[redacted]

שאלה 1:

הרעיון הכללי יהיה למצוא עבור כל אחת מהערים את המסלול הקצר ביותר אליהן במספר כלשהו של ימים.

נתאר את האלגוריתם :

האלגוריתם יעבור יום יום מ1 ועד d , בכל יום נעבור על כל הערים ונבדוק עבור כל אחת מהערים את מחיר המסלול שיווצר כשנתקדם לערים הסמוכות אליה ונמצא את המעבר הזול ביותר.

בכל פעם נשמור את התוצאה ונתקדם בימים כך שכשנגיע ליום האחרון נוכל לדעת מה היו הסכומים של המסלולים בימים 1 עד d מליסבון לאתונה , כלומר אחרי שנסיים את כל התהליך נעבור על כל המסלולים שהתחלנו בליסבון והגענו לאתונה ביום כלשהו בטווח הנתון ונבדוק האם המחיר ששילמנו היה לכל היותר b אם כן נחזיר אמת ואחרת שקר.

נכונות של האלגוריתם :

בכל שלב בלולאה הראשית נקבל מחיר של מסלול כלשהו שאורכו היום הנוכחי באיטרציה בהינתן הימים הקודמים במסלול כלומר בכל יום לכל עיר נתונה נקבל מחיר כלשהו להגיע אליה במספר הימים הנתון ולכן המחיר המינימלי להגיע אליה קטן או שווה מהמחיר שהצלחנו לחשב ולכן אם כאשר הגענו לאתונה ביום כלשהו המחיר היה קטן שווה מb נקבל שאכן המחיר המינימלי קטן או שווה לb .

מבחינת סיבוכיות זמן יש לנו לולאה מקוננת כאשר הלולאה החיצונית רצה d פעמים והלולאה הפנימית רצה n פעמים .

בכל ריצה של הלולאה הפנימית נעבור על כל הערים הסמוכות ונבצע חישובים כלומר o(1) זמן ולכן בסך הכל חלק זה יקח o(nd) , בסוף נרצה לבדוק את תוצאת האלגוריתם ונעבור על כל הימים כלומר נבצע o(d) פעולות.

לכן בסך הכל האלגוריתם ייקח o(nd) זמן.

Q2

1. We will prove that G_b, G_f are DAG's and topological sorts accordingly.

   For G_b :
   It follows that $E_b = \{(v_i, v_j) | i > j\} \implies E$ is the set of all arcs that **leave** $\forall i \in [n] : v_i$
   Let's assume by contradiction that G_b has a cycle. Meaning that the following path $P = \langle v_1, v_2, \ldots, v_k \rangle$ and $v_1 = v_k$ (definition of a cycle) for $k \geq 1$, we can see that $(v_k, v_2) \in E_b$ and (v_{k-1}, v_k) and thus we get $k - 1 > k$ in contradiction to how we built G_b.

The same goes for G_f just all the arcs that come in $\forall i \in [n] : v_i$ and $E_f = \{(v_i, v_j) | j > i\}$

$\implies G_b, G_f$ are DAG's

G_b, G_f are both topological sorts accordingly by definition of the arcs in the sets since for G_b if we go from $\forall i > j$ on $v_n, \ldots, v_1$ then we know that the arc (v_i, v_j) and thus node v_i will be shown before node v_j since we are talking about arcs going from v_n to v_{n-1} etc...

similar for G_f

2. Let us choose an arbitrary node $v \in V$, we get from the no path property that for some node that is not accessible from s that $v.d = \delta(s, v) = \infty$ after calling $relax$ any amount of times since the node isn't accessible from s.

   Otherwise, from the upper bound theory the following is always true when we call $relax$ - $v.d \geq \delta(s, v)$

   Now we just have to show that $v.d \leq \delta(s, v)$:
   from part A we know that for G_b that when we can go through the topological order $v_n, \ldots, v_1$. Meaning $\forall i > j : v_j$ will be found/shown before v_i and thus for the path $P = \langle s = v_{k_1}, \ldots, v_{k_m} \rangle$.
   we will prove in induction that we called after $m = \lceil \frac{n}{2} \rceil$ iterations such that we did the following calls to $Relax$ - $Relax(v_{k_1}, v_{k_2}), \ldots, Relax(v_{k_{m-1}}, v_{k_m})$
   Base Case where $m = 1$:
   where k_1 is the first index and is smaller than k_2 thus we call $Relax(v_{k_1}, v_{k_2})$ on the first iteration of $(v_1, v_2, \ldots, v_n)$
   Assumption:
   That after the m'th iteration we will have made the sequence of calls to- $Relax(v_{k_1}, v_{k_2}), \ldots, Relax(v_{k_{m-1}}, v_{k_m})$
   Step:
   We will split into cases:
   if $k_m > k_{m+1}$:
   we will do the $Relax$ call of these two after k_{m-1}, k_m initially and thus after the m'th iteration we then would call $Relax(k_m, k_{m+1})$
   else:
   if we didn't call Relax on the m'th iteration to k_{m+1}, k_m then $k_m < k_{m+1}$ and we know that in the $m + 1$ iteration that we will call Relax on these two and since we know that the vertices are topologically sorted then we call $Relax(k_m, k_{m+1})$
   Therefore, by induction we proved that after the m'th iteration that we did the required $Relax$ calls that we were looking for. $\implies$ after $\lceil \frac{n}{2} \rceil$ we get that $v.d = \delta(s, v)$ if v is reachable from s for every $v \in V$ ■
3. The runtime doesn't change since the outer loop of the function will run on all the nodes since G_b, G_f are defined on all the nodes in V and furthermore $|E| = |E_b| + |E_f| = m$ so we"ll go through all the arcs similarly to the original Bellman-Ford but adjusted to our algorithm. (if we stop the $\lceil \frac{n}{2} \rceil$ then we"ll go through each time $|E_b|$ which is approximately m when referring to runtime, thus we"ll get $\lceil \frac{n}{2} \rceil \cdot m$ which is a runtime of $\mathcal{O}(m \cdot n)$)
   Therefore, we"ll get a runtime of $\mathcal{O}(n \cdot m)$

שאלה 3 :

ניקח את קבוצת הצמתים ונוסיף לה צומת נוספת t כך שנוסיף לקבוצת הקשתות המקורית קשת לכל אחת מן הצמתים המקוריות קשת לצומת החדשה שמשקלה יהיה 0 .

כעת נריץ את אלגוריתם בלמן- פורד על קבוצת הצמתים הקשתות והמשקלים החדשים שבנינו על צומת המקור t , נעשה זאת בסיבוכיות זמן של nm.

כעת קיבלנו עץ מסלולים קצרים שמתאר את המרחק של הצמתים מצומת t .

כעת נרצה ליצור תת גרף שמכיל את כל הצמתים u, v כך שמתקיים : u.d + w(u,v) = v.d

ונראה שאם גרף זה לא ריק משמע שיש מעגל שמשקלו 0.

הוכחת נכונות:

נראה שאם יש מעגל במשקל 0 אזי מתקיים : u.d + w(u,v) = v.d ולהפך.

כיוון ראשון: נניח שיש מעגל במשקל 0 ששייכים אליו הצמתים u,v ונניח בשלילה שמתקיים:

u.d + w(u,v) > v.d

(אי השיוויון ההפוך לא אפשרי מאחר ומתקיים u.d + w(u,v) הוא מסלול מt לv ולא ייתכן שהוא קטן ממש מהמרחק מt לv לפי הגדרת מרחק)

מתקיים שאורך המסלול מ v לu שנסמנו p על המעגל c מקיים w(p) = w(c) – w(u,v)

ובגלל שמשקל המעגל הוא 0 נקבל w(p) =– w(u,v)

כעת נבחר את המסלול מ t לv למסלול מv ל u ונקבל שאורך מסלול זה שווה ל- v.d- w(u,v)

ולפי הנחת השלילה מתקיים כי אורך מסלול זה שמגיע מt ל u קטן ממש מהמרחק של u וזו סתירה.

כיוון שני: יהיה מעגל c = (v0, v1, v2,..., v0) שמקיים לכל j,i: vi.d + w(vi,vj) = vj.d

ולכן מתקיים:

$$\delta(t, v0) = \delta(t, Vk-1) + w(Vk-1, V0) = \delta(t, Vk-2) + w(Vk-1, V0) + w(Vk-2, Vk-1)$$

נוכל לבצע פעולה זאת הלאה והלאה עד שלבסוף נקבל :

$$\delta(t, v0) = \delta(t, v0) + w(c)$$

ולכן נקבל w(c) = 0

Q5

The statement is true.
Let's choose an arbitrary path $P = \langle s, v_1, \ldots, v_i, \ldots, v_k, v_{k+1}, \ldots, v \rangle$ as the shortest path in G. Let's assume by contradiction such that the order of calls to the function $relax$ on our path P isn't according to order of arcs in the path.

let's denote v_i as the node where until we get to it, we called in the correct order the series of calls to the function $relax$ on it's sub-path $P_1 = \langle s, \ldots, v_i \rangle$. Therefore, we get that P_1 is the shortest sub-path from s to v_i and according to the $shortestpathrelaxation$ theorem we get after the calls of relax on $(s, v_1), \ldots, (v_{i-1}, v_i)$ on the sub-path and thus $v_i.d = \delta(s, v_i)$.
After we call Relaxes and move on to the next arc- (v_k, v_{k+1}) which also get's called before (v_i, v_{i+1}).
At time k is when we remove v_k from Q and therefore, $v_i.d = \delta(s, v_i)$ at that time. Furthermore, $v_k.d = \delta(s, v_k)$ and since we choose the minimal node in the Queue Q we get in time k such that $\delta(s, v_k) = v_k.d \leq v_i.d = \delta(s, v_i)$. In contradiction since the weights of all the arcs are non zero and negative and as we found v_i before v_k when traversing our path, we get that $\delta(s, v_i) < \delta(s, v_k)$ ■