1 Problem 6.2

Let $b[i]$ be the minimum total penalty for stopping at hotel a_i, $1 \leq i \leq n$. We have

$$b[i] = \min_{1 \leq j < i} \{b[j] + (200 - (b[i] - b[j]))^2\}.$$

Also record the value j which yields $\min_{1 \leq j < i} \{b[j] + (200 - (b[i] - b[j]))^2$ and set `b[i].prev = b[j]`. Backtrack from $b[n]$ to get the sequence of hotels to stop by.

2 Problem 6.3

Let $S[i]$ be the maximum total profit we get from building some restaurants in $\{m_1, m_2, \ldots, m_i\}$. Consider 2 cases:

(1) If restaurant m_i should not be built, then $S[i] = S[i-1]$.

(2) If restaurant m_i should be built, then let c_i be the maximum index j which yields $m_i - m_j \geq k$. We then have $S[i] = p_i + S[c_i]$.

Therefore in general,

$$S_i = \max\{S[i-1], p_i + S[c_i]\}.$$

To get the sequence of restaurants, keep an array $R[n]$ such that $R[i] = 1$ if restaurant i is built in the optimal solution, and $R[i] = 0$ otherwise. When we calculate $S[i]$, if the max falls to case (1), $R[i] = 0$. Else, $R[i] = 1$. Output all the $R[i]$s that are 1.

3 Problem 6.4

Consider an array $S[n]$ where $S[i]$ = `true` if the substring $s_1 s_2 \ldots s_i$ is a valid string, and `false` otherwise. We have $S[1] =$ `dict(s[1])` and

$$\begin{aligned} S[i] = (&S[1] \;\&\&\; \texttt{dict(s[2 .. i])} \\ &||(S[2] \;\&\&\; \texttt{dict(s[3..i])} \\ &|| \ldots \\ &||(S[i-1] \;\&\&\; \texttt{dict(s[i..i])})), \end{aligned}$$

where $s[j..i]$ is $s_j s_{j+1} \ldots s_i$.

4 Problem 6.6

Let the input string be $x_1 x_2 \ldots x_n$.
Let $Z = \{a, b, c\}$ and let $T[i,j] \subset Z$ be the set of the possible values that the product $x_i x_{i+1} \ldots x_j$ can yield with all possible parenthesizations.
We see that $T[i,i] = x_i$ for all $1 \leq i \leq n$. We need to compute $T[1,n]$.
Define $A \times B$ as $\{a \cdot b | a \in A, b \in B\}$.
We note that $T[i,i+1] = T[i,i] \bigcup T[i+1,i+1]$ and $T[i,i+2] = (T[i,i] \times T[i+1,i+2]) \bigcup (T[i,i+1] \times T[i+2,i+2])$ (to put it another way, abc can be written as $(a)(bc)$ or $(ab)(c)$).
We therefore see that we already have

$$T[1,1], T[2,2], T[3,3], \ldots,$$

from which we can calculate

$$T[1,2], T[2,3], T[3,4], \ldots,$$

from which we can calculate

$$T[1,3], T[2,4], T[3,5], \ldots$$

and eventually we can expand to $T[1,n]$, which is what we need to find.
In other words,

$$T[i, i+s] = \bigcup_{i \le k < i+s} (T[i,k] \times T[k+1, i+s]).$$

The algorithm is as follows:

```
for i = 1 to n: T[i,i] = x[i].
for s = 1 to n-1:
   for i = 1 to n - s:
      T[i, i + s] = empty
      for k = 1 to i + s - 1:
         T[i, i + s] = T[i, i + s] UNION (T[i, k] * T[k+1,s])
If a is in T[1,n] return true. Else, return false.
```

5 Problem 6.7

Let the input string be $x_1 x_2 \ldots x_n$. Let $T[i,j]$ be the length of the longest palindromic subsequence in $x[i..j]$. We have

$$\begin{cases} T[i.i] = 1 \\ T[i,i+1] = 2 \text{ if } x[i] = x[i+1] \text{ and } 0 \text{ if } x[i] \neq x[i+1] \\ T[i,j] = T[i+1,j+1] \text{ if } x[i] = x[j] \text{ and } \max\{T[i+1,j], T[i,j+1]\} \text{ else} \end{cases}$$

6 Problem 6.8

Let $E[i,j]$ be the length of the largest common substring of $x_1 x_2 \ldots x_i$ and $y_1 y_2 \ldots y_j$ such that $x_i = y_j$. We see that $E[1,j] = 1$ if $x_1 = y_j$ and 0 otherwise. Similarly, $E[j,1] = 1$ if $y_1 = x_j$ and 0 otherwise.
In general, we have

$$E[i,j] = \begin{cases} E[i-1,j-1] + 1 \text{ if } x_i = y_j \\ 0 \text{ if } x_i \neq y_j \end{cases}$$

7 Problem 6.9

Let the input string be $x[0..n-1]$ and the input breakpoint array be $y[1..m]$. Convert y to $y[0..m+1]$ and let $y[0] = -1, y[m+1] = n-1$.
Let $M(i,j)$ be

$$\begin{cases} M(i,i) = 0, \forall i : 0 \le i \le m+1 \\ M(i,i+1) = 0, \forall i : 0 \le i \le m+1 \\ M(i,j) = (y[j] - y[i]) + \min\limits_{l: i<l<j} \{M(i,l) + M(l,j)\} \end{cases}$$

8 Problem 6.10

Let $E[i,j]$ be the probability of obtaining exactly i heads when j coins $c_1, c_2, \ldots, c_j$ with head-probability $p_1, p_2, \ldots, p_j$ are tossed. We have $E[0,j] = (1-p_1)(1-p_2)\ldots(1-p_j)$ and

$$E[i,j] = 1 - \sum_{k=1}^{i} E[i-k,j].$$

9 Problem 6.11

Let $E[i, j]$ be the longest common subsequence of $x_1 x_2 \dots x_j$ and $y_1 y_2 \dots y_j$.
We have $E[1, j] = 1$ if $x_1 = y_j$, for all $1 \leq j \leq m$. Similarly, $E[j, 1] = 1$ if $x_j = y_1$, for all $1 \leq j \leq n$.
In general we have

$$E[i, j] = \begin{cases} E[i-1, j-1] + 1 \text{ if } x_i = y_j \\ \max\{E[i-1, j] + E[i, j-1]\} \text{ if } x_i \neq y_j. \end{cases}$$