리스트 구조를 이용한 학생관리 프로그램

20196014 고지흔

*#define* \_CRT\_SECURE\_NO\_WARNINGS

*#include* <stdio.h>

*#include* <stdlib.h>

*#include* <string.h>

*// 학생 정보 구조체*

typedef struct student

{

    int     id;

    char    name[20];

    int     kor, math, eng, com;

} student;

*// 이중 연결 리스트 구조체*

typedef struct Dnode

{

    struct Dnode    \*llink; *// 왼쪽 노드를 가리키는 포인터*

    student         data;

    struct Dnode    \*rlink; *// 오른쪽 노드를 가리키는 포인터*

} Dnode;

*// 이중 연결 리스트 초기화 함수*

void init(Dnode \*head)

{

*if* (head == NULL)

*return*; *//null guard*

    head->llink = head;

    head->rlink = head;

}

*// 이중 연결 리스트에 노드 삽입 함수*

void dinsert\_node(Dnode \*before, Dnode \*new\_node)

{

*if* (before == NULL || new\_node == NULL)

*return*;*//guard*

    new\_node->rlink = before->rlink;

    new\_node->llink = before;

    before->rlink->llink = new\_node;

    before->rlink = new\_node;

}

*// 학생 번호로 검색하여 정보 출력하는 함수*

void search(Dnode\* head, student data)

{

*if* (head == NULL)

*return*;*//guard*

    Dnode\* p;

*for* (p = head->rlink; p != head; p = p->rlink)

    {

*if* (p->data.id == data.id)

        {

            printf("%6d %10s %6d %6d %6d %6d\n",p->data.id, p->data.name, p->data.kor, p->data.math, p->data.eng, p->data.com);

*return*;

        }

    }

    printf("%d번 학생의 번호 검색 실패\n\n", data.id);

}

*// 학생 번호를 기준으로 오름차순으로 정렬하여 이중 연결 리스트에 삽입하는 함수*

void sort\_dinsert(Dnode\* head)

{

*if* (head == NULL)

*return*; *//gurad*

    Dnode\* p, \* q;

    student tmp;

*for* (p = head->rlink; p->rlink != head;p = p->rlink)

    {

*for* (q = p->rlink; q != head; q = q->rlink)

        {

*if* (q->data.id < p->data.id)

            {

                tmp = p->data;

                p->data = q->data;

                q->data = tmp;

            }

        }

    }

}

*// 이중 연결 리스트에 저장된 학생 정보를 출력하는 함수*

void display(Dnode\* phead)

{

*if* (phead == NULL)

*return*; *//gurad*

    printf("=====================================================\n");

    printf("| 번호 |   이름   | 언어 | 수리 | 영어 | 컴퓨터 |\n");

    printf("=====================================================\n");

*for* (Dnode\* p = phead->rlink; p != phead; p = p->rlink)

        printf("|%3d  | %9s | %4d | %4d | %4d |  %4d |\n",p->data.id, p->data.name, p->data.kor, p->data.math, p->data.eng, p->data.com);

    printf("=====================================================\n");

}

*// 이중 연결 리스트에 동적 할당된 메모리를 해제하는 함수*

void free\_node(Dnode\* phead)

{

*if* (phead == NULL)

*return*;

    Dnode\* p = phead->rlink;

    Dnode\* next;

*while* (p != phead)

    {

        next = p;

        free(p);

        p = p->rlink;

    }

    free(p); *// head free*

}

*// 학생 이름을 기준으로 이중 연결 리스트에서 노드를 삭제하는 함수*

void del\_Dnode(Dnode \*phead, char\* studentname)

{

*if* (phead == NULL)

*return*;*//guard*

*for* (Dnode\* p = phead->rlink; p != phead; p = p->rlink)

    {

*if* (!strcmp(p->data.name, studentname))

        {

            p->llink->rlink = p->rlink;

            p->rlink->llink = p->llink;

            free(p);

            printf("%s 삭제완료!\n\n", studentname);

*return*;

        }

    }

    printf("%s라는 학생은 존재하지 않습니다\n\n", studentname);

}

int main(void)

{

    FILE\*   fp;

    Dnode\*  head = malloc(sizeof(Dnode)); *// head 노드 메모리 동적 할당*

    Dnode\*  tmp;

    student student\_tmp;

    int     flag;

    init(head); *// head 노드 초기화*

    fp= fopen("data.txt", "rt");

*if* (fp == NULL)

    {

        printf("File not found\n");

*return* 0;

    }

*// 파일에서 학생 정보를 읽어와서 이중 연결 리스트에 삽입*

*while* (!feof(fp))

    {

        fscanf(fp, "%d %s %d %d %d %d",&student\_tmp.id, student\_tmp.name, &student\_tmp.kor, &student\_tmp.math, &student\_tmp.eng, &student\_tmp.com);

        printf("%6d %10s %6d %6d %6d %6d\n",student\_tmp.id, student\_tmp.name, student\_tmp.kor, student\_tmp.math, student\_tmp.eng, student\_tmp.com);

        tmp = (Dnode\*)malloc(sizeof(Dnode));

        tmp->data = student\_tmp;

        dinsert\_node(head, tmp);

    }

*// 메뉴 기반으로 학생 정보 검색, 추가, 정렬, 삭제 기능 수행*

*while* (1)

    {

        printf("\n종료(0) 학생 데이터 입력(1) 학생 검색(2) 목록 보기(3) 삭제(4)\n");

        printf("메뉴 입력 : ");

        scanf("%d", &flag);

*if* (flag==0)

*break*;

*else* *if* (flag==1)

        {

            tmp= (Dnode\*)malloc(sizeof(Dnode));

            printf("추가할 학생 번호 : ");

            scanf("%d", &student\_tmp.id);

            printf("이름 : ");

            scanf("%s", student\_tmp.name);

            printf("국어 : ");

            scanf("%d", &student\_tmp.kor);

            printf("수학 : ");

            scanf("%d", &student\_tmp.math);

            printf("영어 : ");

            scanf("%d", &student\_tmp.eng);

            printf("컴퓨터 : ");

            scanf("%d", &student\_tmp.com);

            tmp->data = student\_tmp;

            dinsert\_node(head, tmp);

        }

*else* *if* (flag==2)

        {

            printf("검색할 학생 번호: ");

            scanf("%d", &student\_tmp.id);

            search(head, student\_tmp);

        }

*else* *if* (flag==3)

        {

            printf("\n<정렬된 목록 보기>\n\n");

            sort\_dinsert(head);

            display(head);

        }

*else* *if* (flag==4)

        {

            char studentname[20];

            printf("삭제할 학생 이름 : ");

            scanf("%s", studentname);

            del\_Dnode(head, studentname);

        }

*else*

            printf("잘못된 입력입니다\n");

    }

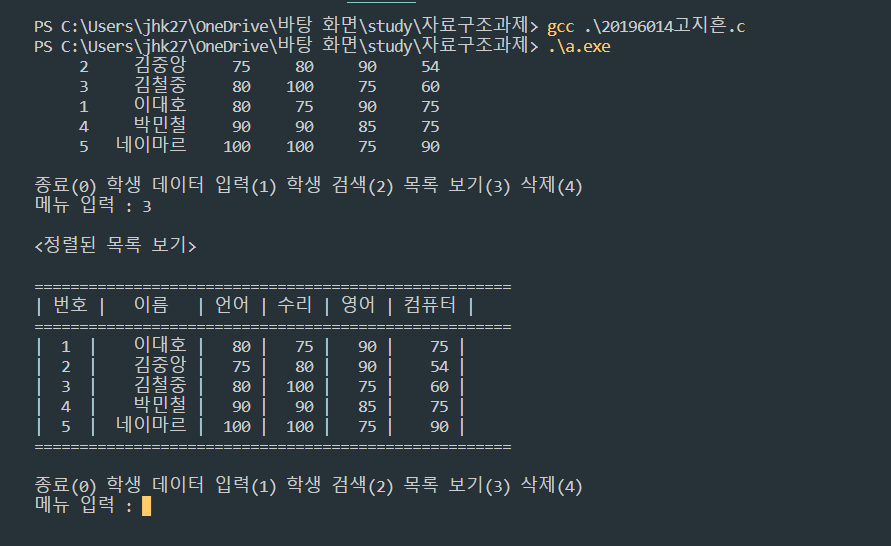
    free\_node(head); *// 동적 할당된 메모리 해제*

    fclose(fp); *//파일포인터 닫기*

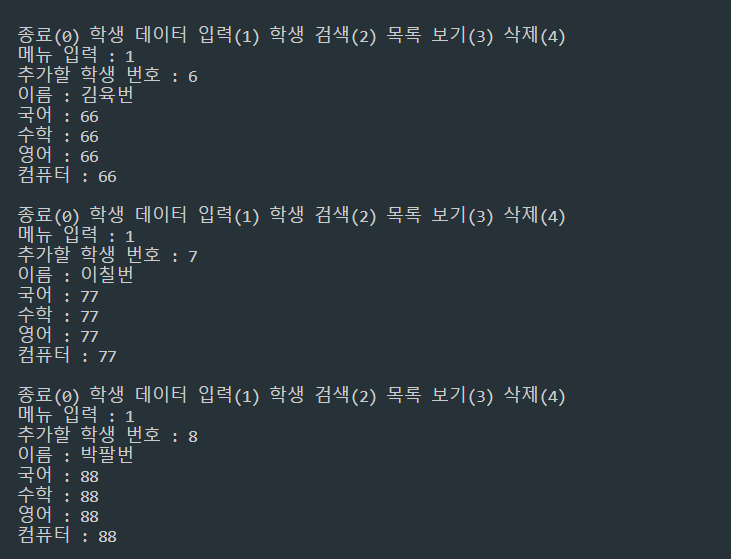
*return* (0);

}

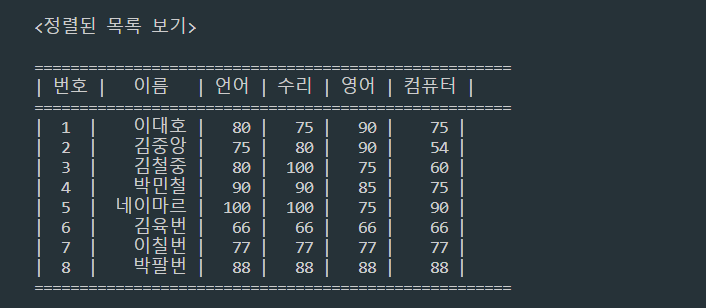
실행 직후 목록보기



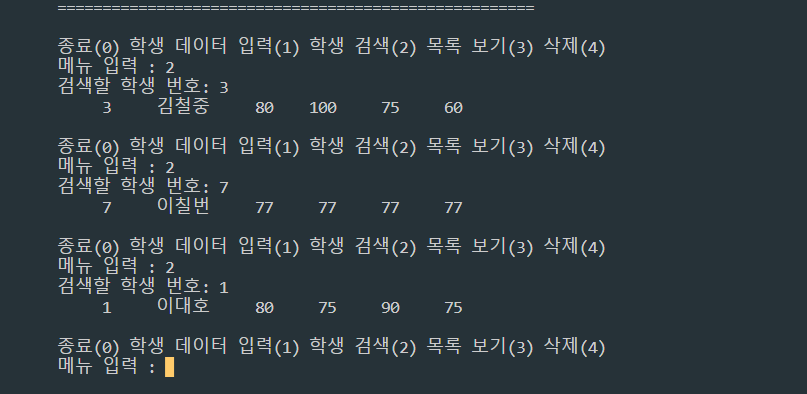
학생 3명 추가



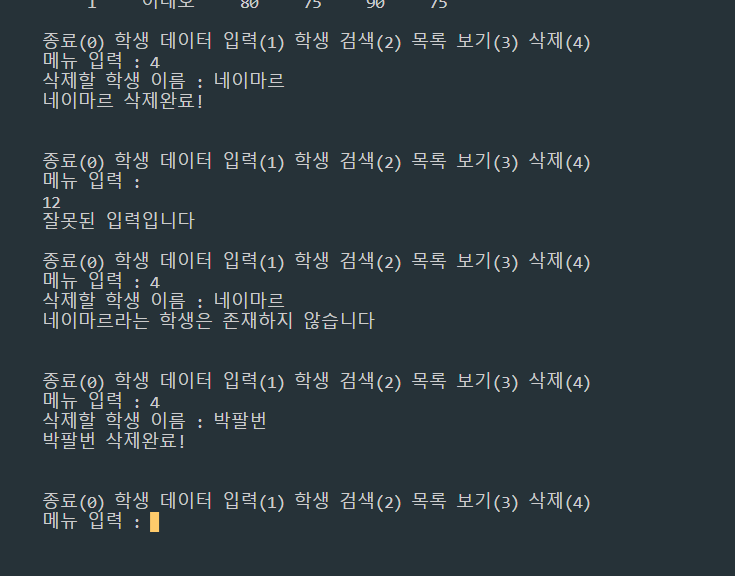
추가 후 목록보기



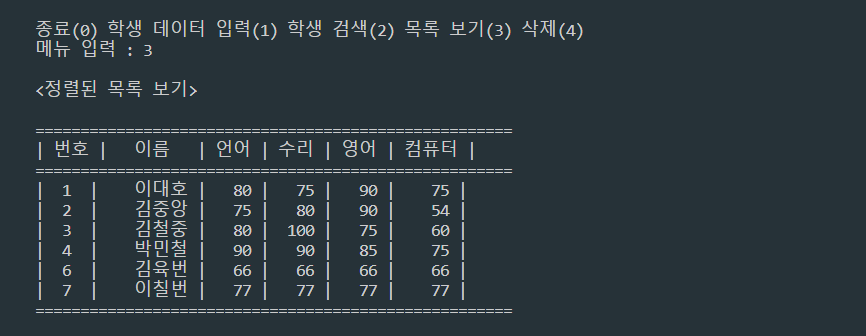
검색 기능 작동 확인



학생 2명 삭제 및 없는 학생 입력 시 알림 확인



삭제 후 목록



정상 종료

