17.3.3

We would like to show that $\mathbf{DP} \subseteq \mathbf{PP}$.

We know that $\mathbf{NP} \subseteq \mathbf{PP}$ (page 257 in the book). Furthermore, since **PP** is closed under complement, $\mathbf{coNP} \subseteq \mathbf{PP}$.

Take any language $L \in \mathbf{DP}$. Thus there exist languages $L_1 \in \mathbf{NP}$ and $L_2 \in \mathbf{coNP}$ such that $L = L_1 \cap L_2$. Take M_1 be the precise non-deterministic Turing machine deciding L_1 by using the **NP**-acceptance condition. By the inclusions above, there exists a precise non-deterministic polynomial time Turing machine M_2 deciding L_2 by using the the **PP**-acceptance condition. Let $p(\cdot)$ be the polynomial time bound of M_2. Now let M be the precise non-deterministic Turing machine that first executes M_1 on x. After each branch on which M_1 answers "no", M executes dummy computations of length $p(|x|)$ such that precisely $2^{p(|x|)-1}$ of them accept. On the other hand, after each branch on which M_1 answers "yes", M executes M_2 on x. Now we have three cases.

1. if $x \notin L_1$, then M never executes M_2 and therefore exactly half of the computations of M accept.
2. if $x \in L_1$ and $x \notin L_2$, at most half of the computations of M accept since after each accepting computation of M_1, at most half of the following computations of M_2 accept.
3. if $x \in L_1$ and $x \in L_2$, then after each accepting computation of M_1, more than half of the following computations of M_2 accept and since there is at least one accepting computation of M_1, more than half of the computations of M accept.

Thus M is a precise non-deterministic Turing machine that **PP**-accepts L and $\mathbf{DP} \subseteq \mathbf{PP}$.

17.3.6

Show that the following language is $\Delta_2\mathbf{P}$-complete: Given an instance of the TSP, is the optimum tour length odd? (Return false if no tour exists.)

We reduce from the following well-known $\Delta_2\mathbf{P}$-complete problem (Krentel, Journal of Computer and System Sciences, 36:490–509, 1988):

> Given a (3SAT) Boolean formula $\varphi(x_1, \ldots, x_n)$, is $x_n = 1$ in φ's

lexicographically greatest satisfying assignment? (Return false if φ is not satisfiable.)

Use the construction of Theorem 17.5 in the book except that the "emergency exists" are removed. Now all the weights in the graph are zero except the following. The weights of arcs corresponding to literals $\neg x_i$ are 2^{2n+1-i} and the arcs corresponding to literals x_i have weight 2^i except that for x_n it is $2^n + 1$ (the only odd weight).

Now if φ cannot be satisfied, then the graph does not have a tour. If φ can be satisfied, then the optimum tour length must correspond to the lexicographically greatest satisfying assignment because of the weight distribution of arcs corresponding to the literals. This is because of the following. Take a tour corresponding to a satisfying truth assignment τ. Take any other satisfying truth assignment τ' that is lexicographically greater than τ. Let x_i be the first variable that is true in τ' but false in τ (there must exist one because τ' is greater than τ). Now the costs of the tours corresponding to τ and τ' are the same when only the weights corresponding to variables $x_1, \ldots, x_{i-1}$ are calculated. Since x_i is true in τ', the tour corresponding to τ' has an advantage of $2^{2n+1-i} - 2^i$ w.r.t. the tour corresponding to τ when only x_i is considered. Take the worst case when all the variables $x_{i+1}, \ldots, x_n$ are false in τ' and true in τ. Then the maximum advantage that τ can have on variables $x_{i+1}, \ldots, x_n$ w.r.t. τ' is $\sum_{j=i+1}^{n}(2^{2n+1-j} - 2^j) = \sum_{j=i+1}^{n} 2^{2n+1-j} - \sum_{j=i+1}^{n} 2^j < 2^{2n+1-i} - 2^i$. Therefore the cost of a tour corresponding to a lexicographically greater satisfying truth assignment is lesser. And because the weight of x_n is the only odd weight, the optimum tour is odd iff the last bit in the lexicographically greatest satisfying truth assignment for φ is 1.