CSC373 Winter 2015 Problem Set # 1

Name: Weidong An
Student Number: 1000385095
UTOR email: weidong.an@mail.utoronto.ca
January 13, 2015

(a) The pseudocode is as follows:

```
OPTIMAL-PERMUTATION()
π = empty array
T = {1, ..., n}
while T ≠ ∅
    min = ∞
    for each p ∈ T // This for-loop finds one of the minimum process in T
        if t_p < min
            min = t_p
            min-process = p
    append min-process to the end of π
    T = T − {min-process}
return π
```

(b) Define π_i be the array π after the i^{th} iteration of the while-loop.
The example: $t_1, t_2, t_3, t_4 = 2, 5, 3, 2$
Then the "partial solutions" are:
$\pi_0 =$ empty array
$\pi_1 = [1]$
$\pi_2 = [1, 4]$
$\pi_3 = [1, 4, 3]$
$\pi_4 = [1, 4, 3, 2]$

(c) Definition: Let π_{OPT} be an optimal permutation of $1, ..., n$ that minimizes the average completion time $(C_1 + ... + C_n)/n$. We say π_{OPT} extends π_i if $\pi_i = \pi_{OPT}[1..i]$. π_1 is promising if there exists some optimal permutation π_{OPT} of $1, ..., n$ that π_{OPT} extends π_i.

(d) Loop invariant: π_i is promising for all $i = 0, ..., n$.
The quantity to induction on: the number of iterations i of the while-loop.

(e) Only one case would be considered. The choice made by the greedy algorithm is the process with minimum processing times among the left processes.

(f) Notation: Let $min\text{-}process_i$ be the value of $min\text{-}process$ after the i^{th} iteration of the while-loop. In the induction hypothesis, let π_{OPT} be the optimal permutation for $i = k$. (I.H: π_k is promising.)
Subcase 1: $min\text{-}process_{k+1} = \pi_{OPT}[k + 1]$
Subcase 2: $min\text{-}process_{k+1} \neq \pi_{OPT}[k + 1]$

(g) Suppose π_i is promising for all $i = 0, ..., n$ which means there exists some optimal permutation π_{OPT} of $1, ..., n$ that π_{OPT} extends π_i for all $i = 0, ..., n$.
In particular, π_n is promising. (i.e There is an optimal permutation π_{OPT} that $\pi_n = \pi_{OPT}[1..n] = \pi_{OPT}$.) The loop terminates after n iterations since the cardinality of T decrements by 1 after each iteration. Hence, the result returned by the algorithm is $\pi_n = \pi_{OPT}$. This has proved that the algorithm from part (a) always produces an optimal solution.