Question 1

In class we talked about the problem of finding the number of inversions in an array. In this problem the input is a sequence of $n$ distinct numbers $a_1, \ldots, a_n$ and we defined an inversion to be a pair $i < j$ such that $a_i > a_j$. Suppose instead that we are given an $n \times n$ matrix $a_{1,1}, \ldots, a_{n,n}$ of $n^2$ numbers. We say that the pair of points $p_1 = (i_1, j_1)$ and $p_2 = (i_2, j_2)$ are inverted if $i_1 < i_2, j_1 < j_2$ AND $a_{i_1,j_1} > a_{i_2,j_2}$. Give an $O(n^2 \log^3 n)$ time algorithm to count the number of inversions in the $n \times n$ matrix. You should prove that your algorithm is correct and analyze its running time.

Question 2

Analyze the asymptotic complexity of each of the following functions (e.g., if $T(n) = 2T(n/2) + n$ then $T(n) \in \Theta(n \log n)$). You should aim to provide tight upper and lower bounds on $T(n)$ and you should prove your answers are correct e.g., using induction or unrolling. In each case you should assume that $T(n) = 1$ for $n < 3$.

- $T(n) = 3 \times T(n/2) + n^2$ for $n \geq 3$.
- $T(n) = T(a_1n) + T(a_2n) + \ldots + T(a_kn) + n$ for $n \geq 3$ where $a_1 + a_2 + \ldots + a_k < 1$ and $a_i \geq 0$ for each $i \leq k$.
- $T(n) = 7 \times T(n/5) + 2 \times T(3n/5) + n^2$ for $n \geq 3$.
- $T(n) = T(n-1) + T(n-5)$ for $n \geq 3$.
- $T(n) = T(n-n^{1/2}) + n^{1/4}$ for $n \geq 3$.

Question 3

Consider a store that sells $n$ items $1, \ldots, n$ where each item $i$ has a volume $v_i \in \{1, \ldots, 100\}$, mass $m_i \in \{1, \ldots, 100\}$ and a a price $p_i \in \{1, \ldots, 100\}$. Furthermore, you also have a personal worth $w_i$ associated with each item. The total worth of a bundle $S \subseteq \{1, \ldots, n\}$ of items is $w(S) = \sum_{i \in S} w_i$. You can only fit a bundle $S$ of goods in your car if the total volume $v(S) = \sum_{i \in S} v_i \leq V_{\text{max}}$ and total mass $m(S) = \sum_{i \in S} m_i \leq M_{\text{max}}$ are suitably small, and you can only afford the bundle if $p(S) = \sum_{i \in S} p_i \leq P_{\text{max}}$. Develop a polynomial time pre-processing algorithm to help find a feasible bundle $S$ of maximum worth. After the pre-processing algorithm completes it should be possible to find the optimal feasible bundle $S$ in linear time $O(n)$. The inputs $V_{\text{max}}, P_{\text{max}}$ and $M_{\text{max}}$ are not revealed until after the pre-processing algorithm completes. For full credit your pre-processing algorithm should use
time $O(n^4)$ and space $O(n^3)$. )

**Question 4**

Captain Crazy has a new algorithm for the deciding whether a graph $G = (V, E)$ has a Hamiltonian Path going from $u$ to $v$. A simple $u$ to $v$ path is called Hamiltonian if it starts at $u$, visits each vertex exactly once and ends at $v$. Let $\Gamma(u)$ denote the set of vertices adjacent to $u$. Below is the algorithm Crazy uses.

1. `BoolHasHamPath(u, v, G)`
2. If $u == v$, is the only node in $G$ return 1
3. If $\text{deg}(u) == 0$ return 0
4. For each $w \in \Gamma(u)$ do:
5. If `HasHamPath(w, v, G - u)`, return 1
6. return 0.

**Part 1:** (5 points) Prove that Captain Crazy’s algorithm is correct.

**Part 2:** (5 points) Find a graph $G$ with $n$ nodes along with a pair of vertices $u, v$ such that Crazy’s algorithm runs in time $\Omega((n - 2)!)$.

**Part 3:** (15 points) Suppose you are given a graph in which every vertex has degree at most 3. Show that in this case Crazy’s algorithm decides `HasHamPath` correctly in time $O(2^{3n/4})$. 