CSCI 6470
Homework 3
Jiakuan Li

1.

Algorithm 1 PAIRWISEALIGHMENT(X, Y)

$m = length(X)$
$n = length(Y)$
$S[0][0] = 0$
for $i = 1$ to m **do**
  $S[i][0] = S[i-1][0] + d[X[i],'-']$
end for
for $j = 1$ to n **do**
  $S[0][j] = S[0][j-1] + d['-', Y[j]]$
end for
$a = S[i-1][j-1] + d[X[i], Y[j]]$
$b = S[i-1][j] + d[X[i],'-']$
$c = S[i][j-1] + d['-', Y[j]]$
if $a > b$ and $a > c$ **then**
  $S[i][j] = a$
  $v[i][j] =' /'$
else if $b > a$ and $b > c$ **then**
  $S[i][j] = b$
  $v[i][j] =' X'$
else
  $S[i][j] = c$
  $v[i][j] =' Y'$
end if
return S

Algorithm 2 TRACEALIGNMENT(X,Y,X',Y',v,m,n)

```
if m = 0 or n = 0 then
  return
else
  if v[m][n]= '/' then
    TRACEALIGNMENT(X,Y,X',Y',v,m,n)
    X'[m] = X[m]
    Y'[n] = Y[n]
  else if v[m][n] = 'X' then
    TRACEALIGNMENT(X,Y,X',Y',v,m − 1,n)
    X'[m] = '-'
    Y'[n] = Y[n]
  else
    TRACEALIGNMENT(X,Y,X',Y',v,m,n − 1)
    X'[m] = X[m]
    Y'[n] = '-'
  end if
end if
print X'
print Y'
```

2.

(1) $V_M = max($
$V_M(i-1, j-1) \cdot t_{M_{j-1}}, M_j \cdot e_{M_j}(x_i)$
$V_I(i-1, j-1) \cdot t_{I_{j-1}}, M_j \cdot e_{M_j}(x_i)$
$V_D(i-1, j-1) \cdot t_{D_{j-1}}, M_j \cdot e_{M_j}(x_i)$
)

$V_I = max($
$V_M(i-1, j-1) \cdot t_{M_j}, I_j \cdot e_{M_j}(x_i)$
$V_I(i-1, j-1) \cdot t_{I_j}, I_j \cdot e_{M_j}(x_i)$
$V_D(i-1, j-1) \cdot t_{D_j}, I_j \cdot e_{M_j}(x_i)$
)

$V_I = max($
$V_M(i, j-1) \cdot t_{M_j}, D_j$
$V_I(i, j-1) \cdot t_{I_j}, D_j$
$V_D(i, j-1) \cdot t_{D_j}, D_j$
)

Basecase:
$V_M(0, 0) = 1$
$V_I(0, 0) = 0$
$V_D(0, 0) = 0$

Algorithm 3 COMPUTE

(2)
```
mtable[i][j]
itable[i][j]
dtable[i][j]
mtable(0, 0) = 1
itable(0, 0) = 0
dtable(0, 0) = 0
for i = 1 to i do
  for j = 1 to j do
    mtable[i][j] = V_M(i, j)
    itable[i][j] = V_I(i, j)
    dtable[i][j] = V_D(i, j)
  end for
end for
```

3.

(1) Find the store that covers most houses every time. Then remove those covered houses. Then repeat the this steps until all the houses are covered.

(2) Input: an array fills with 'h', and 'e'. 'h' stands for house, and 'e' stands for empty spot.

Algorithm 4 FINDLOCATION(S)

while number of 'h' != 0 **do**
  select the spot that covers the most houses
  mark the spot with 's' stands for 'shop'
  mark the covered house with 'c' stands for 'covered'
end while
the 's' in the array
print the index of 's' in the array

4.

Let w_h, v_h be the weight available and value of the item with the highest value/pound ratio. Let $L(i)$ be the weight of item i contained in the thief's loot L. Therefore, the total value V is

$$\sum_{i=1}^{n} L(i) \cdot \frac{v_i}{w_i}$$

If some of item h is left, and $L(j) \neq 0$ for some $j \neq h$, then replacing j with h will yield a higher value.
Now, there are two situations that leads to an optimal solution.
Case 1:
if $W < W_n$, the $L(h) = W$ and for all $j \neq h$, $L(j) = 0$ is the only take.
Case 2:
There is room left after greedy choice. $L(i), L(j) > 0, i \neq j$. Assume item i was the first choice and $j = h$ a subsequent choice.
Therefore, $V = L(i)\frac{v_i}{w_i} + L(h)\frac{v_h}{w_h}$
If we choose the same amount $L(h)$ of item h first, then the value will be

$$V' = L(h)\frac{v_h}{w_h} + L(i)\frac{v_i}{w_i} = V$$

This actually the optimal solution.