Derek McCarthy B00007439
CA 2 Derivation of Algorithms

Question
Derive a solution for the following program specification:

|[
con N: int; {N > 0}
f: array [0..N) of char;
var
freq: int;

S

{freq = #i : 0 ≤ i < N : f.i = 'A'}
]|

Step 1 – Write down an invariants P0 and P1.

Replacing N with n.

P0 : freq = #i : 0 ≤ i < n : f.i = 'A'

P1 : 0 ≤ n ≤ N

Step 2 – Write out an outlier solution

S0;

{P, Bound function N-n}

do n ≠ N →

{P ∧ n ≠ N}

S1;

{P}

od

{P ∧ n = N}

{freq = #i : 0 ≤ i < N : f.i = 'A'}

Step 3 – Derive S0, variable initialisation.

Suggest n := 0;

(P0 ∧ P1)(n := 0)

≣ [Substitution]

freq = #i : 0 ≤ i < 0 : f.i = 'A' ∧ 0 ≤ 0 ≤ N

≣ [Contradiction]

freq = #i : false : f.i = 'A' ∧ 0 ≤ 0 ≤ N

≣ [# Over empty range]

freq = 0 ∧ 0 ≤ 0 ≤ N

≣ [Exclude the middle, Constants]

freq = 0

Therefore S0 ; given by,
freq := 0;
n := 0;

Step 4 – Derive S1; loop body
Suggest n := n + 1

(P0)(n := n + 1)

≣ [Substitution]

freq = #i : 0 ≤ i < n + 1 : f.i = 'A'

≣ [Split off i = n]

freq = (#i : 0 ≤ i < n : f.i = 'A') + #(f.n = 'A')

≣ [⇐ P]

freq = freq + #(f.n = 'A')

≣ [Case Analysis]

freq := freq + 1, if f.n = 'A'

freq := freq + 0, if f.n ≠ 'A'

<u>Logic behind case analysis</u>

≣ [if..fi]
if f.n = 'A' →
freq := freq + 1

[] f.n ≠ ‘A’ →
Skip;
fi

Therefore S1; becomes
if f.n = ‘A’ →
freq := freq + 1
[] f.n ≠ ‘A’ →
Skip;
fi
n : n + 1

Step 5 – Prove termination

Initialisation	Loop Body
(N – n ≥ 0) (n := 0)	(N - n) (n := n + 1)
≣ [Substitute]	≣ [Substitute]
N – 0 ≥ 0	N – (0 + 1)
≣ [Arithmetic]	≣ [Arithmetic]
N ≥ 0	N – n - 1
⇐ {Given N ≥ 0}	<
	N - n
	Therefore Decreasing

Step 6 - Complete Solution

```
|[ Con N : int; {N ≥ 0}
    f: array [0..N) of char;
    var
    freq, n : int;
    freq, n := 0, 0;
    do n ≠ N →
        if f.n = ‘A’ →
            freq := freq + 1;
        [] f.n ≠ ‘A’ →
            Skip;
```

```
            fi
        od
        {freq = #i : 0 ≤ i < N : f.i = 'A'}
]|
```