**알고리즘YD 20212127 송하성**

**13주차 과제**

**[예제 1] 퀵정렬 (p.419)**

|  |
| --- |
| C |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  #include <stdlib.h>  #include<string.h>  typedef struct employee {  long eid;  char name[20];  int escore;  char dname[5];  }edatatype;  void quick\_sorta(int a[], int left, int right);  void quick\_sortd(int a[], int left, int right);  void quick\_sorta(edatatype a[], int left, int right) {  long pivot;  int i, j;  edatatype tmp;  if (left < right) {  i = left; j = right + 1;  pivot = a[left].eid;  while (i < j) {  do i++;  while ((a[i].eid <= pivot) && (i < right));  do j--;  while((a[j].eid >= pivot) && (j > left));  if (i < j) {  tmp = a[i];  a[i] = a[j];  a[j] = tmp;  }  }  if (j != left) {  tmp = a[j];  a[j] = a[left];  a[left] = tmp;  }  quick\_sorta(a, left, j - 1);  quick\_sorta(a, j + 1, right);  }  }  void quick\_sortd(edatatype a[], int left, int right) {  int pivot;  int i, j;  edatatype tmp;  if (left < right) {  i = left; j = right + 1;  pivot = a[left].escore;  while (i < j) {  do i++;  while ((a[i].escore >= pivot) && (i < right));  do j--;  while ((a[j].escore <= pivot) && (j > left));  if (i < j) {  tmp = a[i];  a[i] = a[j];  a[j] = tmp;  }  }  if (j != left) {  tmp = a[j];  a[j] = a[left];  a[left] = tmp;  }  quick\_sortd(a, left, j - 1);  quick\_sortd(a, j + 1, right);  }  }  void main(int argc, char\* argv[]) {  edatatype edb[50];  FILE\* infile, \* out;  int i = 0, j;  char choice;  char dcode[5];  if (argc != 3) {  printf("명령인수를 잘못 주었습니다. \n");  exit(1);  }  if ((infile = fopen(argv[1], "r")) == NULL) {  printf("입력파일을 열 수 없습니다.\n");  exit(1);  }  if ((out = fopen(argv[2], "w")) == NULL) {  printf("결과 파일을 열 수 없습니다.\n");  exit(1);  }  while (fscanf(infile, "%ld %s %d %s", &(edb[i].eid), edb[i].name, &(edb[i].escore), edb[i].dname) != EOF)  i++;  printf("student.txt : \n");  for (j = 0; j < i; j++)  printf(" %ld\t%s\t%d\t%s\n", (edb[j].eid), edb[j].name, (edb[j].escore), edb[j].dname);  quick\_sorta(edb, 0, i - 1);  printf("\n\nSorted Date....\n");  for (j = 0; j < i; j++)  fprintf(out, "%ld\t%s\t%d\t%s\n", (edb[j].eid), edb[j].name, (edb[j].escore), edb[j].dname);    printf("Sales Department : S110\n");  printf("Research Department : R250\n");  printf("Managemnet Department : M210\n");  printf("\nEnter the Department code : ");  scanf("%s", dcode);  for (j = 0; j < i; j++)  if (strcmp(dcode, edb[j].dname) == 0)  printf("%ld\t%s\t%d\n", (edb[j].eid), edb[j].name, (edb[j].escore));  printf("\n\nFrom the hightest English score To the Lowest . . . \n");  quick\_sortd(edb, 0, i - 1);  for(j=0; j<i; j++)  printf("%ld\t%s\t%d\t%s\n", (edb[j].eid), edb[j].name,(edb[j].escore), edb[j].dname);  } |

**텍스트, 스크린샷, 블랙이(가) 표시된 사진

자동 생성된 설명**

**텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명**

**[예제 2] 합병 정렬 프로그래밍 연습 (1) (p.430)**

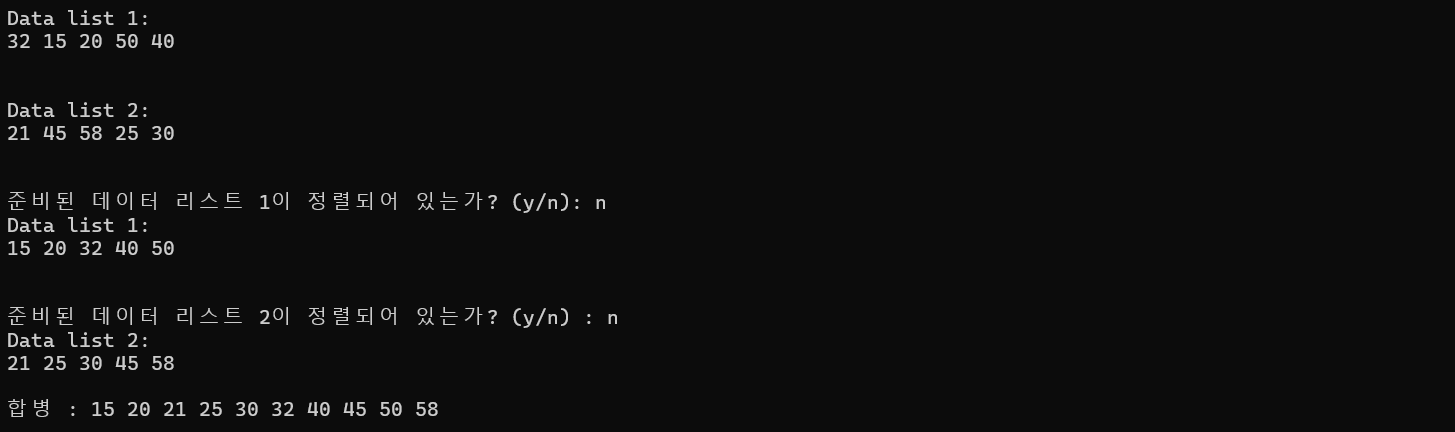
|  |
| --- |
| C |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  #include <stdlib.h>  void print\_list(int\* list, int n, char\* mesg);  int two\_one(int\* a1, int\* a2, int\* a, int n1, int n2) {  int i = 0, j = 0, k = 0;  while (i < n1 && j < n2) {  if (a1[i] <= a2[j])  a[k++] = a1[i++];  else if (a1[i] > a2[j])  a[k++] = a2[j++];  }  if (i == n1)  while (j < n2) a[k++] = a2[j++];  else  while (i < n1) a[k++] = a1[i++];  return k;  }  void print\_list(int\* list, int n, char\* mesg) {  int i;  printf(mesg);  for (i = 0; i < n; i++) {  printf("%d ", list[i]);  if ((i % 10) == 9) printf("\n");  }  printf("\n\n");  }  void main() {  int list1[] = { 10, 20, 30, 40, 50, 60, 70, 77, 80, 90 };  int list2[] = { 25, 27, 35, 38, 40 };  int list3[] = { 15, 45, 47, 57, 59 };  int list4[] = { 1, 2, 5, 9, 60, 62, 65, 67, 69 };  int tlist1[50], tlist2[50], list[100];  int n1, n2, n3, n4, dnum1, dnum2, dnum;  n1 = sizeof(list1) / sizeof(int);  n2 = sizeof(list2) / sizeof(int);  n3 = sizeof(list3) / sizeof(int);  n4 = sizeof(list4) / sizeof(int);  dnum1 = two\_one(list1, list2, tlist1, n1, n2);  print\_list(tlist1, dnum1, "중간 데이터 리스트 1 : \n");  dnum2 = two\_one(list3, list4, tlist2, n3, n4);  print\_list(tlist2, dnum2, "중간 데이터 리스트 2 : \n");  s  dnum = two\_one(tlist1, tlist2, list, dnum1, dnum2);  print\_list(list, dnum, "합병 데이터 리스트 1 : \n");  } |

**텍스트, 스크린샷, 폰트, 블랙이(가) 표시된 사진

자동 생성된 설명**

**[예제 3] 합병정렬 & 퀵정렬 연습**

|  |
| --- |
| C |
| #define \_CRT\_SECURE\_NO\_WARNINGS  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  #define MAX\_SIZE 100  void print\_list(int\* list, int n, char\* mesg);  void quick\_sort(int a[], int left, int right);  void merge(int a[], int left, int mid, int right);  int two\_one(int a1[], int a2[], int a[], int n1, int n2) {  int i = 0, j = 0, k = 0;  while (i < n1 && j < n2) {  if (a1[i] <= a2[j])  a[k++] = a1[i++];  else if (a1[i] > a2[j])  a[k++] = a2[j++];  }  if (i == n1)  while (j < n2) a[k++] = a2[j++];  else  while (i < n1) a[k++] = a1[i++];  return k;  }  void merge(int a[], int left, int mid, int right) {  int i = left, j = mid + 1, k = left, h;  int tlist[MAX\_SIZE];  while (i <= mid && j <= right) {  if (a[i] <= a[j]) {  tlist[k] = a[i];  i++; k++;  }  else {  tlist[k] = a[j];  j++; k++;  }  }  if (i > mid)  while (j <= right) tlist[k++] = a[j++];  else  while (i <= mid) tlist[k++] = a[i++];  for (h = left; h <= right; h++)  a[h] = tlist[h];  }  void print\_list(int a[], int n, char\* mesg) {  int i;  printf(mesg);  for (i = 0; i < n; i++) {  printf("%d ", a[i]);  if ((i % 10) == 9) printf("\n");  }  printf("\n\n");  }  void quick\_sort(int a[], int left, int right) {  int pivot, i, j, tmp;  if (left < right) {  i = left; j = right + 1;  pivot = a[left];  while (i < j) {  do i++;  while ((a[i] <= pivot) && (i < right));  do j--;  while ((a[j] >= pivot) && (j > left));  if (i < j) {  tmp = a[i];  a[i] = a[j];  a[j] = tmp;  }  }  if (j != left) {  tmp = a[j];  a[j] = a[left];  a[left] = tmp;  }  quick\_sort(a, left, j - 1);  quick\_sort(a, j + 1, right);  }  }  void main() {  int list1[] = { 32, 15, 20, 50, 40 };  int list2[] = { 21, 45, 58, 25, 30 };  int list[100];  int n1, n2, n;  char sorted;  n1 = sizeof(list1) / sizeof(int);  n2 = sizeof(list2) / sizeof(int);  print\_list(list1, n1, "\nData list 1: \n");  print\_list(list2, n2, "\nData list 2: \n");  printf("\n준비된 데이터 리스트 1이 정렬되어 있는가? (y/n): ");  sorted = getche();  if (sorted == 'n') quick\_sort(list1, 0, n1 - 1);  printf("\n");  print\_list(list1, n1, "Data list 1: \n");  printf("\n준비된 데이터 리스트 2이 정렬되어 있는가? (y/n) : ");  sorted = getche();  if (sorted == 'n') quick\_sort(list2, 0, n2 - 1);  printf("\n");  print\_list(list2, n2, "Data list 2: \n");  n = two\_one(list1, list2, list, n1, n2);  quick\_sort(list, 0, n - 1);  print\_list(list, n, "합병 : ");  } |

****