Assignment-4

1.) Selecting using least duration

i	1	2	3	4
s_i	1	3	0	5
f_i	4	7	6	7
D_i	3	4	6	2

Greedy Sol^n = {4}
Optimal Sol^n = {1, 4}

(i)

i	1	2	3	4	5	6
s_i	0	0	3	6	9	7
f_i	2	2	5	8	10	10
No. of Conflicts	2	2	0	1	1	1

Greedy Solution = i={3}
Optimal Solution = {3, 4, 5}

(ii) ~~Take the same~~

i	1	2	3	4
s_i	1	3	0	5
f_i	4	7	6	7

Greedy Solution = {3, 1}
But Incompatible, as they conflict.

2.)(a) arr[ ] = {25, 10, 5, 1}

K = 4 n = n

for{ i = 1 to K

d = n / arr[i]

printf(" The denominations are : ");

r = n % arr[i]

n = r

Proof:

Let (a_1', a_2', a_3', a_4') be the optimal solution and (a_1, a_2, a_3, a_4) be the greedy solution where they give no. of coins of Quarters, nickel, dime and pennies respectively.

Considering the casses :-

if $a_3' >= 3$, greedy solution will give a quarter and a nickel which is less no. of coins.

The algorithms contradict here.

Similarly,

if $a_3 <= 2$, greedy solution will give combination of a_1, a_2, a_3 and a_4.

So, there exists a solution (a_1, a_2', a_3', a_4') to give optimal solution.

Hence, the other a_2, a_3 and a_4 can be

proved in similar way to give similar optimal solution.
So, $(a_1', a_2', a_3', a_4') = (a_1, a_2, a_3, a_4)$

(b) arr[i] = func(c, k)

for(i = K to =0
    d = n/arr[i]
    r = n%arr[i]
    n = r

Proof -
Let $(a_K', a_{K-1}' \ldots a_0')$ be the optimal solution and $(a_K, a_{K-1} \ldots a_0)$ be the greedy solution for certain value of c and K, where the give no. of coins for $(c^K, c^{K-1}, \ldots c^0)$ value of coins.
Considering the case :-
if a let c = 2 K = 3
$C = (2^3, 2^2, 2^1, 2^0)$
(a_3', a_2', a_1', a_0')
if $a_2' \geq 2$, greedy solution will give 1 coins of a_3 or $2^3 = 8$.
So, This is a contradiction.
$a_2' <= 1$, greedy will give combination of a_0 or a_1.

So, there exists a solution (a_3', a_2, a_1, a_0'). This can be proved for all other values as well.

Hence, $(a_k', a_{k-1}', \ldots a_0') = (a_k, a_{k+1}, \ldots, a_0)$.

This proves that greedy gives an optimal solution.

(c) Consider the coins $C = \{1, 4, 5\}$

$n = 8$ cents

So, greedy solution is - $5 \times 1 + 1 \times 3 = 8$

No. of coins = $4 = 3+1$

Optimal solution - $4 \times 2 = 8$ cents

No. of coins = 2

(d) int map [c] [n+1] c → No. of coins

n → Value to be made change of

```
for i = 0 to c
    for j = 0 to n+1
        map[i][0] = 0

for j = 0 to n+1
    map[0][j] = j
```

```
for (i=1 to c
    for j = 0 to n+1
        if j >= coins[i]
            map[i][j] = min(map[i-1][j],
                coin=i | map[i][j - coins[i]])
        else
            map[i][j] = map[i-1][j]

printf(map[c][n])
printf("coins used are: "coin)
```