/\*선택 정렬\*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#define SIZE 10

int main(void)

{

int list[SIZE] = { 3, 2, 9, 7, 1, 4, 8, 0, 6, 5 };

int i, j, temp, least;

for(i = 0; i < SIZE-1; i++)

{

least = i;

for(j = i + 1; j < SIZE; j++)

if(list[j] < list[least])

least = j;

temp = list[i];

list[i] = list[least];

list[least] = temp;

}

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

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

printf("\n");

return 0;

}

/\*이분탐색\*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#define SIZE 16

int binary\_search(int list[], int n, int key);

int main(void)

{

int key;

int grade[SIZE] = { 2,6,11,13,18,20,22,27,29,30,34,38,41,42,45,47 };

printf("탐색할 값을 입력하시오:");

scanf("%d", &key);

printf("탐색 결과= %d\n", binary\_search(grade, SIZE, key));

return 0;

}

int binary\_search(int list[], int n, int key)

{

int low, high, middle;

low = 0;

high = n-1;

while( low <= high ){ // 아직 숫자들이 남아있으면

printf("[%d %d]\n",low,high);

middle = (low + high)/2; // 중간 요소 결정

if( key == list[middle] ) // 일치하면 탐색 성공

return middle;

else if( key > list[middle] )// 중간 원소보다 크다면

low = middle + 1; // 새로운 값으로 low 설정

else

high = middle - 1; // 새로운 값으로 high 설정

}

return -1;

}