1.

#include <stdio.h>

#include <stdlib.h>

#define MAX\_NUM 1000

#define BUCKETS 10

#define DIGITS 4

void counting\_sort(int \*list, int n, int exp) {//n은 0~9를 나타냄, k는 몇번 째 자리

int output[MAX\_NUM];

int count[10];

int i;

for (i = 0; i < n + 1; i++) {//Count 변수 초기화

count[i] = 0;

}

for (i = 0; i < MAX\_NUM; i++)//각 자리수의 숫자 count에 넣기

count[(list[i] / exp) % 10]++;

for (i = 1; i < 10; i++)//누적합 계산해 줘서 각 숫자가 어떤 index에 들어갈 지 계산해주기

count[i] += count[i - 1];

for (i = MAX\_NUM - 1; i >= 0; i--)

{

output[count[(list[i] / exp) % 10] - 1] = list[i];//각각 index에 맞게 output에 숫자 넣어주기

count[(list[i] / exp) % 10]--;//이미 숫자 하나가 들어갔으니 index 하나 - 시켜서 output 에 넣어주기 가능

}

for (i = 0; i < MAX\_NUM; i++) {

list[i] = output[i];

}

}

void radix\_sort(int \*list, int n) {

int i = 0;

int exp=1;//자리 수 나누는 변수. 1,10,100,1000까지 증가.

for (i=0; i<DIGITS;i++) {

counting\_sort(list, n, exp);

exp \*= 10;

}

}

int main() {

int i;

int list[MAX\_NUM];

int temp;

for (i = 0; i < MAX\_NUM; i++) {

temp = (rand() / (double)RAND\_MAX) \* 10000;

list[i] = temp;

}

for (i = 0; i < 20; i++)

printf("%d ", list[i]);

radix\_sort(list, 9);

printf("정렬후 \n");

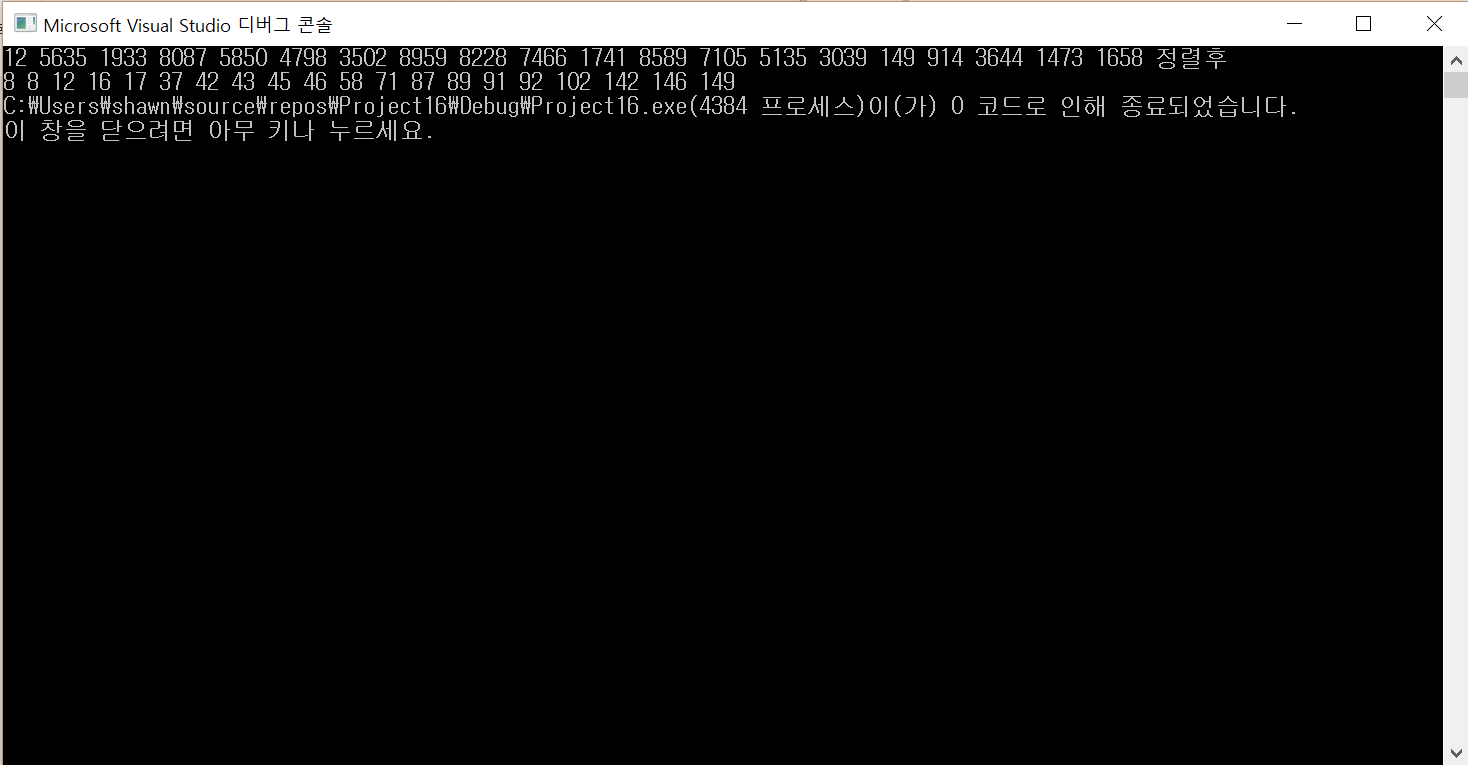
for (i = 0; i < 20; i++)

printf("%d ", list[i]);

}

2. 결과 콘솔 창

출력을 위해 20까지만 출력하였다.



3. 코드분석

Radix sort 의 queue 대신에 counting sort로 대체하려면 우선 일시적으로 output list를 만들어줘야 한다. lsd에서 msd 로 가는 순서이기 때문에 1자리 수부터 먼저 정렬을 해준다. Counting sort 부분에서는 count[]부분에 각각 숫자의 개수를 새어주고, 누적합 계산을 통해서 각각 몇번 인덱스에 들어갈 수 있는지 저장해준다. 그 다음 output list에 count[]를 이용하여 정렬해준다.

데이터의 maximum digit이 4이기 때문에 총 4번의 counting sort 를 해줘야 한다. 이 반복문은 exp에 10을 4번씩 곱해주면서 해주었다. 각각의 counting sort을 통해 output 리스트를 만들고 나서는 다시 원래 list에 복원시켜 주었다.