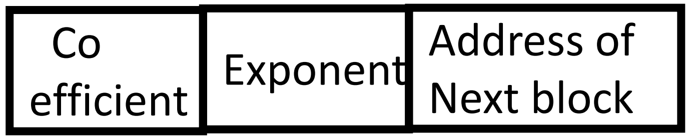
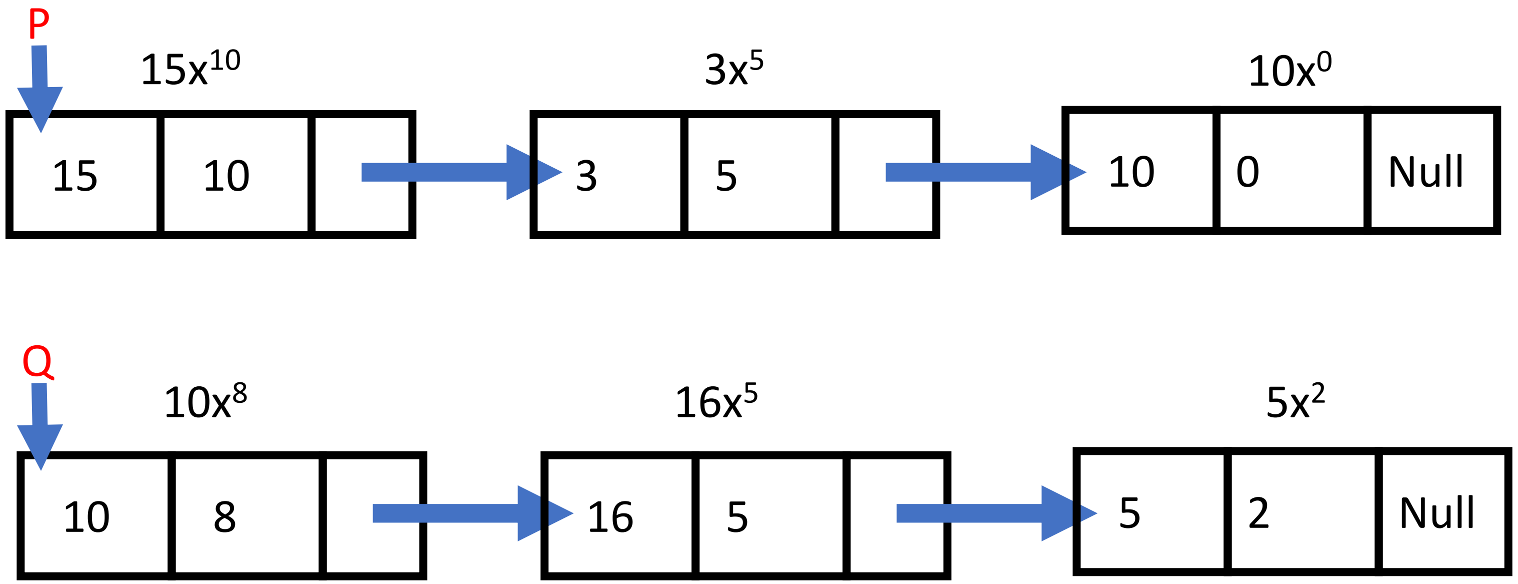
Handling Polynomial Using Linked List

Representing Polynomials using Linked List

$P(x) = 15x^{10}+3x^{5}+10$

$Q(x) = 10x^{8}+16x^{5}+5x^{2}$





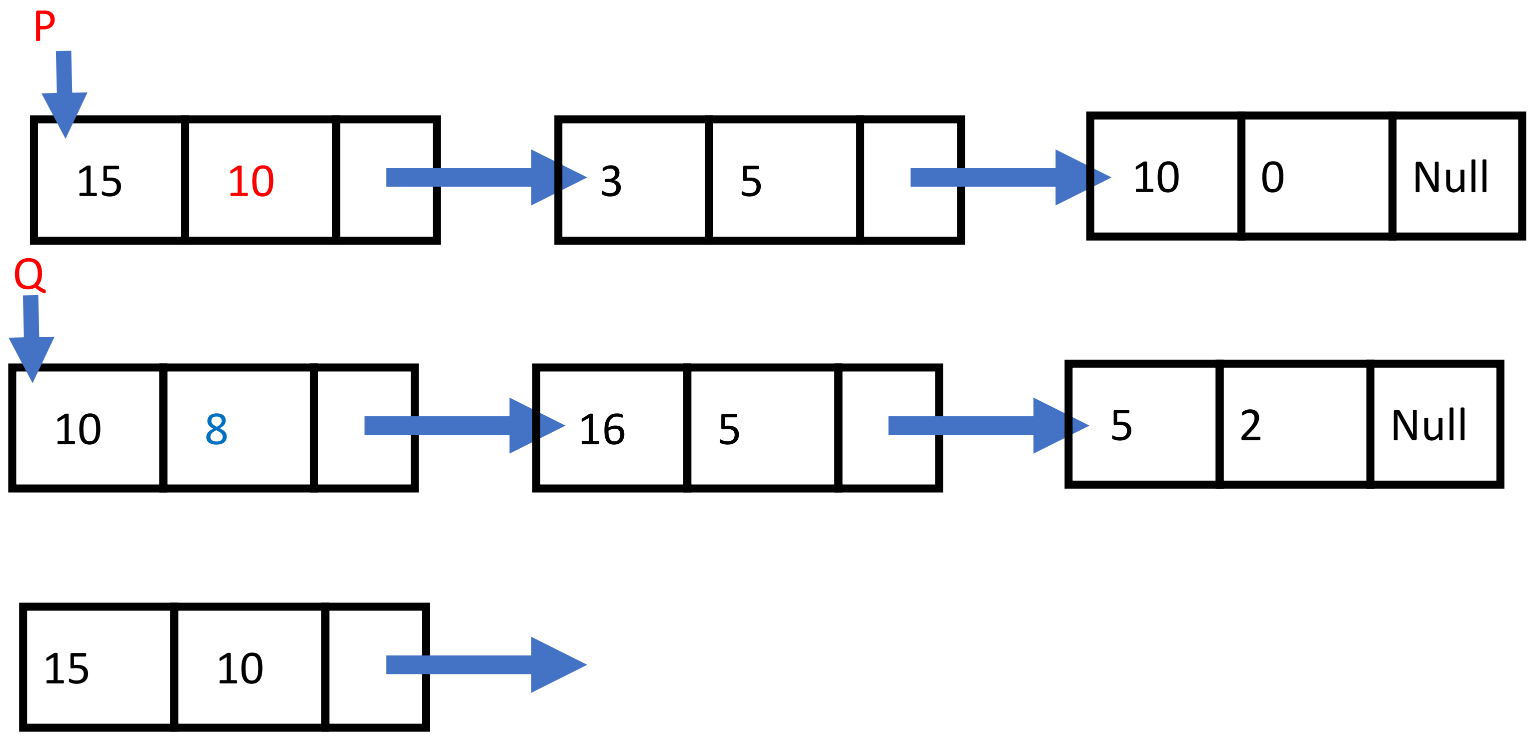
Adding Polynomials using Linked List

$P(x) = 15x^{10} + 3x^5 + 10$

$Q(x) = 10x^8 + 16x^5 + 5x^2$

$P(x) + Q(x) = 15x^{10} + 10x^8 + 19x^5 + 5x^2 + 10$

Check Exponent



Move P one step

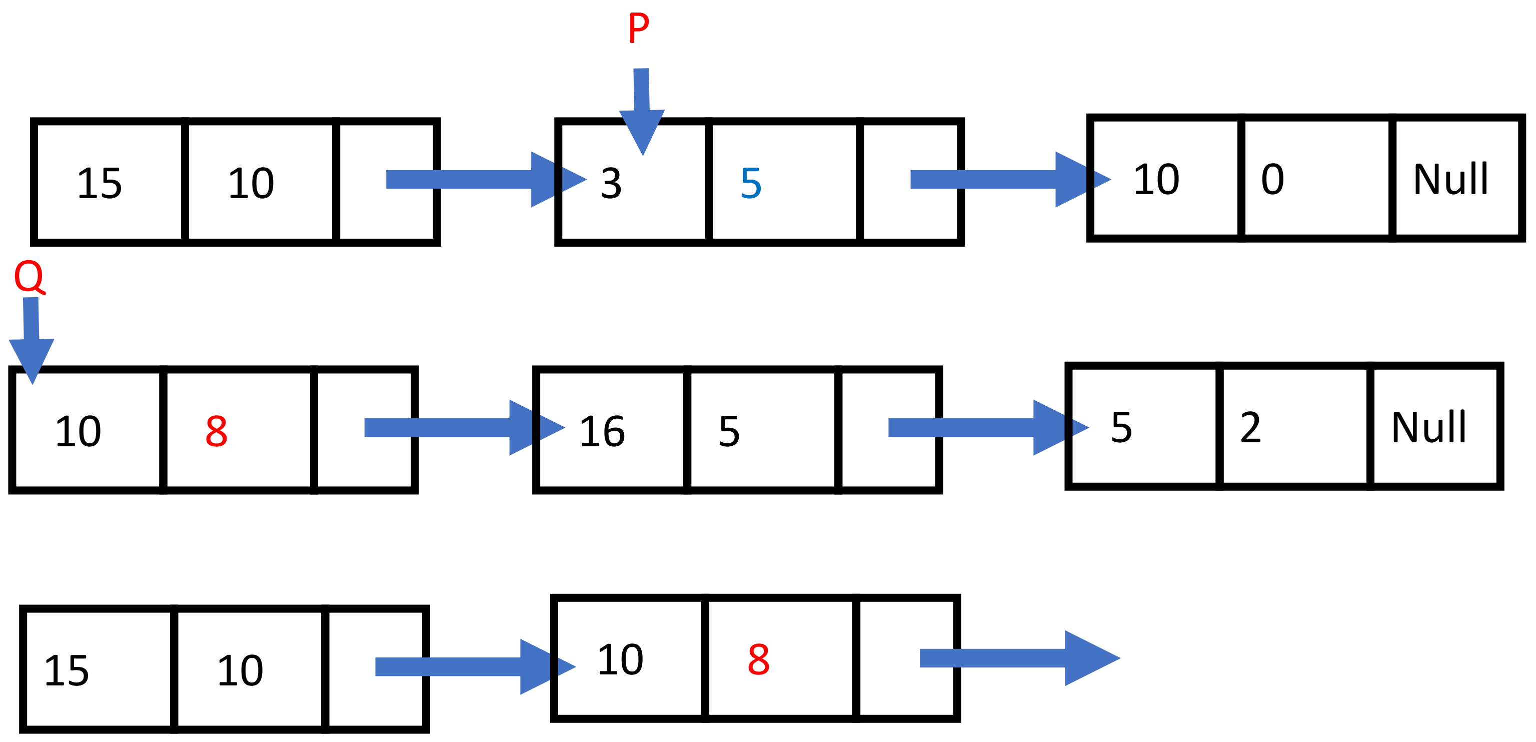
Adding Polynomials using Linked List

$P(x) = 15x^{10}+3x^{5}+10$

$Q(x) = 10x^{8}+16x^{5}+5x^{2}$

$P(x)+Q(x) = 15x^{10} + 10x^{8}+19x^{5}+5x^{2}+10$

Check Exponents



Move Q one step

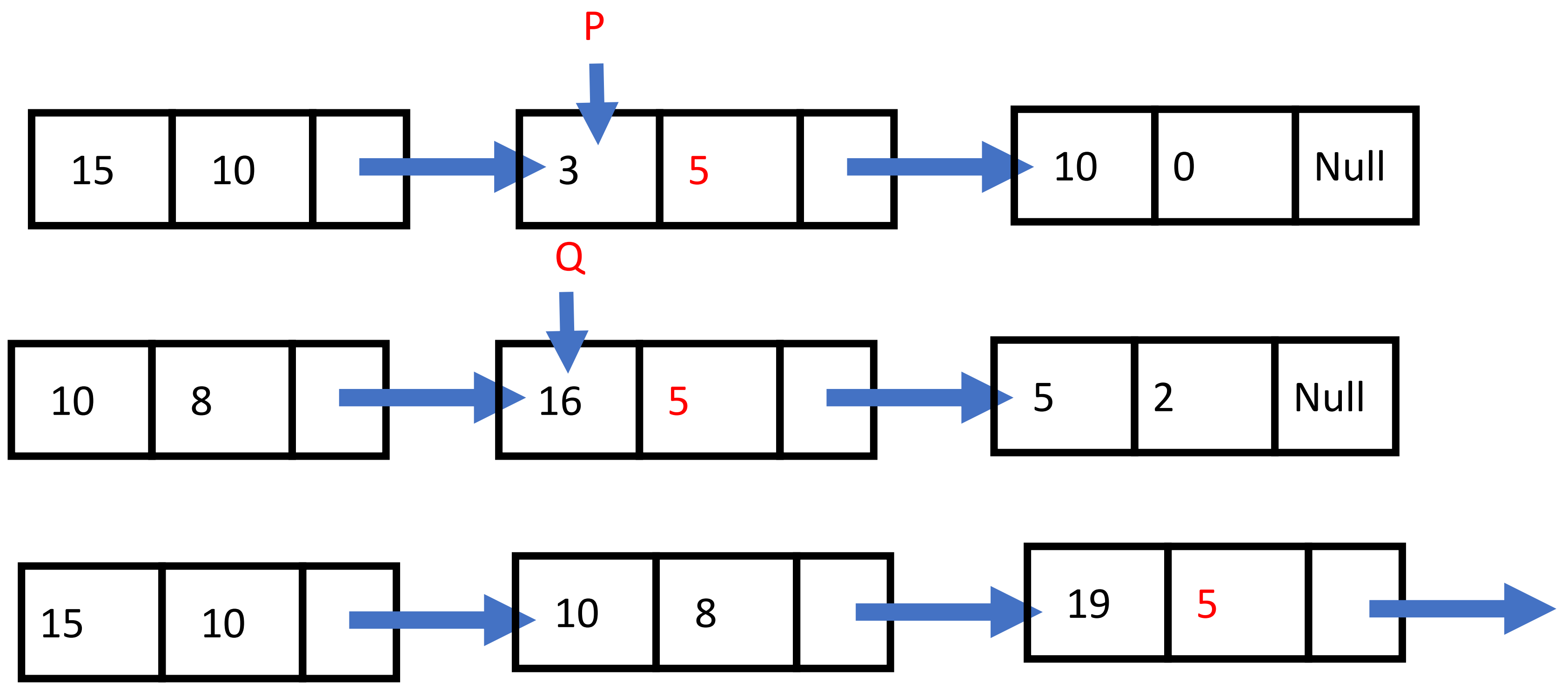
Adding Polynomials using Linked List

$P(x) = 15x^{10} + 3x^5 + 10$

$Q(x) = 10x^8 + 16x^5 + 5x^2$

$P(x)+Q(x) = 15x^{10} + 10x^8 + 19x^5 + 5x^2 + 10$

Same Exponent-Add!!



Move P and Q both

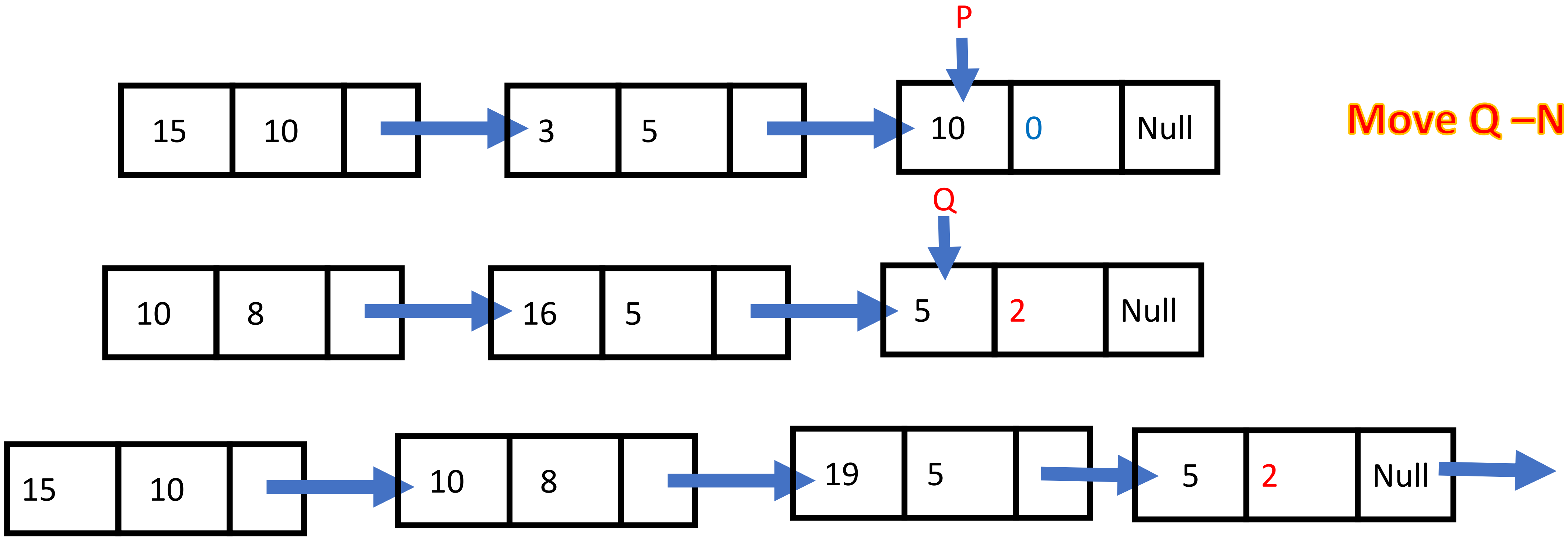
Adding Polynomials using Linked List

$P(x) = 15x^{10}+3x^5+10$

$Q(x) = 10x^8+16x^5+5x^2$

$P(x)+Q(x) = 15x^{10} + 10x^8+19x^5+5x^2+10$

Check Exponent



Move Q –Null-So end

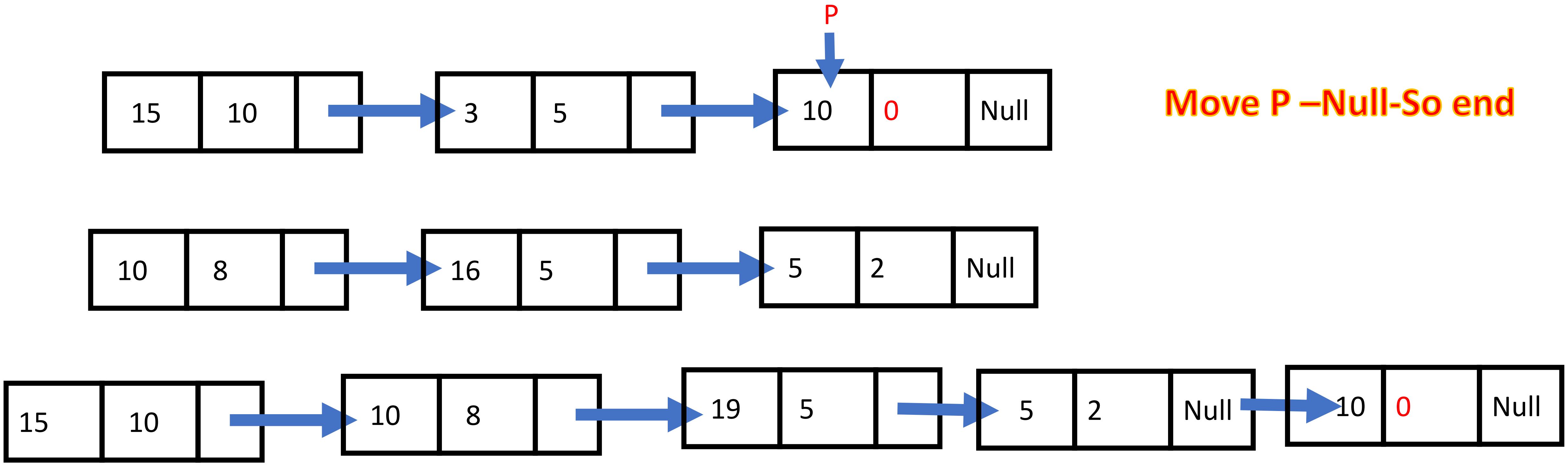
Adding Polynomials using Linked List

$P(x) = 15x^{10}+3x^5+10$

$Q(x) = 10x^8+16x^5+5x^2$

$P(x)+Q(x) = 15x^{10} + 10x^8+19x^5+5x^2+10$

Check Exponent



Move P –Null-So end

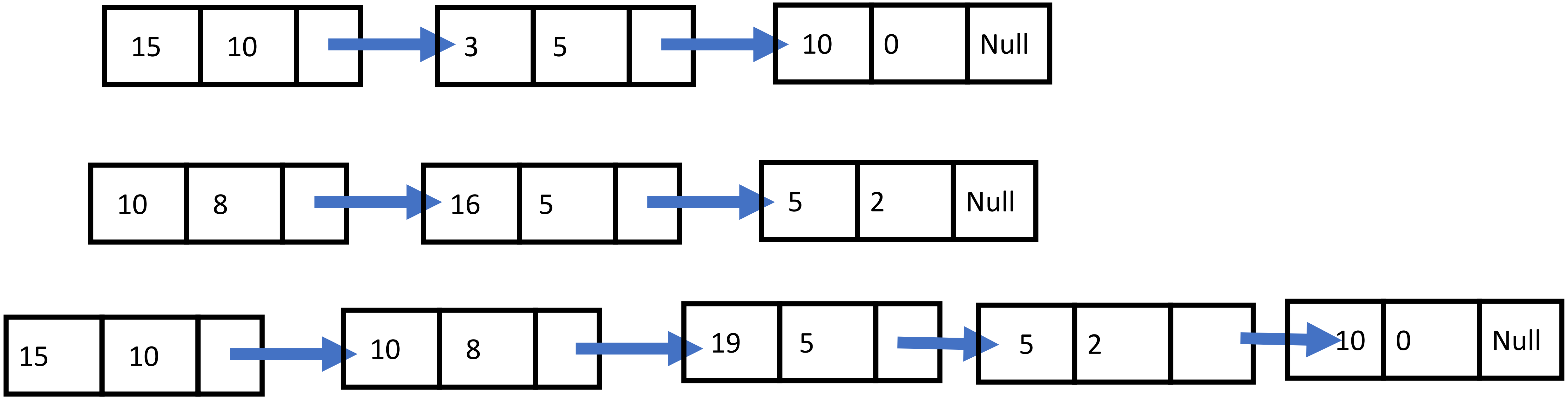
Adding Polynomials using Linked List

$P(x) = 15x^{10}+3x^5+10$

$Q(x) = 10x^8+16x^5+5x^2$

$P(x)+Q(x) = 15x^{10} + 10x^8+19x^5+5x^2+10$

End of Addition!!



Print the final linked list with a pointer

$15x^{10} + 10x^8+19x^5+5x^2+10$

Implementation

```
#include<bits/stdc++.h>
using namespace std;
struct Node{
  int coeff;
  int pow;
  struct Node *next;
};

void create_node(int x, int y, struct Node **temp){
  struct Node *r, *z;
  z = *temp;
  if(z == NULL){
    r =(struct Node*)malloc(sizeof(struct Node));
    r->coeff = x;
    r->pow = y;
    *temp = r;
    r->next = (struct Node*)malloc(sizeof(struct Node));
    r = r->next;
    r->next = NULL;
  } else {
    r->coeff = x;
    r->pow = y;
    r->next = (struct Node*)malloc(sizeof(struct Node));
    r = r->next;
    r->next = NULL;
  }
}
```

```
void polyadd(struct Node *p1, struct Node *p2, struct Node *result){
  while(p1->next && p2->next){
    if(p1->pow > p2->pow){
      result->pow = p1->pow;
      result->coeff = p1->coeff;
      p1 = p1->next;
    }
    else if(p1->pow < p2->pow){
      result->pow = p2->pow;
      result->coeff = p2->coeff;
      p2 = p2->next;
    } else {
      result->pow = p1->pow;
      result->coeff = p1->coeff+p2->coeff;
      p1 = p1->next;
      p2 = p2->next;
    }
    result->next = (struct Node *)malloc(sizeof(struct Node));
    result = result->next;
    result->next = NULL;
  }
  while(p1->next || p2->next){
    if(p1->next){
      result->pow = p1->pow;
      result->coeff = p1->coeff;
      p1 = p1->next;
    }
    if(p2->next){
      result->pow = p2->pow;
      result->coeff = p2->coeff;
      p2 = p2->next;
    }
    result->next = (struct Node *)malloc(sizeof(struct Node));
    result = result->next;
    result->next = NULL;
  }
}
```

Implementation

```
void printpoly(struct Node *node){
  while(node->next != NULL){
    printf("%dx^%d", node->coeff, node->pow);
    node = node->next;
    if(node->next != NULL)
      printf(" + ");
  }
}
```

```
int main(){
   struct Node *p1 = NULL, *p2 = NULL, *result = NULL;
   create_node(41,7,&p1);
   create_node(12,5,&p1);
   create_node(65,0,&p1);
   create_node(21,5,&p2);
   create_node(15,2,&p2);
   printf("polynomial 1: ");
   printpoly(p1);
   printf("\npolynomial 2: ");
   printpoly(p2);
   result = (struct Node *)malloc(sizeof(struct Node));
   polyadd(p1, p2, result);
   printf("\npolynomial after adding p1 and p2 : ");
   printpoly(result);
   return 0;
}
```

Output

```
polynomial 1: 41x^7 + 12x^5 + 65x^0
polynomial 2: 21x^5 + 15x^2
polynomial after adding p1 and p2 : 41x^7 + 33x^5 + 15x^2 + 65x^0
```

Thank You