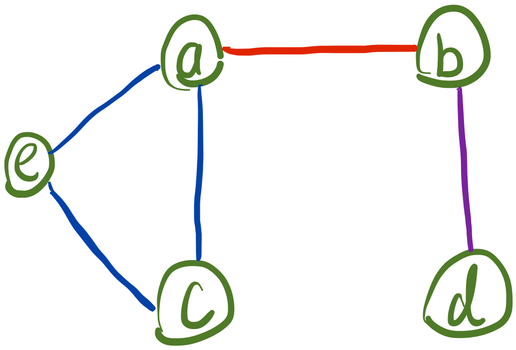
- **Case 1:** If $u < v$ in the ordering we are given, then we will first explore the vertices adjacent to u, which means that we would reach x from u, and T would need to include the edge $\{u, x\}$.
- **Case 2:** If $v < u$ in the ordering we are given, then we will first explore the vertices adjacent to v, which means that we would reach w from v, and T would need to include the edge $\{v, w\}$.

$\square$

Let’s think about a class schedule problem. Assume there are 5 classes, which are a, b, c, d, e and 3 students, X, Y and Z. The course registry information is shown below.

Student X takes class a, b
Student Y takes class a, c, e
Student Z takes class b, d

Assume all the classes take 1 hour, then how many separate periods do we need to schedule courses?



Assume that we don’t color the same for vertices, who are neighbors to each other. Then, . . .

Could we color them using 5 colors?
Could we color them using 4 colors?
Could we color them using 3 colors?
Could we color them using 2 colors?
Could we color them using 1 color?

9 Standard 9: Huffman Coding

Problem 9. Consider a message that uses the following letters with the specified frequencies:

a	b	c	d	e
1	2	3	4	5

You should

- Specify the intermediate trees/forest constructed at each stage of Huffman's algorithm. You may specify your trees either by drawing them by hand & including the images, using a LaTeX package such as TikZ, or with nested parentheses.*
- Specify the encoding of each of the five letters.

Answer. The forest after the first step of the algorithm has (a, b) with frequency $1 + 2 = 3$, and the rest are the original singletons.

The forest after the second step of the algorithm is $((a, b), c)$ with frequency $3 + 3 = 6$, with d and e still singletons.

The forest after the third step of the algorithm consists of $((a, b), c)$ and (d, e).

The forest after the fourth step of the algorithm is $(((a, b), c), (d, e))$.

The final encoding of the letters is as follows:

a	b	c	d	e
000	001	01	10	11

(Note: depending on your placement of the vertices in the tree and your conventions about encoding, you may have different bit strings than this. But you strings should have the same length as those given here.)

□

```
HUFFMAN(C)
n = |C|
Q = C
for i = 1 to n - 1
    allocate a new node z
    z.left = x = EXTRACT-MIN(Q)
    z.right = y = EXTRACT-MIN(Q)
    z.freq = x.freq + y.freq
    INSERT(Q, z)
return EXTRACT-MIN(Q)        // return the root of the tree
```