**Lab 09**

|  |
| --- |
| 그림입니다. 원본 그림의 이름: YU_UI_RGB-10.png 원본 그림의 크기: 가로 2256pixel, 세로 3047pixel 프로그램 이름 : Adobe ImageReady |

|  |  |
| --- | --- |
| 과목명 | 프로그래밍 언어 |
| 교수님 | 김영탁 교수님 |
| 이 름 | 김주환 |
| 학 번 | 21812158 |
| 일 자 | 2021.05.07.금 |

**9.1 구조체 Date 정의 및 관련 함수 구현**

(1) 연월일 ( int year, int month, int day ) 정보가 포함된 날짜를 나타내기 위하여 구조체 Date를 typedef를 사용하여 정의하라. 구조체 Date는 별도의 헤더파일 “Date.h”에 저장된다.

typedef struct {

int year;

int month;

int day;

} Date;

(2) 구조체 Date 변수를 출력하기 위한 함수 void printDate(Date date)를 Date.cpp 파일에 작성하라. 날짜의 출력 포멧은 년 4자리, 월 2자리, 일 2자리로 출력하며, 월과 일의 10 단위 숫자가 없는 경우 공란으로 출력하라. 년, 월, 일 사이에는 콤마를 출력할 것.

void printDate(Date date) {

printf("(%04d, %2d, %2d)",

date.year, date.month, date.day);

}

(3) 구조체 Date 변수 2개의 크기를 비교하는 함수 int compareDate(Date d1, Date d2)를 Date.cpp 파일에 작성하라. 이 날짜 비교 함수는 년월일을 차례로 비교하여 d1 값이 d2 보 다 크면 1, 같으면 0, 작으면 -1을 반환한다.

int compareDate(Date d1, Date d2) {

if (d1.year > d2.year)

return 1;

else if (d1.year < d2.year)

return -1;

else {

if (d1.month > d2.month)

return 1;

else if (d1.month < d2.month)

return -1;

else {

if (d1.day > d2.day)

return 1;

else if (d1.day < d2.day)

return -1;

else

return 0;

}

}

}

**9.2 구조체 Tel\_Number 정의 및 관련 함수 구현**

(1) 전화번호 정보 (국가 번호 nation\_code, 지역 번호 region\_code, 교환기 번호 switch\_no, 회선 번호 line\_no)를 unsigned short로 포함하는 구조체 Tel\_Number를 typedef를 사용하여 정의하라. 구조체 Tel\_Number는 별도의 헤더파일 “Telephone\_Number.h”에 저장된다.

typedef struct {

U\_SHORT nation\_code;

U\_SHORT region\_no;

U\_SHORT switch\_no;

U\_SHORT line\_no;

} Tel\_Number;

(2) 전화번호를 출력하기 위한 함수 void printTelephoneNumber(Tel\_Number tel\_no)를 Telephone\_Number.cpp 파일에 구현하라. 전화번호의 출력 포멧은 ‘+’ 국가번호 (2자리) - 지역번호 (4자리) - 교환기번호 (4자리) - 회선번호(4자리)로 출력하라. 국가번호, 지역번호, 교환기번호, 회선 번호 사이에는 하이펀 (‘-‘)이 출력되도록 하라. 지역번호, 교환기번호와 회선 번호에서 빈 자리는 0으로 채울 것.

void printTelephoneNumber(Tel\_Number telNo) {

printf("(tel: +%d-", telNo.nation\_code);

printf("%03d-%04d-%04d)", telNo.region\_no, telNo.switch\_no, telNo.line\_no);

}

(3) 2개의 전화번호를 비교하는 함수 int compareTelNumber(Tel\_Number tn1, Tel\_Number tn2)를 구현하라. 전화번호 비교에서는 국가번호, 지역번호, 교환기번호, 회선번호의 순서로 값의 크기를 비교하며, 값이 큰 경우 1, 값이 같은 경우 0, 값이 작은 경우 -1을 반환한다. 만약 상위 단위 (예: 국가번호) 값이 다른 경우 나머지 하위 단위 번호들과는 상관없이 상위 단 위 값에 의하여 비교가 결정된다.

void printTelephoneNumber(Tel\_Number telNo) {

printf("(tel: +%d-", telNo.nation\_code);

printf("%03d-%04d-%04d)", telNo.region\_no, telNo.switch\_no, telNo.line\_no);

}

**9.3 구조체 Student 정의 및 관련 함수 구현**

(1) 학번 (int st\_id), 이름 (char name[MAX\_NAME\_LEN]), 생년윌일(Date date\_of\_birth), 전화번 호(Tel\_Number tel\_no), 학점평균(double GPA) 정보를 가지는 구조체 Student를 typedef를 사용하여 정의하라. 구조체 Student는 별도의 헤더파일 “Student.h”에 저장할 것.

typedef struct {

int st\_id;

char name[MAX\_NAME\_LEN];

Date birth\_date;

Tel\_Number tel\_number;

double GPA; // Grade Point Average

} Student;

(2) 학생의 정보를 출력하기 위한 함수 void printStudent(Student \*pSt)를 구현하라. 구조체 Student에 관련된 함수들의 함수원형은 헤더파일 “Student.h”에 저장할 것.

void printStudent(Student\* pST) {

printf("Student [ID: %08d, %-10s", pST->st\_id, pST->name);

printf(", GPA: %5.2lf", pST->GPA);

printf(", ");

printDate(pST->birth\_date);

printf(", ");

printTelephoneNumber(pST->tel\_number);

printf("]");

}

(3) 학생들의 정보가 포함된 구조체 Student의 배열을 전달받아 포함된 학생들의 정보를 차례 로 출력하는 함수 void printStudents(Student \*stArray, int num\_students)를 구현하라. 이 함 수는 printStudent() 함수를 호출하여 개별 학생 정보를 출력한다.

void printStudents(Student\* stArr, int num) {

Student\* st = stArr;

for (int i = 0; i < num; i++) {

printStudent(st);

printf("\n");

st++;

}

}

(4) 구조체 Student의 배열을 전달받아 배열의 순서를 뒤섞어 주는 함수 void shuffleStudents(Student students[], int num\_students)를 구현하라.

void shuffleStudents(Student stArray[], int num\_students) {

int st\_1, st\_2;

Student st\_temp;

srand(time(0));

for (int i = 0; i < num\_students; i++) {

st\_1 = rand() % num\_students;

st\_2 = rand() % num\_students;

st\_temp = stArray[st\_1];

stArray[st\_1] = stArray[st\_2];

stArray[st\_2] = st\_temp;

}

}

(5) 구조체 Student의 배열을 전달받아 학생들의 성적 GPA의 최대, 최소, 평균 값을 찾고, 계 산하여 출력하는 함수 void statisticsGPA(Student students[], int num\_students)를 구현하라.

void statisticsGPA(Student students[], int num\_students) {

/\* Calculate the statistics of students' GPA : max, min, avg \*/

Student\* pST, \* pST\_max, \* pST\_min;

pST\_max = pST\_min = pST = students;

double GPA\_max, GPA\_min, GPA\_avg, GPA\_sum = 0.0;

GPA\_max = GPA\_min = pST->GPA;

GPA\_sum = pST->GPA;

for (int i = 1; i < num\_students; i++) {

pST = &students[i];

if (GPA\_max < pST->GPA)

GPA\_max = pST->GPA;

if (GPA\_min > pST->GPA)

GPA\_min = pST->GPA;

GPA\_sum += pST->GPA;

} // end for

GPA\_avg = GPA\_sum / (double)num\_students;

printf("GPA\_max (%5.2lf), GPA\_min (%5.2lf), GPA\_avg (%5.2lf)\n",

GPA\_max, GPA\_min, GPA\_avg);

}

(6) 구조체 Student의 배열을 전달받아 학생들의 성적 GPA 중 가장 높은 성적의 학생이 저장 되어 있는 주소를 반환하는 함수 Student \*searchBestGPAStudent(Student students[], int num\_students)를 구현하라.

Student\* searchBestGPAStudent(Student students[], int num) {

/\* Search the student with highest GPA \*/

Student\* pST, \* pST\_max;

pST\_max = pST = students;

double GPA\_max;

GPA\_max = pST->GPA;

for (int i = 1; i < num; i++) {

pST = &students[i];

if (GPA\_max < pST->GPA) {

GPA\_max = pST->GPA;

pST\_max = pST;

}

}

return pST\_max;

}

(7) 구조체 Student의 배열을 전달받아 학생들의 성적 GPA 중 가장 낮은 성적의 학생이 저장되어 있는 주소를 반환하는 함수 Student \*searchWorstGPAStudent(Student students[], int num\_students)를 구현하라.

Student\* searchWorstGPAStudent(Student students[], int num) {

/\* Search the student with highest GPA \*/

Student\* pST, \* pST\_min;

pST\_min = pST = students;

double GPA\_min;

GPA\_min = pST->GPA;

for (int i = 1; i < num; i++) {

pST = &students[i];

if (GPA\_min > pST->GPA) {

GPA\_min = pST->GPA;

pST\_min = pST;

}

}

return pST\_min;

}

(8) 구조체 Student의 배열을 전달받아 학번 (st\_id)의 오름 차순으로 정렬하는 함수sortStudent\_by\_ST\_ID(Student students[], int num\_students)를 구현하라.

void sortStudents\_by\_ST\_ID(Student students[], int num) {

/\* Selection Sorting student array by ST\_ID in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_id\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_id\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (pST\_min->st\_id > pST->st\_id) {

pST\_min = pST;

st\_id\_min = j;

}

} // end for

if (st\_id\_min != i) {

temp = students[st\_id\_min];

students[st\_id\_min] = students[i];

students[i] = temp;

}

}

}

(9) 구조체 Student의 배열을 전달받아 성적 평점 (GPA)의 내림 차순으로 정렬하는 함수sortStudent\_by\_GPA(Student students[], int num\_students)를 구현하라.

void sortStudents\_by\_GPA(Student students[], int num) {

/\* Selection Sorting student array by GPA in decreasing order \*/

Student\* pST, \* pST\_max;

Student temp;

int st\_max\_id;

for (int i = 0; i < num; i++) {

pST\_max = pST = &students[i];

st\_max\_id = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (pST\_max->GPA < pST->GPA) {

pST\_max = pST;

st\_max\_id = j;

}

} // end for

if (st\_max\_id != i) {

temp = students[st\_max\_id];

students[st\_max\_id] = students[i];

students[i] = temp;

}

}

}

(10) 구조체 Student의 배열을 전달받아 이름 (name)의 오름 차순으로 정렬하는 함수sortStudent\_by\_name(Student students[], int num\_students)를 구현하라.

void sortStudents\_by\_name(Student students[], int num) {

/\* Selection Sorting student array by name in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_name\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_name\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (strcmp(pST->name, pST\_min->name) < 0) {

pST\_min = pST;

st\_name\_min = j;

}

} // end inner for

if (st\_name\_min != i) {

temp = students[st\_name\_min];

students[st\_name\_min] = students[i];

students[i] = temp;

}

} // end outer for

}

(11) 구조체 Student의 배열을 전달받아 생년월일 (date\_of\_birth)의 오름 차순으로 정렬하는 함수sortStudent\_by\_BirthDate(Student students[], int num\_students)를 구현하라.

void sortStudents\_by\_BirthDate(Student students[], int num) {

/\* Selection Sorting student array by BirthDate in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_date\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_date\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (compareDate(pST->birth\_date, pST\_min->birth\_date) < 0) {

pST\_min = pST;

st\_date\_min = j;

}

}

if (st\_date\_min != i) {

temp = students[st\_date\_min];

students[st\_date\_min] = students[i];

students[i] = temp;

}

}

}

(12) 구조체 Student의 배열을 전달받아 전화번호 (tel\_no)의 오름 차순으로 정렬하는 함수sortStudent\_by\_TelNumber(Student students[], int num\_students)를 구현하라.

void sortStudents\_by\_TelNumber(Student students[], int num) {

/\* Selection Sorting student array by TelNo in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_telno\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_telno\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (compareTelNumber(pST->tel\_number, pST\_min->tel\_number) < 0) {

pST\_min = pST;

st\_telno\_min = j;

}

}

if (st\_telno\_min != i) {

temp = students[st\_telno\_min];

students[st\_telno\_min] = students[i];

students[i] = temp;

}

}

}

// main.cpp(2)

switch (menu)

{

case 1:

printStudents(students, num\_students);

printf("\n");

break;

case 2:

statisticsGPA(students, num\_students);

break;

case 3:

pST\_GPA\_max = searchBestGPAStudent(students, num\_students);

pST\_GPA\_min = searchWorstGPAStudent(students, num\_students);

printf("Student with best GPA : ");

printStudent(pST\_GPA\_max);

printf("\n");

printf("Student with worst GPA : ");

printStudent(pST\_GPA\_min);

printf("\n");

break;

case 4:

sortStudents\_by\_ST\_ID(students, num\_students);

printf("After sorting students by increasing order of student ID:\n");

printStudents(students, num\_students);

printf("\n");

break;

case 5:

sortStudents\_by\_GPA(students, num\_students);

printf("After sorting students by decreasing order of GPA:\n");

printStudents(students, num\_students);

printf("\n");

break;

case 6:

sortStudents\_by\_name(students, num\_students);

printf("After sorting students by increasing order of student name:\n");

printStudents(students, num\_students);

printf("\n");

break;

case 7:

sortStudents\_by\_BirthDate(students, num\_students);

printf("After sorting students by increasing order of student birth date:\n");

printStudents(students, num\_students);

printf("\n");

break;

case 8:

sortStudents\_by\_TelNumber(students, num\_students);

printf("After sorting students by increasing order of student's telephone number:\n");

printStudents(students, num\_students);

printf("\n");

break;

default:

break;

}

shuffleStudents(students, num\_students); // shuffle array for next experiments

}

}

/\*

\* 파일명 : main.cpp

\* 목적 및 기본 기능

\* 주어진 구초체 데이터를 특성을 확인하고 각 분류에 따라 정렬을 마친 후 실행창에 출력한다.

\* 작성자 : 김주환(21812158)

\* 작성일 : 2021년 5월 7일

\*/

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include "Student.h"

#include "Telephone\_Number.h"

#include "Date.h"

#define MAX\_NUM\_STUDENTS 100

void main() {

int num\_students = 0;

int menu;

Student\* pST\_GPA\_max, \* pST\_GPA\_min;

extern Student students[];

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

{

if (students[i].st\_id == -1)

break;

else

num\_students++;

}

printf("Number of studetns = %d\n", num\_students);

while (1)

{

printf("\n");

printf(" 1 : print student records\n");

printf(" 2 : calculate statistics GPA (max, min, avg) of students' \n");

printf(" 3 : search students of best GPA and worst GPA\n");

printf(" 4 : sort students by student ID\n");

printf(" 5 : sort students by GPA\n");

printf(" 6 : sort students by name\n");

printf(" 7 : sort students by date of birth\n");

printf(" 8 : sort studetns by telephone number\n");

printf("0 : Quit\n");

printf("Input menu = ");

scanf("%d", &menu);

if (menu == 0)

break;

// Student.h

#ifndef STUDENT\_H

#define STUDENT\_H

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include "Student.h"

#include "Date.h"

#include "Telephone\_number.h"

#define MAX\_NAME\_LEN 20

#define NUM\_STUDENTS 10

typedef struct {

int st\_id;

char name[MAX\_NAME\_LEN];

Date birth\_date;

Tel\_Number tel\_number;

double GPA; // Grade Point Average

} Student;

void printStudent(Student\* pST);

void printStudents(Student\* stArr, int num\_students);

void shuffleStudents(Student students[], int num\_students);

void statisticsGPA(Student students[], int num\_students);

Student\* searchBestGPAStudent(Student students[], int num);

Student\* searchWorstGPAStudent(Student students[], int num);

void sortStudents\_by\_GPA(Student students[], int num);

void sortStudents\_by\_ST\_ID(Student students[], int num);

void sortStudents\_by\_name(Student students[], int num);

void sortStudents\_by\_BirthDate(Student students[], int num);

void sortStudents\_by\_TelNumber(Student students[], int num);

#endif

// StudentRecords.cpp

#include "Student.h"

Student students[] =

{

{ 21911000, "Kim, G-M", { 1990, 10, 5 }, { 82, 53, 805, 1234 }, 3.57 },

{ 21822075, "Yoon, S-M", { 1990, 4, 5 }, { 82, 53, 811, 1550 }, 4.37 },

{ 21433015, "Hwang, S-S", { 1989, 1, 10 }, { 82, 53, 817, 1005 }, 2.72 },

{ 21644054, "Lee, K-M", { 1991, 5, 15 }, { 82, 10, 9112, 9876 }, 3.35 },

{ 21255340, "Hong, G-M", { 1990, 2, 5 }, { 82, 55, 810, 5678 }, 3.57 },

{ 21766056, "Zang, S-M", { 1990, 3, 15 }, { 82, 10, 9112, 1600 }, 4.45 },

{ 21177017, "Park, S-S", { 1989, 7, 10 }, { 82, 34, 817, 1098 }, 4.12 },

{ 21588053, "Choi, Y-H", { 1992, 9, 25 }, { 82, 53, 845, 5764 }, 3.85 },

{ 21399019, "Ahn, D-J", { 1988, 10, 3 }, { 82, 31, 817, 1038 }, 3.21 },

{ 21010053, "Kwak, S-B", { 1994, 11, 15 }, { 82, 2, 897, 8778 }, 4.43 },

{ 21021053, "Song, W-B", { 1993, 5, 15 }, { 82, 34, 345, 1234 }, 4.47 },

{ 21133053, "Gong, G-W", { 1992, 6, 25 }, { 82, 55, 543, 4321 }, 2.35 },

{ 20911042, "Bhang, S-H", { 1987, 12, 25 }, { 82, 2, 123, 4567 }, 3.75 },

{-1}

};

// Telephone\_Number.h

#ifndef TELEPHONE\_NUMBER\_H

#define TELEPHONE\_NUMBER\_H

#define U\_SHORT unsigned short

typedef struct {

U\_SHORT nation\_code;

U\_SHORT region\_no;

U\_SHORT switch\_no;

U\_SHORT line\_no;

} Tel\_Number;

void printTelephoneNumber(Tel\_Number telNo);

int compareTelNumber(Tel\_Number tn1, Tel\_Number tn2);

#endif

// Date.h

#ifndef DATE\_H

#define DATE\_H

#include <stdio.h>

#include "Date.h"

typedef struct {

int year;

int month;

int day;

} Date;

void printDate(Date date);

int compareDate(Date d1, Date d2);

#endif

// Telephone\_Number.cpp

#ifndef TELEPHONE\_NUMBER\_H

#define TELEPHONE\_NUMBER\_H

#include <stdio.h>

#include "Telephone\_number.h"

#define U\_SHORT unsigned short

typedef struct

{

U\_SHORT nation\_code;

U\_SHORT region\_no;

U\_SHORT switch\_no;

U\_SHORT line\_no;

} Tel\_Number;

void printTelephoneNumber(Tel\_Number telNo) {

printf("(tel: +%d-", telNo.nation\_code);

printf("%03d-%04d-%04d)", telNo.region\_no, telNo.switch\_no, telNo.line\_no);

}

int compareTelNumber(Tel\_Number tn1, Tel\_Number tn2)

{

if (tn1.nation\_code > tn2.nation\_code)

return 1;

else if (tn1.nation\_code < tn2.nation\_code)

return -1;

else if (tn1.region\_no > tn2.region\_no)

return 1;

else if (tn1.region\_no < tn2.region\_no)

return -1;

else if (tn1.switch\_no > tn2.switch\_no)

return 1;

else if (tn1.switch\_no < tn2.switch\_no)

return -1;

else if (tn1.line\_no > tn2.line\_no)

return 1;

else if (tn1.line\_no < tn2.line\_no)

return -1;

else

return 0;

}

#endif

// Date.cpp

#ifndef DATE\_H

#define DATE\_H

#include <stdio.h>

#include "Date.h"

typedef struct {

int year;

int month;

int day;

} Date;

void printDate(Date date) {

printf("(%04d, %2d, %2d)",

date.year, date.month, date.day);

}

int compareDate(Date d1, Date d2) {

if (d1.year > d2.year)

return 1;

else if (d1.year < d2.year)

return -1;

else {

if (d1.month > d2.month)

return 1;

else if (d1.month < d2.month)

return -1;

else {

if (d1.day > d2.day)

return 1;

else if (d1.day < d2.day)

return -1;

else

return 0;

}

}

}

#endif

// Student.cpp(2)

void statisticsGPA(Student students[], int num\_students) {

/\* Calculate the statistics of students' GPA : max, min, avg \*/

Student\* pST, \* pST\_max, \* pST\_min;

pST\_max = pST\_min = pST = students;

double GPA\_max, GPA\_min, GPA\_avg, GPA\_sum = 0.0;

GPA\_max = GPA\_min = pST->GPA;

GPA\_sum = pST->GPA;

for (int i = 1; i < num\_students; i++) {

pST = &students[i];

if (GPA\_max < pST->GPA)

GPA\_max = pST->GPA;

if (GPA\_min > pST->GPA)

GPA\_min = pST->GPA;

GPA\_sum += pST->GPA;

} // end for

GPA\_avg = GPA\_sum / (double)num\_students;

printf("GPA\_max (%5.2lf), GPA\_min (%5.2lf), GPA\_avg (%5.2lf)\n",

GPA\_max, GPA\_min, GPA\_avg);

}

Student\* searchBestGPAStudent(Student students[], int num) {

/\* Search the student with highest GPA \*/

Student\* pST, \* pST\_max;

pST\_max = pST = students;

double GPA\_max;

GPA\_max = pST->GPA;

for (int i = 1; i < num; i++) {

pST = &students[i];

if (GPA\_max < pST->GPA) {

GPA\_max = pST->GPA;

pST\_max = pST;

}

}

return pST\_max;

}

Student\* searchWorstGPAStudent(Student students[], int num) {

/\* Search the student with highest GPA \*/

Student\* pST, \* pST\_min;

pST\_min = pST = students;

double GPA\_min;

GPA\_min = pST->GPA;

for (int i = 1; i < num; i++) {

pST = &students[i];

if (GPA\_min > pST->GPA) {

GPA\_min = pST->GPA;

pST\_min = pST;

}

}

return pST\_min;

}

// Student.cpp

#ifndef STUDENT\_H

#define STUDENT\_H

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

#include <string.h>

#include "Student.h"

#include "Date.h"

#include "Telephone\_number.h"

#define MAX\_NAME\_LEN 20

#define NUM\_STUDENTS 10

typedef struct {

int st\_id;

char name[MAX\_NAME\_LEN];

Date birth\_date;

Tel\_Number tel\_number;

double GPA; // Grade Point Average

} Student;

void printStudent(Student\* pST) {

printf("Student [ID: %08d, %-10s", pST->st\_id, pST->name);

printf(", GPA: %5.2lf", pST->GPA);

printf(", ");

printDate(pST->birth\_date);

printf(", ");

printTelephoneNumber(pST->tel\_number);

printf("]");

}

void printStudents(Student\* stArr, int num) {

Student\* st = stArr;

for (int i = 0; i < num; i++) {

printStudent(st);

printf("\n");

st++;

}

}

void shuffleStudents(Student stArray[], int num\_students) {

int st\_1, st\_2;

Student st\_temp;

srand(time(0));

for (int i = 0; i < num\_students; i++) {

st\_1 = rand() % num\_students;

st\_2 = rand() % num\_students;

st\_temp = stArray[st\_1];

stArray[st\_1] = stArray[st\_2];

stArray[st\_2] = st\_temp;

}

}

// Student.cpp(3)

void sortStudents\_by\_GPA(Student students[], int num) {

/\* Selection Sorting student array by GPA in decreasing order \*/

Student\* pST, \* pST\_max;

Student temp;

int st\_max\_id;

for (int i = 0; i < num; i++) {

pST\_max = pST = &students[i];

st\_max\_id = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (pST\_max->GPA < pST->GPA) {

pST\_max = pST;

st\_max\_id = j;

}

} // end for

if (st\_max\_id != i) {

temp = students[st\_max\_id];

students[st\_max\_id] = students[i];

students[i] = temp;

}

}

}

void sortStudents\_by\_ST\_ID(Student students[], int num) {

/\* Selection Sorting student array by ST\_ID in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_id\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_id\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (pST\_min->st\_id > pST->st\_id) {

pST\_min = pST;

st\_id\_min = j;

}

} // end for

if (st\_id\_min != i) {

temp = students[st\_id\_min];

students[st\_id\_min] = students[i];

students[i] = temp;

}

}

}

void sortStudents\_by\_name(Student students[], int num) {

/\* Selection Sorting student array by name in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_name\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_name\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (strcmp(pST->name, pST\_min->name) < 0) {

pST\_min = pST;

st\_name\_min = j;

}

} // end inner for

if (st\_name\_min != i) {

temp = students[st\_name\_min];

students[st\_name\_min] = students[i];

students[i] = temp;

}

} // end outer for

}

// Student.cpp(4)

void sortStudents\_by\_BirthDate(Student students[], int num) {

/\* Selection Sorting student array by BirthDate in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_date\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_date\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (compareDate(pST->birth\_date, pST\_min->birth\_date) < 0) {

pST\_min = pST;

st\_date\_min = j;

}

}

if (st\_date\_min != i) {

temp = students[st\_date\_min];

students[st\_date\_min] = students[i];

students[i] = temp;

}

}

}

void sortStudents\_by\_TelNumber(Student students[], int num) {

/\* Selection Sorting student array by TelNo in increasing order \*/

Student\* pST, \* pST\_min;

Student temp;

int st\_telno\_min;

for (int i = 0; i < num; i++) {

pST\_min = pST = &students[i];

st\_telno\_min = i;

for (int j = i + 1; j < num; j++) {

pST = &students[j];

if (compareTelNumber(pST->tel\_number, pST\_min->tel\_number) < 0) {

pST\_min = pST;

st\_telno\_min = j;

}

}

if (st\_telno\_min != i) {

temp = students[st\_telno\_min];

students[st\_telno\_min] = students[i];

students[i] = temp;

}

}

}

#endif

Oral Test

|  |
| --- |
| **(1) 구조체 배열을 동적으로 생성하는 절차를 예를 들어 설명하라.**  구조체 배열을 동적으로 생성하는 절차의 골격은 일반적인 배열의 동적 할당과 크게 다르지 않다.  (구조체 이름)\* (변수 이름) = (구조체 자료형)calloc((메모리 개수), (메모리 크기));   * 자료형 메모리의 자료형은 지정된 구조체의 자료형을 사용한다. 해당 구조체 내부에 여러 자료형이 존재할 수 있지만 이 구조체 자체로 모두를 표현하는 것이므로 지정된 구조체의 자료형을 사용해준다. * 메모리 크기와 개수 메모리의 크기는 지정된 구조체에 sizeof() 함수를 사용하여 구해준다. 메모리의 개수는 기존 동적할당과 다르지 않다. |

|  |
| --- |
| **(2) 구조체를 포인터로 가리키게 한 후, 구조체의 각 항목을 포인터를 사용하여 읽거나 쓸 수 있게 하는 방법에 대하여 예를 들어 설명하라.**  기본적인 방법으로는 (\*pST).member와 같은 형식으로 구조체의 각 항목을 포인터를 사용하여 읽거나 쓸 수 있다. 하지만 Arrow Operator를 사용하면 pST -> member와 같은 형식으로 구조체의 각 항목의 값을 읽거나 쓸 수 있다. |

|  |
| --- |
| **(3) 비트 단위 구조체를 어떻게 선언하며, 비트 단위 구조체를 사용하면 어떤 장점이 있는가 에 대하여 예를 들어 설명하라**  비트 단위 구조체는 아래와 같은 형식으로 선언할 수 있다.  typedef struct {  char f1:3, f2:3, f3:2;  char c;  short s1:3, s2:3, s3:2, s4:3, s5:2, s6:3;  } BFU;  이는 해당 자료형 내부에서 더 작은 비트 단위로 나누어 선언하는 것을 말한다.  위와 같이 구조체를 비트 단위로 선언하면 메모리를 절약하는데 많은 도움이 된다. |

|  |
| --- |
| **(4) 4바이트 크기의 정수형 (integer) 데이터를 저장할 때 CPU 종류에 따라 Little Endian과 Big Endian에 따라 서로 다르게 저장되는 byte ordering (바이트 저장 순서)의 차이점에 대하여 예를 들어 설명하라.**  여러 바이트로 표현되는 데이터를 메모리에 저장할 경우, CPU에 따라서 바이트 저장 순서가 다르다.   * Big Edian System : 시작(기준)주소에 MSB(Big End) 저장 * Little Edian Systen : 시작(기준)주소에 LSB(Little End) 저장   하나의 컴퓨터에서만 데이터를 저장하고 사용하는 경우에는 문제가 없지만, 바이트 저장 순서를 고려하지 않고 인터넷을 통해 다른 PC로 데이터를 전달하는 경우, 송신 측과 수신 측의 바이트 저장 순서 체계가 다르면 문제가 발생된다.  이는 인터넷으로 데이터를 전송할 때 항상 Big Endian 체계로 변환시켜 전송하면 해결 가능하다. |