1. **Determine the number of character comparisons made by the brute-force algorithm in searching for the pattern GANDHI in the text THERE_IS_MORE_TO_LIFE_THAN_INCREASING_ITS_SPEED . Assume that the length of the text, it is 47 characters long, is known before the search starts.**

   The first letter, **G** , in the given *pattern* GANDHI does not occur in the *text* until the 36^{th} index. Until **G** is matched, no second comparison will be made; subsequently we know there will be a single comparison between **G** and the current index of the *text* between indices 0 and 35. When GHANDI reaches the 36^{th} index, we know **G** will match the *G* in INGREASING_, and so a second comparison between **A** and _ will be made, but fail. So for index 36, there will be two comparisons. From here the index compared against **G** will be incremented to 37, but because no more G's exist in the *text* this will continue until the brute-force algorithm terminates at index $m - n = 47 - 6 = 41$. In total we have 1 comparison for indices 0 to 35, 2 comparisons for index 36, and one comparison for indices 37 to 41.

$$\begin{aligned} Comparisons &= \sum_{i=0}^{35} 1 + \sum_{i=36}^{36} 2 + \sum_{i=37}^{41} 1 \\ &= (35 - 0 + 1) + (2) + (41 - 37 + 1) \\ &= (36) + (2) + (5) \\ &= 43 \end{aligned} \tag{1}$$

   To save some space, and because it is incredibly difficult to format correctly, the below illustrates the brute-force approach to this pattern matching.

   THERE_IS_MORE_TO_LIFE_THAN_INCREASING_ITS_SPEED

   GANDHI

   GANDHI

   GANDHI

   . . .

   GANDHI

   GANDHI

   GANDHI

   . . .

2. **Find the convex hulls of the following sets and identify their extreme points (if they have any):**

   a. **A line segment**

      The convex hull is the line segment itself since the set of points defining a line segment is a convex set. If the line segment is closed from (x_1, y_1) to (x_2, y_2) then (x_1, y_1) and (x_2, y_2) are its *extreme points*.

   b. **A square**

Since the set of points defining the square is convex, then the convex hull is the square itself (the same set of points). The four vertices which define the squares corners, $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4)$, are the *extreme points* of the convex set.

c. The boundary of a square

Since the boundary of a square dos not include the enclosed area, the set of points defining the boundary is not convex. The convex hull is then the set of points that define the square formed by the boundary (including the points of the boundary and within in). The four vertices which define the square's corners (x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4) - then serve as the *extreme points*.

d. A straight line

By definition a straight line is convex, so the convex hull is the straight line itself (the set of points that define it, that is). However, since the line is not finite there are no *extreme points*.

3. ThreeJugs - Simeon Denis Poisson (1781 to 1840), a famous French mathematician and physicist, is said to have become interested in mathematics after encountering some version of the following old puzzle. Given an 8-pint jug full of water and two empty jugs of 5-pint and 3-pint capacity, get exactly 4 pints of water in one of the jugs by completely filling up and/or emptying jugs into others. Solve this puzzle by using breadth-first search.

To begin our breadth first search (BFS), we must begin by constructing a graph under the rules in which all reachable vertices from any given vertex in the graph must follow the restrictions of the puzzle. That is, if we are on a node with the 8-pint jug filled (i.e. $8_8, 0_3, 0_5$), our only allowed transitions are to vertices where the 3-pint or the 5-pint is filled using water from the full 8-pint jug (i.e. $5_8, 3_3, 0_5$ or $3_8, 0_3, 5_5$). Each nodes label will represent the filled volume of each container where the subscript denotes the containers maximum volume and the associated value denotes the volume of fluid currently in that container. In addition, we will add the restriction to terminate the BFS once a node containing the value 4 is encountered. Note that previously visited vertices will not be listed again in the $reachable\ vertex$ column (this includes vertices currently in the queue). To begin we select $8_8, 0_3, 0_5$ as our starting point:

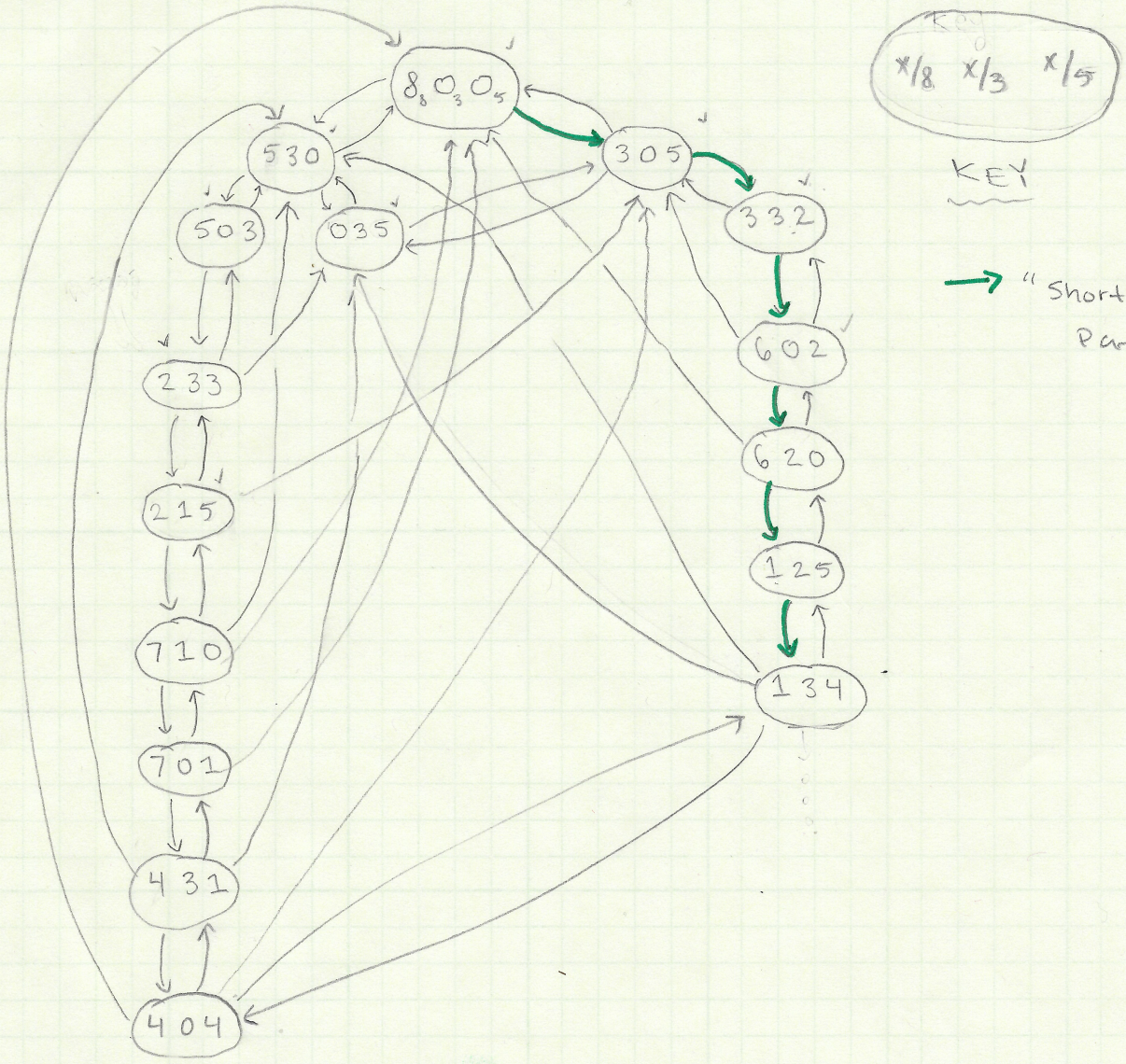
$step$	$reachable + unvisited$	$queue\ (front\ signifies\ current\ vertex)$
1	$(5_8, 3_3, 0_5), (3_8, 0_3, 5_5)$	$[(8_8, 0_3, 0_5)]$
2	$(0_8, 3_3, 5_5), (5_8, 0_3, 3_5)$	$[5_8, 3_3, 0_5), (3_8, 0_3, 5_5)]$
3	$(3_8, 3_3, 2_5)$	$[(3_8, 0_3, 5_5), (0_8, 3_3, 5_5), (5_8, 0_3, 3_5)]$
4	$None$	$[(0_8, 3_3, 5_5), (5_8, 0_3, 3_5), (3_8, 3_3, 2_5)]$
5	$(2_8, 3_3, 3_5)$	$[(5_8, 0_3, 3_5), (3_8, 3_3, 2_5)]$
6	$(6_8, 0_3, 2_5)$	$[(3_8, 3_3, 2_5), (2_8, 3_3, 3_5)]$
7	$(2_8, 1_3, 5_5)$	$[(2_8, 3_3, 3_5), (6_8, 0_3, 2_5)]$
8	$(6_8, 2_3, 0_5)$	$[(6_8, 0_3, 2_5), (2_8, 1_3, 5_5)]$
9	$(7_8, 1_3, 0_5)$	$[(2_8, 1_3, 5_5), (6_8, 2_3, 0_5)]$
10	$(1_8, 2_3, 5_5)$	$[(6_8, 2_3, 0_5), (7_8, 1_3, 0_5)]$
11	$(7_8, 0_3, 1_5)$	$[(7_8, 1_3, 0_5), (1_8, 2_3, 5_5)]$
12	$(1_8, 3_3, 4_5)$	$[(1_8, 2_3, 5_5), (7_8, 0_3, 1_5)]$
	$Halt$	$[(7_8, 0_3, 1_5), (1_8, 3_3, 4_5)]$

Using our BFS to reconstruct a path from $(8_8, 0_3, 0_5)$ to $(1_8, 3_3, 4_5)$ we have (with annota-

tions):

step	*comment*
$(8_8, 0_3, 0_5) \rightarrow (3_8, 0_3, 5_5)$	Fill up the 5-pint jug using the 8-pint
$(3_8, 0_3, 5_5) \rightarrow (3_8, 3_3, 2_5)$	Fill up the 3-pint jug using the 5-liter (leaving 2-pints leftover)
$(3_8, 3_3, 2_5) \rightarrow (6_8, 0_3, 2_5)$	Empty the 3-pint jug into the 8-pint jug
$(6_8, 0_3, 2_5) \rightarrow (6_8, 2_3, 0_5)$	Empty the 5-pint (the 2-pint leftover) into the 3-pint jug
$(6_8, 2_3, 0_5) \rightarrow (1_8, 2_3, 5_5)$	Fill up the 5-pint jug using the water from the 8-pint jug
$(1_8, 2_3, 5_5) \rightarrow (1_8, 3_3, 4_5)$	Fill up the 3-pint jug using the water from the 5-pint jug.

See attached page for a rough *sketch* of the directional graph representing the puzzle and possible solutions. Although the shortest path is outlined above, there are alternative routes.


8 0 5
530
305
503
035
332
602
233
620
215
125
710
134
701
431
404
KEY
x/8 x/3 x/5
"Shortest Path"