#problem 1 코드

```
typedef struct STUDENT student;


void *print_student_list(student *stu,int n)
{ int i,j;
        student tmp;
        for (i=0 ;i<n-1 ; i++)
        {
                for(j=0;j<n-1;j++)
                {
                        if(stu[j].sem>stu[j+1].sem)
                        {
                                tmp = stu[j];
                                stu[j]=stu[j+1];
                                stu[j+1]=tmp;
                        }
                }
        }
        printf("번호₩t이름₩t학번₩t학기₩t수강과목₩n");
        for(i=0;i<n;i++)
        {
                printf("%d
번 : %s₩t%d₩t %d₩t%s₩n",(i+1),stu[i].name,stu[i].id,stu[i].sem,stu[i].sub);
```

```
        }
}
```

#problem 1 작성과정

1. main에서 만든 stu 배열과 개수를 넘겨받음
2. student 구조체에 tmp를 만들고, stu 배열을 for문을 통해 비교하여 정렬
3. 번호 이름 학번 학기 수강과목 출력
4. for문을 통해 stu 배열의 각각 항목 출력

#problem 2 코드

```
void *find_student(student *stu,int n)
{ print_student_list(stu,n);
        char target[20];
   char stop[]="exit";

        while(1)
        {
                printf("찾으려는 학생의 이름 또는 수강과목을 입력하세요 : ");
                scanf("%s",&target);
                if(strcmp(target,stop)==0)
                {
                        break;
                }
                int i;
```

```
                int count=0;
                for (i=0;i<n-1;i++)
                {
                        if(strcmp(stu[i].name,target)==0)
                        {

        printf("%s₩t%d₩t %d₩t%s₩n",stu[i].name,stu[i].id,stu[i].sem,stu[i].sub);
                                count=count+1;
                        }
                        else if(strcmp(stu[i].sub,target)==0)
                        {

        printf("%s₩t%d₩t %d₩t%s₩n",stu[i].name,stu[i].id,stu[i].sem,stu[i].sub);
                                count=count+1;
                        }
                        else if((strcmp(stu[i].name,target)!=0) && (strcmp(stu[i].sub,target)!=0))
                        { if(i==n-2 && count==0)
                                printf("해당 학생 정보를 찾을 수 없습니다₩n");
                        }
                }
        }
  printf("프로그램 종료.");
}
```

#problem 2 작성과정

1. print_student_list(stu,n) 을 통해 stu 정렬 (단 위의 코드에서 printf는 삭제 – 학생 전체에 대해 출력해줄 필요는 없기 때문에
2. stop이라는 배열에 exit을 넣어주기
3. while반복문을 통해 exit을 입력할 때 까지 반복
4. strcmp를 통해 exit을 입력하면 break;를 만나 반복문 탈출 -> printf("프로그램 종료.");
5. exit이 아니라면 입력한 학생 이름 또는 수강과목을 for문을 통해 stu배열과 strcmp를 통해 비교
6. 과목이나 학생이름이 같은 것만 출력
7. 만약 이름과 수강과목이 존재하지 않으면 prinft실행 (그러나, "해당 학생 정보를 찾을 수 없습니다" 를 한번만 출력해주기 위해 if를 통해 조건 추가)

#probelm 3 코드

```
void *print_presentation_list(student*stu,int n)

{

student p_stu[n];

int count=0;



while(1)

    {

    bool isDup=false;

    int button;

    printf("발표리스트에 추가하려면 1번, 삭제하려면 2번, 종료하려면 3번을 누르세요 : ");

    scanf("%d",&button);

    if (button==1)
```

```
{
    int select;
    printf("발표리스트에 추가할 학생 번호를 입력하세요 : ");
    scanf("%d",&select);
    if(select>n)
    {
        printf("잘못된 번호입니다.₩n");
        continue;
    }
     for(int i=0;i<count;i++)
     {
        int b = stu[select-1].id;
        if(p_stu[i].id==b)
        {
        printf("이미 발표리스트에 있는 학생입니다₩n");
        isDup=true;
        }
      }

      if(isDup==true)
      {
      continue;
      }
```

```
    count = count+1;
    p_stu[count-1] = stu[select-1];
    printf("<<발표리스트>>₩n");
    for(int i=0;i<count;i++)
    {
    printf("%d번 : %s₩t%d₩t %d₩t%s₩n",i+1,p_stu[i].name,p_stu[i].id,p_stu[i].sem,p_stu[i].sub);
    }
    printf("총 발표명 수 : %d명₩n",count);

    }
if (button==2)
{
    printf("발표리스트에서 삭제할 학생 번호를 입력하세요 : ");
    int del_tar=0;
    scanf("%d",&del_tar);
    if(del_tar>count)
    {
        printf("삭제할 학생이 없습니다₩n");
        continue;
    }
    if(del_tar==1){
        for(int i=1;i<count;i++)
```

```
        {
            p_stu[i-1]=p_stu[i];
        }
    }
    else
        p_stu[del_tar-1]=p_stu[del_tar];
    count=count-1;
    printf("<<발표리스트>>₩n");
      for(int i=0;i<count;i++)
        {
      printf("%d번 : %s₩t%d₩t %d₩t%s₩n",i+1,p_stu[i].name,p_stu[i].id,p_stu[i].sem,p_stu[i].sub);
      }
        printf("총 발표명 수 : %d명₩n",count);
}
if (button==3)
{
    printf("종료합니다.");
    break;
}
if(count==5)
{
    printf("발표리스트에 5명이 추가되어 자동으로 프로그램이 종료됩니다");
    break;
```

```
    }
 }
}
```

#problem 3 작성과정

1. Student 구조체에 p_stu라는 새로운 배열 만들기 (발표리스트를 위해)
2. While(1)을 통해 4를 입력할 때까지 반복
3. button이라는 변수에 1번, 2번, 3번 중 선택한 숫자 넣기
4. 1을 선택했다면 발표리스트에 추가할 학생 번호를 입력받아 select에 넣기
5. 입력받은 숫자가 stu에 있는 학생수보다 큰 수라면 잘못된 번호입니다 출력 후 continue로 뒤의 코드 생략 후 다시 인접한 반복문으로 돌아가기
6. 잘못된 번호가 아니라면 for문을 통해 선택한 학생의 id(stu에 있는)와 발표리스트에 있는 학생의 id와(p_stud에 있는) 비교하여 같은 번호가 있다면 이미 발표리스트에 있는 학생입니다 출력 (id가 그 학생의 고유번호이기 때문에 비교하기에 적합)
7. 이미 발표리스트에 있다면 반복문 초기에 선언한 bool형의 isDup변수를 true로 바꿔준 뒤 true이면 continue;를 만나 뒤의 코드를 생략하고 다시 while 반복문으로 돌아가게 함
8. 만약 발표리스트에 없다면 count=count+1을 통해 발표명 수를 추가해주고, 발표리스트에 학생정보 추가하기
9. for문을 통해 발표리스트로 저장한 p_stu 출력, 총 발표명 수 count 출력
10. 2를 선택했다면 del_tar을 통해 삭제하고 싶은 번호를 입력받은 뒤 발표리스트에 있는 번호보다 크다면 삭제할 학생이 없습니다 출력
11. 삭제할 학생이 있다면 삭제하고 싶은 학생의 줄에 그 뒷 학생의 정보를 넣어주면서 삭제된 것처럼 보이게 함
12. for문을 통해 p_stu출력
13. 3번을 입력 시 break를 만나 프로그램 종료
14. 총 발표명 수인 count가 5가 되면 break;를 만나 반복문 탈출 및 프로그램 종료

전체 소스코드

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
struct STUDENT{
    char name[10];
    int id;
    int sem;
    char sub[7];
};
typedef struct STUDENT student;



void *find_student(student *stu,int n)
{
        char target[20];
  char stop[]="exit";

        while(1)
        {
                printf("찾으려는 학생의 이름 또는 수강과목을 입력하세요 : ");
                scanf("%s",&target);
                if(strcmp(target,stop)==0)
```

```
        {
                break;
        }
        int i;
        int count=0;
        for (i=0;i<n-1;i++)
        {
                if(strcmp(stu[i].name,target)==0)
                {

printf("%s₩t%d₩t %d₩t%s₩n",stu[i].name,stu[i].id,stu[i].sem,stu[i].sub);
                        count=count+1;
                }
                else if(strcmp(stu[i].sub,target)==0)
                {

printf("%s₩t%d₩t %d₩t%s₩n",stu[i].name,stu[i].id,stu[i].sem,stu[i].sub);
                        count=count+1;
                }
                else if((strcmp(stu[i].name,target)!=0) && (strcmp(stu[i].sub,target)!=0))
                { if(i==n-2 && count==0)
                        printf("해당 학생 정보를 찾을 수 없습니다₩n");
                }
        }
```

```
            }
  printf("2번 프로그램 종료.₩n");
}

void *print_student_list(student *stu,int n)
{ int i,j;
       student tmp;
     for (i=0 ;i<n-1 ; i++)
     {
          for(j=0;j<n-1;j++)
          {
               if(stu[j].sem>stu[j+1].sem)
               {
                    tmp = stu[j];
                    stu[j]=stu[j+1];
                    stu[j+1]=tmp;
               }
          }
     }
     printf("번호₩t이름₩t학번₩t학기₩t수강과목₩n");
     for(i=0;i<n;i++)
     {
          printf("%d번 : %s₩t%d₩t %d₩t%s₩n",(i+1),stu[i].name,stu[i].id,stu[i].sem,stu[i].sub);
```

```
    }


}


void *print_presentation_list(student*stu,int n)
{
student p_stu[n];
int count=0;


while(1)
    {
    bool isDup=false;
    int button;
    printf("발표리스트에 추가하려면 1번, 삭제하려면 2번, 종료하려면 3번을 누르세요 : ");
    scanf("%d",&button);
    if (button==1)
     {
         int select;
         printf("발표리스트에 추가할 학생 번호를 입력하세요 : ");
         scanf("%d",&select);
         if(select>n)
         {
             printf("잘못된 번호입니다.₩n");
```

```
        continue;
    }
     for(int i=0;i<count;i++)
     {
        int b = stu[select-1].id;
        if(p_stu[i].id==b)
        {
        printf("이미 발표리스트에 있는 학생입니다₩n");
        isDup=true;
        }
      }

      if(isDup==true)
      {
      continue;
      }

count = count+1;
p_stu[count-1] = stu[select-1];
printf("<<발표리스트>>₩n");
for(int i=0;i<count;i++)
{
printf("%d번 : %s₩t%d₩t %d₩t%s₩n",i+1,p_stu[i].name,p_stu[i].id,p_stu[i].sem,p_stu[i].sub);
```

```
    }
    printf("총 발표명 수 : %d명₩n",count);


    }
if (button==2)
{
    printf("발표리스트에서 삭제할 학생 번호를 입력하세요 : ");
    int del_tar=0;
    scanf("%d",&del_tar);
    if(del_tar>count)
    {
        printf("삭제할 학생이 없습니다₩n");
        continue;
    }
    if(del_tar==1){
        for(int i=1;i<count;i++)
        {
            p_stu[i-1]=p_stu[i];
        }
    }
    else
        p_stu[del_tar-1]=p_stu[del_tar];
    count=count-1;
```

```
        printf("<<발표리스트>>₩n");
           for(int i=0;i<count;i++)
            {
           printf("%d번 : %s₩t%d₩t %d₩t%s₩n",i+1,p_stu[i].name,p_stu[i].id,p_stu[i].sem,p_stu[i].sub);
           }
            printf("총 발표명 수 : %d명₩n",count);
    }
    if (button==3)
    {
        printf("종료합니다.");
        break;
    }
    if(count==5)
    {
        printf("발표리스트에 5명이 추가되어 자동으로 프로그램이 종료됩니다");
        break;
    }
 }
}



int main()
{
```

```
student stu[]={

{"Minsu",2021001,2,"class1"},{"Ayoung",2021015,2,"class2"},{"Jihoon",2018016,8,"class5"},{"Minsu",20
20013,4,"class5"},{"Sojung",2021033,2,"class3"},{"Eunmi",2019010,3,"class4"},{"Eunmi",2018022,5,"clas
s1"},{"Heejoon",2018001,7,"class4"}};

int n;

n=sizeof(stu)/sizeof(student);

find_student(stu,n);

print_student_list(stu,n);

print_presentation_list(stu,n);

}
```

최종코드 작성과정

1. STUDENT라는 구조체를 만들어주고 main함수 내에서 내용 넣기
2. 변수 n에 구조체 배열의 크기 넣기
3. Find_student함수를 실행하기 (stu배열과 n을 넘겨줌) – 이에 해당하는 내용은 위에 기술한 것과 같음
4. Print_student_list(stu,n)함수 실행하기 - 이에 해당하는 내용은 위에 기술한 것과 같음
5. Print_student_list에서 exit을 입력하면 이 함수 종료하고 뒤이어 print_presentation_list 함수 실행