선택 정렬.

```c
void Select_Sort(int*a, int count) //정렬할 배열의 원소 개수
{
    int i, j;
    int min_value, min_index;

    for(i=0; i<count;i++) //count-1까지. 왜? count +1 로 해도 값 같은데. 최소한의 배열 개수 표현한거. count -2 부터 값 달라짐.
    {
        // 현재 위치를 최소값으로 초기화
        min_index = i;
        min_value = a[i];

        // 현재 위치 이후의 배열을 순회하며 최소값을 찾음
        for(j=i+1; j< count; j++)
        {
            if(min_value > a[j])
            {
                // 현재까지의 최소값보다 작은 값을 찾으면 최소값과 인덱스 업데이트
                min_index = j;
                min_value = a[j];
            }
        }
    // 현재 위치의 값과 최소값 위치의 값을 교환
    a[min_index] = a[i];
    a[i] = min_value;
    }
}
```

버블 정렬

```c
void Bubble_Sort(int *a, int count)
{
    int i, j;
    int temp;

    // 배열을 처음부터 끝까지 반복
    for(i = 0;i < count-1;i++)
    {
        // 각 패스마다 큰 값이 배열 끝으로 이동하도록 반복
        for(j = 1; j < count-i;j++)
        {
            // 현재 원소와 이전 원소 비교하여 크기 순서 맞춤
            if(a[ j-1] > a[ j])
            {
                // Swap: 이전 원소와 현재 원소의 위치를 교환
                temp = a[ j-1];
                a[j -1] = a[ j];
                a[j] = temp;
            }
        }
    }
}
```

삽입 정렬

```c
void Insert_Sort(int* a, int count)
{
    int i, j;
    int temp;

    // 배열을 처음부터 끝까지 반복
    for (i = 1; i < count; i++)
    {
        temp = a[i];
        j = i;

        // 현재 원소와 이전 원소를 비교하면서 정렬
        while ((a[j - 1] > temp) && (j > 0))
        {
            a[j] = a[j - 1];
            j = j - 1;
        }

        // 정렬된 위치에 현재 원소 삽입
        a[j] = temp;
    }
}
```

셸 정렬

```c
void Shell_Sort(int* a, int count)
{
    int i, j, inc, h;
    int temp;

    // 간격(h)을 조정하며 반복
    for (h = count / 2; h > 0; h /= 2)
    {
        // 각 부분 배열에 대해 삽입 정렬 수행
        for (i = 0; i < h; i++)
        {
            for (j = i + h; j < count; j += h)
            {
                temp = a[j];
                inc = j;

                // 간격(h)에 따라 삽입 정렬 수행
                while (inc > h - 1 && a[inc - h] > temp)
                {
                    a[inc] = a[inc - h];
                    inc = inc - h;
                }

                a[inc] = temp;
            }
        }
    }
}
```

퀵 정렬

```
void Quick_Sort(int* a, int count)
{
    int i, j;
    int v, temp;
    if (count > 1)// 배열 크기가 1보다 큰 경우에만 정렬 수행
    {
        v = a[count - 1]; // 기준 원소를 배열의 마지막 원소로 선택

        i = -1;
        j = count - 1;

        for (;;)// 무한 루프를 돌면서 정렬 수행
        {
            while (a[++i] < v);// 기준 원소보다 작은 값을 찾음
            while (a[--j] > v);// 기준 원소보다 큰 값을 찾음
            if (i >= j) break;// i와 j가 교차하면 반복 중단
            temp = a[i];// a[i]와 a[j]의 위치를 교환
            a[i] = a[j];
            a[j] = temp;
        }
        temp = a[i];// 기준 원소와 a[i]의 위치를 교환
        a[i] = a[count - 1];
        a[count - 1] = temp;

        Quick_Sort(a, i);// 분할된 부분 배열에 대해 재귀적으로 정렬 수행
        Quick_Sort(a + i + 1, count - i - 1);
    }
}
```