Problem 0.1:

(a) [DPV] 0.1(c): $f(n) = 100n + \log n, g(n) = n + (\log n)^2$

Solution.

C: $f(n) = O(g(n))$ and $g(n) = O(f(n))$
$\log n$ and $(\log n)^2$ are both $O(n)$, and can be ignored when comparing $f(n)$ and $g(n)$. Ignoring constant factors, both $f(n)$ and $g(n)$ are $\Theta(n)$, so $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

(b) [DPV] 0.1(d): $f(n) = n \log n, g(n) = 10n \log 10n$

Solution.

C: $f(n) = O(g(n))$ and $g(n) = O(f(n))$
$\log 10n = \log 10 + \log n$, and $\log 10$ can be ignored here. Ignoring constant factors, both $f(n)$ and $g(n)$ are $\Theta(n \log n)$, so $f(n) = O(g(n))$ and $g(n) = O(f(n))$.

(c) [DPV] 0.1(k): $f(n) = \sqrt{n}, g(n) = (\log n)^3$

Solution.

B: $g(n) = O(f(n))$
Taking the cube root of each $f(n)$ and $g(n)$ yields $n^{1/6}$ and $\log n$. Any positive power of n, including powers less than 1, will dominate $\log n$, so $g(n) = O(f(n))$.

(d) [DPV] 0.1(ℓ): $f(n) = \sqrt{n}, g(n) = 5^{\log_2 n}$

Solution.

A: $f(n) = O(g(n))$

$$5^{\log_2 n} = \left(2^{\log_2 5}\right)^{\log_2 n} = \left(2^{\log_2 n}\right)^{\log_2 5} = n^{\log_2 5}.$$

Since $\log_2 5 > 1$ we have that $n^{\log_5 2} > n > n^{.5}$. Therefore, $f(n) = O(g(n))$.

Problem 6.2: Hotel stops with minimum penalty

(a) Define the entries of your table in words. E.g., $T(i)$ or $T(i,j)$ is ...

Solution.

$T(i)$ is the minimum penalty obtainable for the trip from mile 0 to mile a_i, with the last stop at hotel i.

(b) State recurrence for entries of table in terms of smaller subproblems.

Solution.

Each entry $T(i)$ is computed as the minimum over all previous hotels k of the minimum penalty to get to hotel k plus the penalty from hotel k to hotel i.
$T(i) = \min_k\{T(k) + (200 - (a_i - a_k))^2 : 0 \leq k \leq i-1\}$

(c) Write pseudocode for your algorithm to solve this problem.

Solution.

```
T(0) = 0
for i = 1 to n:
    T(i) = (200 − a_i)^2
    prev(i) = NULL
    for k = 1 to i − 1:
        if T(i) > T(k) + (200 − (a_i − a_k))^2 then
            T(i) = T(k) + (200 − (a_i − a_k))^2
            prev(i) = k

# T(n) is now the minimum penalty
# Next, output the optimal sequence of hotels
i = n
output(i)
while prev(i) ≠ NULL:
    i = prev(i)
    output(i)
```

(d) Analyze the running time of your algorithm.

Solution.

Each entry in $T(i)$ takes $O(n)$ time to compute, since each is the minimum over up to $n-1$ expressions. There are n entries in T, so this algorithm takes $O(n^2)$ time.

Problem 6.3: Yuckdonald's

(a) Define the entries of your table in words. E.g., $T(i)$ or $T(i, j)$ is ...

Solution.

$T(i)$ is the maximum profit from a valid subset of locations from $m_1, m_2, ...m_i$ that includes m_i.

(b) State recurrence for entries of table in terms of smaller subproblems.

Solution.

Each entry $T(i)$ is computed as the profit from opening location i, plus the maximum profit from previous valid sets of locations at least k miles away from location i.
$T(i) = p_i + \max_j\{T(j) : j < i, m_j \leq m_i - k\}$

(c) Write pseudocode for your algorithm to solve this problem.

Solution.

```
T(0) = 0
for i = 1 to n:
    T(i) = p_i
    for j = 1 to i - 1:
        if m_j ≤ m_i - k then
            if T(i) < T(j) + p_i then
                T(i) = T(j) + p_i

return max_i L(i)
```

(d) Analyze the running time of your algorithm.

Solution.

Each entry in $T(i)$ takes $O(n)$ time to compute, since each is the maximum over up to $n - 1$ expressions. There are n entries in T, so this algorithm takes $O(n^2)$ time.