coefficient will be added to the polynomial D. We note that the avail variable stores the current term's posn plus one of the polynomial D. According to our assumption, ∀K, startB < K <= finishB, terms[K].expon < terms[y].expon. Also, ∀K, startA < K <= finishA, terms[K].expon < terms[x].expon. ∴ It's easy to see that terms[y].expon has the only possible coefficient for the polynomial D, which is terms[y].coef, since it can't be added to others.

∴ We have correctly added the coefficient to the apt exponent on line 7.

- On line 8, we have increased startB by 1 and on line 9, we have broken from the switch block.
- It's easy to see that the loop invariant still holds before the start of the next iteration.
- When terms[x].expon > terms[y].expon, the COMPARE function returns 1. The case is evaluated in lines 16-18. Why we did what we did and why it's correct is exactly similar to the above proof.

- Now, let's consider the case where terms[x].expon = terms[y].expon. The COMPARE function returns 0 and we evaluate lines 10-15.

On line 11, we store the summation of terms[x] and terms[y]'s coefficients and store it in coefficient. If the coefficient is non-zero (line 12), we use the attach function to add the corresponding coefficient and the exponent to the polynomial D on line 13, and we know that the attach function is correct.

- By our initial assumption, terms[0].expon > terms[i_1].expon > terms[x].expon > terms[i_2].expon > terms[finishA].expon, where $0 < i_1 < x < i_2 < finishA$, $\forall i_1, i_2$.
- Similarly, terms[finishA+1].expon > terms[i_3].expon > terms[y].expon > terms[i_4].expon > terms[finishB].expon, where $finishA+1 < i_3 < y < i_4 < finishB$, $\forall i_3, i_4$.

∴ (terms[x].coef + terms[y].coef) is the only possible coefficient of $x^{terms[x].expon}$ for the polynomial D, ∵ only like coefficients can be added.