

Bubble_sort.c

```
void bubbleSort(int arr[], int n) {
    int i, j;
    for(i = 0; i < n - 1; i++) {
        for(j = 0; j < n - i - 1; j++) {
            if(arr[j] > arr[j+1]) {
                int temp = arr[j];
                arr[j] = arr[j+1];
                arr[j+1] = temp;
            }
        }
    }
}
```

Insertion_sort.c

```
void insertionSort(int arr[], int n) {
    int i, j, key;
    for(i = 1; i < n; i++) {
        key = arr[i];
        j = i - 1;
        while(j >= 0 && arr[j] > key) {
            arr[j+1] = arr[j];
            j--;
        }
        arr[j+1] = key;
    }
}
```

Selection_sort.c

```
void selectionSort(int arr[], int n) {
    int i, j, min_idx;
    for(i = 0; i < n - 1; i++) {
        min_idx = i;
        for(j = i + 1; j < n; j++) {
            if(arr[j] < arr[min_idx])
                min_idx = j;
        }
        if(min_idx != i) {
            int temp = arr[i];
            arr[i] = arr[min_idx];
            arr[min_idx] = temp;
        }
    }
}
```

Main.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>

// 앞서 정의한 정렬 함수들의 프로토타입 선언
void bubbleSort(int arr[], int n);
void selectionSort(int arr[], int n);
void insertionSort(int arr[], int n);

int main() {
    int sizes[] = {50000, 100000, 200000, 400000, 1000000};
    int numSizes = sizeof(sizes) / sizeof(sizes[0]);
    int i, j;

    for(i = 0; i < numSizes; i++) {
        int n = sizes[i];
        int *arr = (int*) malloc(n * sizeof(int));
        int *temp = (int*) malloc(n * sizeof(int));

        for(j = 0; j < n; j++) {
            arr[j] = rand();
        }

        clock_t start, end;
        double time_taken;

        memcpy(temp, arr, n * sizeof(int));
        start = clock();
        bubbleSort(temp, n);
        end = clock();
        time_taken = (double)(end - start) / CLOCKS_PER_SEC;
        printf("버블 정렬, n = %d, 실행 시간: %f 초\n", n, time_taken);

        memcpy(temp, arr, n * sizeof(int));
        start = clock();
        selectionSort(temp, n);
        end = clock();
        time_taken = (double)(end - start) / CLOCKS_PER_SEC;
        printf("선택 정렬, n = %d, 실행 시간: %f 초\n", n, time_taken);

        memcpy(temp, arr, n * sizeof(int));
        start = clock();
        insertionSort(temp, n);
        end = clock();
        time_taken = (double)(end - start) / CLOCKS_PER_SEC;
        printf("삽입 정렬, n = %d, 실행 시간: %f 초\n", n, time_taken);
    }
}
```

```

        printf("-----\n");

        free(arr);
        free(temp);
    }

    return 0;
}

```

컴파일

```
gcc main.c bubble_sort.c insertion_sort.c selection_sort.c -o sort_test -O2
```

결과


N/초(s)	버블 정렬	선택 정렬	삽입 정렬
5 만	4.90	5.05	0.51
10 만	19.57	20.20	2.03
20 만	78.75	80.81	8.13
40 만	315.06	323.62	32.58
100 만	1981.71	2033.48	205.27

결과 사진

```

202020827@cslinux2:~/com/sort$ ls
bubble_sort.c insertion_sort.c main.c selection_sort.c
202020827@cslinux2:~/com/sort$ gcc main.c bubble_sort.c insertion_sort.c selection_sort.c -o sort_test -O2
202020827@cslinux2:~/com/sort$ ./sort_test
버블 정렬, n = 50000, 실행 시간: 4.901962 초
선택 정렬, n = 50000, 실행 시간: 5.049897 초
삽입 정렬, n = 50000, 실행 시간: 0.505919 초
-----
버블 정렬, n = 100000, 실행 시간: 19.571970 초
선택 정렬, n = 100000, 실행 시간: 20.204694 초
삽입 정렬, n = 100000, 실행 시간: 2.028913 초
-----
버블 정렬, n = 200000, 실행 시간: 78.746626 초
선택 정렬, n = 200000, 실행 시간: 80.805809 초
삽입 정렬, n = 200000, 실행 시간: 8.132125 초
-----
버블 정렬, n = 400000, 실행 시간: 315.061639 초
선택 정렬, n = 400000, 실행 시간: 323.617845 초
삽입 정렬, n = 400000, 실행 시간: 32.577873 초
-----
버블 정렬, n = 1000000, 실행 시간: 1981.707610 초
선택 정렬, n = 1000000, 실행 시간: 2033.484689 초
삽입 정렬, n = 1000000, 실행 시간: 205.274112 초
-----

```

 만약 1 억개의 데이터에 대하여 각 정렬을 실행할때 얼마나 걸릴지 추정

If - 모든 정렬 알고리즘이 $O(n^2)$ 시간복잡도를 가진다고 가정하고, $n = 10^6$ 때
 측정된 시간을 기준으로 $\left(\frac{10^8}{10^6}\right) = 10000$ 배 증가한다고 계산

n=1,000,000 일 때의 실제 시간:

- 버블 정렬: 약 1,982 초 (약 33 분)
- 선택 정렬: 약 2,021 초 (약 34 분)
- 삽입 정렬: 약 285 초 (약 5 분)

$O(n^2)$ 가정에 따른 1 억($=10^8$)개 데이터로의 확장

$$T(10^8) = T(10^6) \times \left(\frac{10^8}{10^6}\right)^2 = T(10^6) \times 10,000$$

버블 정렬

$$1981.79 \times 10,000 \approx 1.98 \times 10^7 \text{ 초} \rightarrow \text{약 229 일}$$

선택 정렬

$$2021.33 \times 10,000 \approx 2.02 \times 10^7 \text{ 초} \rightarrow \text{약 234 일}$$

삽입 정렬

$$285.27 \times 10,000 \approx 2.85 \times 10^6 \text{ 초} \rightarrow \text{약 33 일}$$