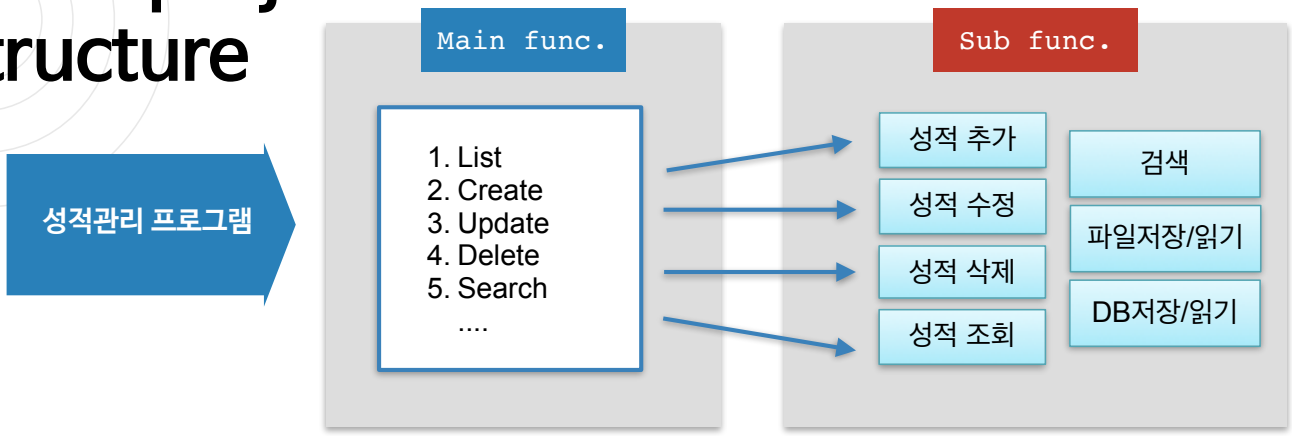
3

CRUD
(C Programming)

CRUD

- 데이터 모델(구조)을 다루기 위한 기본적인 처리 기능
- Create(추가), Read(조회), Update(수정), Delete(삭제)
- 추가적으로 User Interface(UI), 검색, 정렬 기능이 필요함
- 데이터 저장기능을 위해 파일이나 DB를 사용하여 저장 및 조회

CRUD project Structure



CRUD 예제 : 성적관리

```
이름은? 홍길동
국어는? 90
영어는? 90
수학은? 95
```

1. 사용자 데이터 타입 정의
2. 하나의 데이터를 추가하는 기능(함수) 구현 (Create)
3. 하나의 데이터 조회 기능(함수) 구현(Read)
4. 하나의 데이터 수정 기능(함수) 구현(Update)
5. 하나의 데이터 삭제 기능(함수) 구현(Delete)
6. 각 함수를 사용하여 전체 프로그램을 구성하는 main 함수 구현

CRUD 예제 : 성적관리

```
이름은? 홍길동
국어는? 90
영어는? 90
수학은? 95
```

1. 사용자 데이터 타입 정의

- 이름/국어점수/영어점수/수학점수
- 사용자 정의 자료형 생성 및 변수 선언

```
typedef struct {
  char name[20];
  int kor;
  int eng;
  int math;
} Score;
```

```
typedef struct {
  char name[20];
  int score[3];
}Score;
```

```
Score s1;
Score kim, hong;
Score s[10];
Score *sp = &kim;

printf("%s", s1.name);
printf("%d", s[0].kor);
printf("%d", sp->eng);
```


CRUD 예제 : 성적관리

```
이름은? 홍길동
국어는? 90
영어는? 90
수학은? 95
```

2. 하나의 데이터를 추가하는 기능 구현 (Create)

- **함수 정의** / 함수 생성 / main에서 호출
- 함수명 / 입력 Parameters 정의 / 결과(Return type)
- addScore() / Score s1 / 성공여부(1,0)

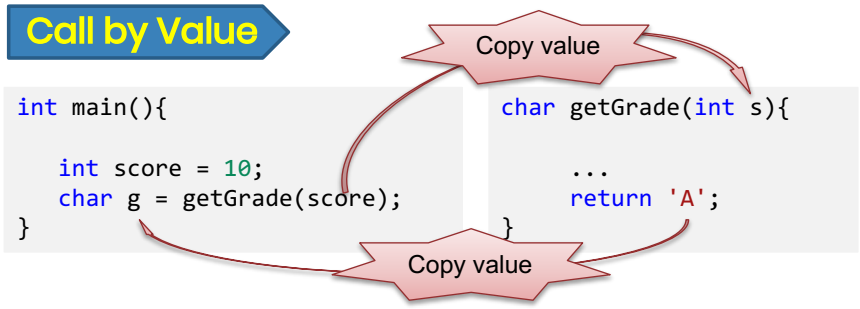
```
결과타입 함수명 ( 입력변수 ) {
    함수에서 실행할 문장1;
    함수에서 실행할 문장2;

    return value;
}
```

parameters

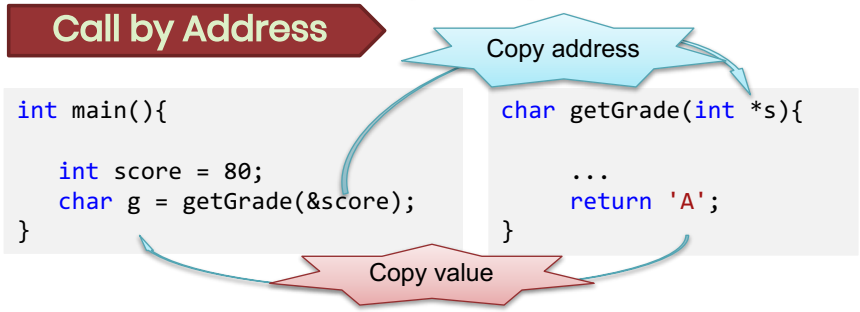
```
int addScore(Score *s1){
  //함수에서 실행할 문장들


  return 1;
}
```

함수호출

Call by Value



Call by Address



Call by Value

```c
#include <stdio.h>

void swapNumber(int first, int second){
   int temp;

   temp = first;
   first = second;
   second = temp;
}

int main(){
   int num1 = 10;
   int num2 = 20;

   printf("%d %d\n", num1, num2);
   swapNumber(num1, num2);
   printf("%d %d\n", num1, num2);

   return 0;
}
```

```
./main
10 20
10 20
```

Call by Address

```c
#include <stdio.h>

void swapNumber(int *first, int *second){
   int temp;

   temp = *first;
   *first = *second;
   *second = temp;
}

int main(){
   int num1 = 10;
   int num2 = 20;

   printf("%d %d\n", num1, num2);
   swapNumber(&num1, &num2);
   printf("%d %d\n", num1, num2);

   return 0;
}
```

```
./main
10 20
20 10
```

CRUD 예제 : 성적관리

```
이름은? 홍길동
국어는? 90
영어는? 90
수학은? 95
```

2. 하나의 데이터를 추가하는 기능 구현 (Create)

• 함수정의 / 함수생성 / main에서 호출

```c
int addScore(Score *s){
 printf("이름은? ");
 scanf("%s",s->name);

 printf("국어는? ");
 scanf("%d", &s->kor);

 printf("영어는? ");
 scanf("%d", &s->eng);

 printf("수학은? ");
 scanf("%d", &s->math);
 return 1;
}
```

```c
int main(void) {
  int result = 0;
  Score s1;

  result = addScore(&s1);

  return 0;
}
```


CRUD 예제 : 성적관리

3. 하나의 데이터 조회 기능 구현(Read)

• readScore() / Score s / 결과없음

```c
void readScore(Score s){
  int sum = s.kor + s.eng + s.math ;
  double avg= sum/3.0;

  printf("%8s %4d %4d %4d %5d %5.1f\n",
    s.name, s.kor, s.eng, s.math, sum, avg);

}
```

```
이름은? 홍길동
국어는? 100
영어는? 90
수학은? 80
 name    kor  eng  math  sum  avg
 홍길동  100   90   80   270  90.0
```

```c
int main(void) {
  int result = 0;
  Score s1;

  result = addScore(&s1);

  printf("Name Kor Eng Math Sum Avg \n");
  readScore(s1);

  return 0;
}
```

CRUD 예제 : 성적관리

4. 하나의 데이터 수정 기능 구현(Update)

- updateScore() / Score s / 성공여부(0,1)

```
이름은? 홍길동
국어는? 100
영어는? 90
수학은? 80
Name    Kor  Eng  Math  Sum  Avg
홍길동  100   90   80   270  90.0
이름은? 홍길순
국어는? 90
영어는? 80
수학은? 70
=> 수정성공!
Name    Kor  Eng  Math  Sum  Avg
홍길순   90   80   70   240  80.0
```

```c
int updateScore(Score *s){
 printf("이름은? ");
 scanf("%s",s->name);
 printf("국어는? ");
 scanf("%d", &s->kor);
 printf("영어는? ");
 scanf("%d", &s->eng);
 printf("수학은? ");
 scanf("%d", &s->math);
 printf("=> 수정성공!\n");
 return 1;
}
```

```c
int main(void) {
 ...

   updateScore(&s);
   printf("Name Kor Eng
Math Sum Avg \n");
   readScore(s);

   return 0;
}
```


CRUD 예제 : 성적관리

```
이름은? 홍길동
국어는? 100
영어는? 90
수학은? 80
Name    Kor  Eng  Math  Sum  Avg
홍길동  100   90   80   270  90.0
=> 삭제됨!
```

5. 하나의 데이터 삭제 기능 구현(D)

- 삭제조건 : kor, math, eng 모두 -1 경우
- deleteScore() / Score s / 성공여부(0,1)

```c
int deleteScore(Score *s){
 s->kor = -1;
 s->eng = -1;
 s->math = -1;
 return 1;
}
```

```c
int main(void) {
 ...

  int isdel = deleteScore(&s);
  if(isdel == 1)
     printf("=> 삭제됨!\n ");


   return 0;
}
```

CRUD 예제 : 성적관리

6. 전체 프로그램을 구성하는 main 함수 구현

- CRUD 함수 사용

```
*** 점수계산기 ***
1. 조회
2. 추가
3. 수정
4. 삭제
0. 종료

=> 원하는 메뉴는?
```

```c
int selectMenu(){
 int menu;
 printf("\n*** 점수계산기 ***\n");
 printf("1. 조회\n");
 printf("2. 추가\n");
 printf("3. 수정\n");
 printf("4. 삭제\n");
 printf("0. 종료\n\n");
 printf("=> 원하는 메뉴는? ");
 scanf("%d", &menu);
 return menu;
}
```

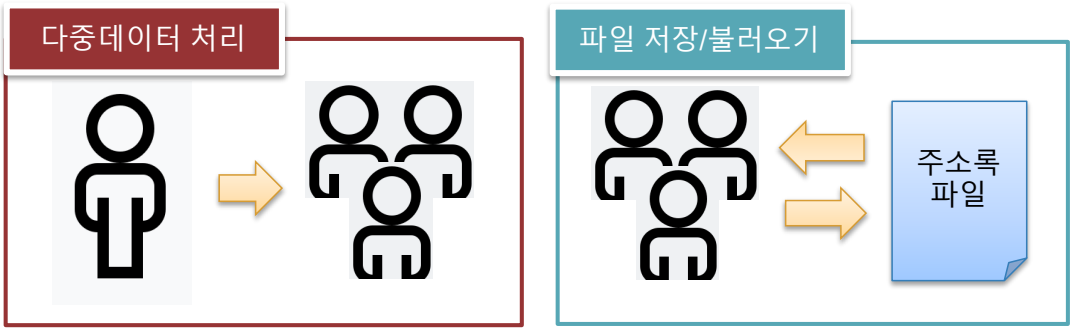
```c
int main(void) {
 Score s;
 int count=0, menu;

 while(1){
  menu = selectMenu();
  if(menu == 0) break;
  if(menu == 1 || menu ==3 || menu == 4)
       if(count == 0) continue;
  if(menu == 1){
    printf("Name Kor Eng Math Sum Avg \n");
    readScore(s);
  }
  else if(menu == 2) addScore(&s);
  else if(menu == 3) updateScore(&s);
  else if(menu == 4) deleteScore(&s);
 }
 return 0;
}
```


4

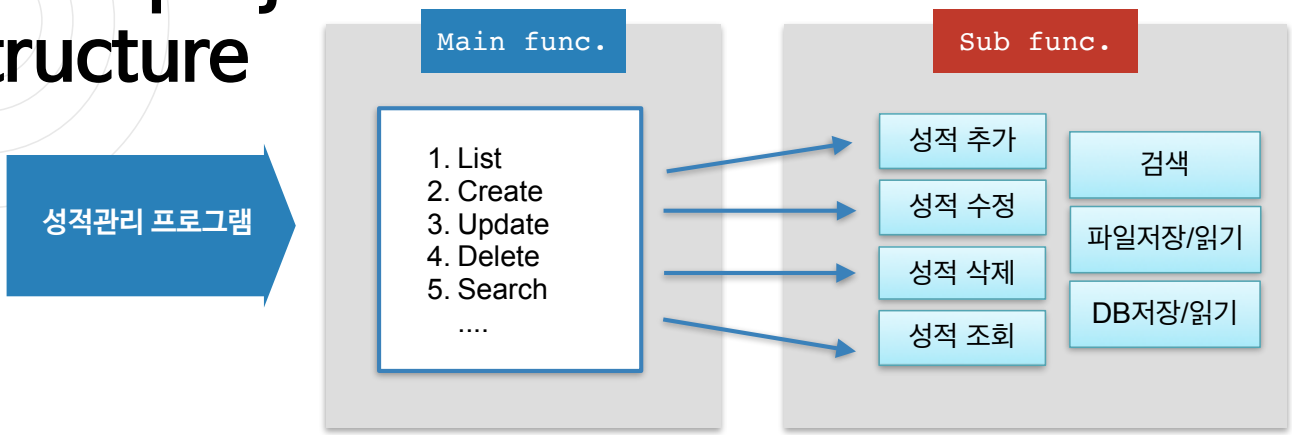
CRUD, 다중데이터 (C Programming)

CRUD

- Create(추가), Read(조회), Update(수정), Delete(삭제)
- 다중 데이터 처리
- 데이터 저장 / 불러오기 기능(File or Database)
- 부가기능 : 검색, 보고서 기능



CRUD project Structure



CRUD 예제 : 성적관리

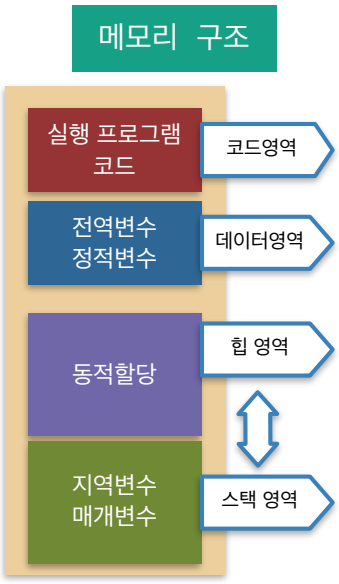
```
이름은? 홍길동
국어는? 90
영어는? 90
수학은? 95
```

1. 사용자 데이터 타입 정의
2. 하나의 데이터를 추가하는 기능(함수) 구현 (C)
3. 하나의 데이터 조회 기능(함수) 구현(R)
4. 하나의 데이터 수정 기능(함수) 구현(U)
5. 하나의 데이터 삭제 기능(함수) 구현(D)
6. 각 함수를 사용하여 전체 프로그램을 구성하는 main 함수 구현
7. **다중 데이터 처리**
8. 데이터 파일로 저장
9. 파일 데이터 가져오기 기능

CRUD 예제 : 성적관리

7. 다중 데이터 처리

- 배열 변수 , 포인터 배열 변수
- 포인터 배열 변수 사용시 데이터가 추가될 때 동적 메모리 할당 (stdlib.h)
  - malloc 함수 : 힙 영역에 메모리를 동적으로 할당
  - free 함수 : 힙 영역에 할당된 메모리를 해제



CRUD 예제 : 성적관리

다중데이터(배열)

1. 다중 데이터 처리를 위한 배열 변수 선언(데이터 수 고정)
   =〉 정적 배열
2. 데이터 추가 함수 호출 수정
3. 리스트 함수 추가()
   =〉 함수명 , parameter, return type
4. 수정 함수 호출 수정
5. 삭제 함수 호출 수정

CRUD 예제 : 성적관리

다중데이터(배열)

1. **다중데이터 처리를 위한 배열 변수 선언**
   - 배열 사용시 현재 위치(index)와 개수(count) 변수가 필요함

```c
int main(void) {
   int result = 0;
   Score s1;

   result = addScore(&s1);

   return 0;
}
```

```c
int main(void) {
   int count = 0; //데이터개수
   int result = 0;

   Score slist[100];
   int index =0; //데이터번호

   result = addScore(&slist[index]);

   return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(배열)

2. 데이터 추가 함수 호출 수정

- 배열 사용시 현재 위치(index)와 개수(count) 변수가 필요함

```c
int addScore(Score *s);
int main(void) {
  …

  else if(menu == 2) addScore(&s1);
   …

  return 0;
}
```

```c
int main(void) {
  …

  else if(menu == 2)
       count += addScore(&slist[index++]);
   …

  return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(배열)

3. 리스트 함수 추가(listScore)

- 기존 readScore() 함수 이용
- listScore() / 데이터배열, 배열개수/결과값 없음

```
No Name  Kor  Eng  Math  Sum  Avg
================================
 1 철수  100  100  100  300 100.0
 2 영희   90   95  100  285  95.0
 3 서연  100   95   90  285  95.0
```

```c
void listScore(Score *s, int count){
  printf("\nNo Name Kor Eng Math Sum Avg\n");
  printf("================================\n");
  for(int i =0; i <count ; i++){
     if(s[i].kor == -1) continue;
     printf("%2d ", i+1);
     readScore(s[i]);
  }
  printf("\n");
}
```

```c
int main(void) {
  …

  if(menu == 1)
      if(count > 0)
        listScore(slist, index);
   …

  return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(배열)

4. 수정 함수 호출 수정

- 수정하기 위한 리스트 보여준 후 선택한 번호 수정
- selectDataNo() / 데이터배열, 개수 / 선택한 번호 리턴

```
=> 원하는 메뉴는? 3

No Name  Kor  Eng  Math  Sum  Avg
================================
 1 철수  100  100  100  300 100.0
 2 영희   90   95  100  285  95.0
 3 서연  100   95   90  285  95.0

번호는 (취소 :0)? 1
이름은?김철수
국어는?95
영어는?95
수학은?95
=> 수정됨!
```

```c
int selectDataNo(Score *s, int count){
    int no;
    listScore(s, count);
    printf("번호는 (취소 :0)? ");
    scanf("%d", &no);
    return no;
}
```

```c
int main(void) {
  …

  else if(menu == 3){
    int no = selectDataNo(slist, index);
    if(no == 0){
        printf("=> 취소됨!\n");
        continue;
    }
    updateScore(&slist[no-1]);
   }
   …

  return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(배열)

5. 삭제 함수 호출 수정

- 삭제하기 위한 리스트 보여준 후 선택한 번호 수정
- 수정에서 리스트 보여주는 함수 사용(selectData
- 삭제한 데이터는 보이지 않도록 처리(listScore)

```
=> 원하는 메뉴는? 4

No Name  Kor  Eng  Math  Sum  Avg
================================
 1 김철수   95   95   95  285  95.0
 2 영희    90   95  100  285  95.0
 3 서연   100   95   90  285  95.0

번호는 (취소 :0)? 1
정말로 삭제하시겠습니까?(삭제 :1)1
=> 삭제됨!
```

```c
int main(void) {

    int no = selectDataNo(slist, index);
    if(no == 0){
      printf("=> 취소됨!\n");
      continue;
    }
    int deleteok;
    printf("정말로 삭제하시겠습니까?(삭제 :1)");
    scanf("%d", &deleteok);
    if(deleteok == 1){
       if(deleteScore(&slist[no-1])) count--;
    }
    …

  return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(포인터배열)

1. 다중 데이터 처리를 위한 포인터배열 변수 선언
2. 데이터 추가 함수 호출 수정
3. 리스트 함수 추가(listScore)
4. 수정 함수 호출 수정
5. 삭제 함수 호출 수정

CRUD 예제 : 성적관리

다중데이터(포인터배열)

1. 다중데이터 처리를 위한 포인터 배열 변수 선언

```
int main(void) {

   int count = 0;
   int result = 0;

   Score slist[100];
   int index =0;

  return 0;
}
```

```
int main(void) {
   int count = 0;
   int result = 0;

   Score *sp[100];
   int index =0;

   result = addScore(&slist[index]);

  return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(포인터배열)

2. 데이터 추가 함수 호출 수정

```
int main(void) {
  …
  else if(menu == 2) addScore(&s1);
  …
  return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>

int main(void) {
  …
  else if(menu == 2){
       sp[index] = (Score *)malloc(sizeof(Score));
       count += addScore(sp[index++]);
  }
  return 0;
}
```


CRUD 예제 : 성적관리

다중데이터(포인터배열)

3. 리스트 함수 추가(listScore)

- 조회함수(readScore)함수를 이용하여 리스트보기 함수 구현

```
No Name  Kor  Eng  Math  Sum  Avg
================================
 1 철수  100  100  100  300 100.0
 2 영희   90   95  100  285  95.0
 3 서연  100   95   90  285  95.0
```

```
void listScore2(Score *s[], int count){
  printf("\nNo Name Kor Eng Math Sum Avg\n");
  printf("================================\n");
  for(int i =0; i <count ; i++){
     if(s[i]->kor == -1) continue;
     printf("%2d ", i+1);
     readScore(*s[i]);
  }
  printf("\n");
}
```

```
int main(void) {
  Score *sp[100];
  …
  if(menu == 1){
      if(count > 0)
        listScore2(sp, index);
      else
        printf("데이터가 없습니다.\n");
  }
  …
  return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(포인터배열)

4. 수정 함수 호출 수정

- 수정하기 위한 리스트 보여준 후 선택한 번호 수정

```
=> 원하는 메뉴는? 3
No Name  Kor  Eng  Math  Sum  Avg
================================
1 철수   100  100  100   300 100.0
2 영희    90   95  100   285  95.0
3 서연   100   95   90   285  95.0
번호는 (취소 :0)? 1
이름은?김철수
국어는?95
영어는?95
수학은?95
=> 수정됨!
```

```
int selectDataNo2(Score *s[], int count){
   int no;
   listScore2(s, count);
   printf("번호는 (취소 :0)? ");
   scanf("%d", &no);
   return no;
}
```

```
int main(void) {
  …

  else if(menu == 3){
    int no = selectDataNo2(sp, index);
    if(no == 0){
        printf("=> 취소됨!\n");
        continue;
    }
    updateScore(sp[no-1]);
   }
   …

  return 0;
}
```

CRUD 예제 : 성적관리

다중데이터(포인터배열)

```
int main(void) {
  …

   int no = selectDataNo2(sp, index);
   if(no == 0){
     printf("=> 취소됨!\n");
     continue;
   }
   int deleteok;
   printf("정말로 삭제하시겠습니까?(삭제 :1)");
   scanf("%d", &deleteok);
   if(deleteok == 1){
      if(sp[no-1]) free(sp[no-1]);
      sp[no-1] = NULL;
      count--;
   }
   …

  return 0;
}
```

```
void listScore2(Score *s[], int count){
  printf("\nNo Name Kor Eng Math Sum Avg\n");
  printf("================================\n");
  for(int i =0; i <count ; i++){
     if(s[i] == NULL) continue;
     printf("%2d ", i+1);
     readScore(*s[i]);
  }
  printf("\n");
}
```

요약

1. 배열 변수 사용
2. 배열 변수와 반복문
3. 포인터 변수의 이해
4. 포인터 배열 변수의 이해
5. 동적 메모리 할당 / 해제하는 법
6. 다수 데이터 관리방법

5

CRUD(File IO), Library (C Programming)

문자열 입력

- 공백 포함하지 않고 문자열 입력

```
scanf("%s", str);
```

```
./main
문자열? Hello World!!!
Hello
```

- 공백 포함하여 문자열 입력
  - scanf 함수 사용 : 문자열 + ₩0(널문자) (개행문자 포함안됨)

```
scanf("%[^₩n]s", str);
```

```
./main
문자열? Hello World!!!
Hello World!!!
```

  - fgets 함수 사용 : 문자열 + ₩0 (개행문자 포함)

```
fgets(str, sizeof(str), stdin);
```

```
./main
문자열? Hello World!!!
Hello World!!!
```


입력버퍼

```
#include <stdio.h>

int main(){

  char str[20];
  char c;

  scanf("%s", str);
  printf("%s\n", str);
  scanf("%c", &c);

  scanf("%[^\n]", str);
  printf("%s\n", str);

  return 0;
}
```

```
./main
hello
hello
hello world!!!
hello world!!!
```

- 표준 입력(키보드) 문자를 **입력버퍼**에 임시 저장 후 처리
- 입력버퍼에 문자가 있다면 scanf()함수가 실행되지 않은 것 처럼 보임

  예) 문자열 (숫자)입력 후 문자 입력받는 경우

- 해결방법
  - getchar() 함수 사용

h	e	l	l	o	\0

```
scanf("%s", str);
getchar();
scanf("%c", &c);
```

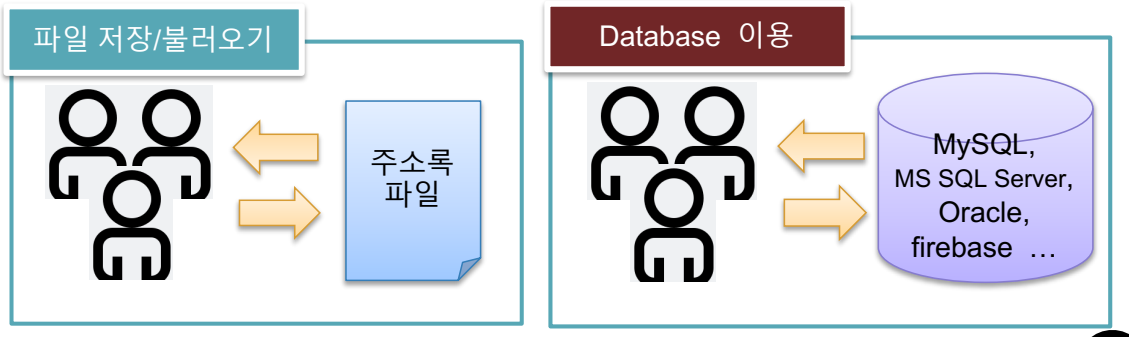
  - scanf 함수 대신 fgets 함수 사용(개행문자가 포함)

h	e	l	l	o	\n	\0

```
fgets(str, sizeof(str), stdin);
str[strlen(str)-1] = '\0';
scanf(" %c", &c);
```

CRUD

- Create(추가), Read(조회), Update(수정), Delete(삭제)
- 다중 데이터 처리
- 데이터 저장 / 불러오기 기능(File or Database)
- 부가기능 : 검색, 보고서 기능



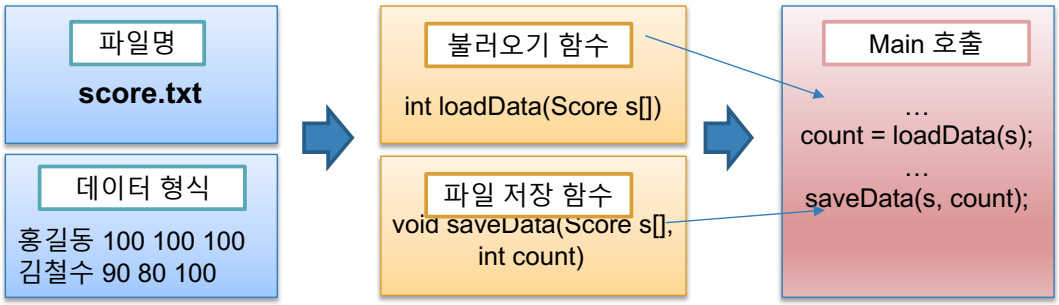
CRUD 예제 : 성적관리

```
이름은? 홍길동
국어는? 90
영어는? 90
수학은? 95
```

1. 사용자 데이터 타입 정의
2. 하나의 데이터를 추가하는 기능(함수) 구현 (C)
3. 하나의 데이터 조회 기능(함수) 구현(R)
4. 하나의 데이터 수정 기능(함수) 구현(U)
5. 하나의 데이터 삭제 기능(함수) 구현(D)
6. 각 함수를 사용하여 전체 프로그램을 구성하는 main 함수 구현
7. 다중 데이터 처리
8. **파일에 데이터 저장 기능 구현**
9. **파일 데이터 가져오기 기능 구현**
10. **검색 / 라이브러리 생성**

File IO

1. 파일명, 파일에 작성할 데이터 포맷 정의
2. 파일에 데이터 저장하는 기능(함수) 구현
3. 파일에서 데이터 불러오는 기능(함수) 구현
4. main 함수에서 호출



CRUD 예제 : 성적관리

8. 파일에 데이터 저장 기능 구현

1. 데이터를 파일에 저장하는 함수 정의
   - saveData()/데이터배열, 개수/ 결과값 없음
2. 파일 포인터 생성
3. 파일 열기
4. 파일에 데이터 출력
5. 파일 닫기
6. main함수에서 호출

```
void saveData(Score *s, int count)
{
  FILE *fp;
  fp = fopen("score.txt", "wt");

  for(int i = 0; i < count; i++){
    if(s[i].kor == -1) continue;
    fprintf(fp, "%s %d %d %d\n"
      ,s[i].name,s[i].kor,s[i].eng
      ,s[i].math);
  }
  fclose(fp);
  printf("=> 저장됨! ");
}
```

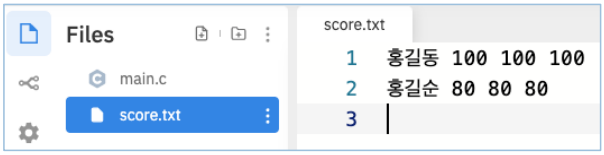
CRUD 예제 : 성적관리

8. 파일에 데이터 저장 기능 구현

• 메뉴추가 / main함수에 연결

```
*** 점수계산기 ***
1. 조회
2. 추가
3. 수정
4. 삭제
5. 저장
0. 종료
```

```
=> 원하는 메뉴는? 5
=> 저장됨!
```




```c
int main(void) {
 Score slist[100];
 int count=0, index = 0, menu;

 while(1){
  menu = selectMenu();
  if(menu == 0) break;
  if(menu == 1 || menu ==3 || menu == 4)
       if(count == 0) continue;
  if(menu == 1){
    listScore(slist, index);
  }

  . . .

  else if(menu == 5){
    saveData(slist, index);
  }
 }
 return 0;
}
```


CRUD 예제 : 성적관리

9. 파일 데이터 가져오기 기능 구현

1. 파일 데이터를 가져오는 함수 정의
   • loadData()/데이터배열/ 데이터개수
2. 파일 포인터 생성
3. 파일 열기
4. 데이터 읽어오기
5. 파일 닫기
6. main함수에서 프로그램 시작시 바로 호출

```c
int loadData(Score *s){
  int count = 0, i = 0;
  FILE *fp;
  fp = fopen("score.txt", "rt");
  for(; i < 100; i++){
    fscanf(fp, "%s", s[i].name);
    if(feof(fp)) break;
    fscanf(fp, "%d", &s[i].kor);
    fscanf(fp, "%d", &s[i].eng);
    fscanf(fp, "%d", &s[i].math);
  }
  fclose(fp);
  printf("=> 로딩 성공!\n");
  return i;
}
```

```c
int main(void) {
 Score slist[100];
 int count=0, index = 0, menu;

 count = loadData(slist);
 index = count;

 while(1){
```

CRUD 예제 : 성적관리

```
=> 원하는 메뉴는? 6
검색할 이름? 철수

No Name  Kor  Eng  Math  Sum  Avg
================================
 2 김철수   90   90   100  280  93.3
```

```
=> 원하는 메뉴는? 6
검색할 이름? 영희

No Name  Kor  Eng  Math  Sum  Avg
================================
=> 검색된 데이터 없음!
```

10. 이름 검색 기능 구현

- 이름검색 함수 생성
  - searchName()/배열,배열개수/결과값 없음
- strstr(대상 문자열, 검색문자열) 함수 사용
  - #include <string.h> 에 선언
  - 리턴값 NULL이면 검색 실패
- 메뉴에 추가 / main 함수에서 호출

CRUD 예제 : 성적관리

10. 이름 검색 기능 구현

```
=> 원하는 메뉴는? 6
검색할 이름? 철수

No Name  Kor  Eng  Math  Sum  Avg
================================
 2 김철수   90   90   100  280  93.3
```

```
=> 원하는 메뉴는? 6
검색할 이름? 영희

No Name  Kor  Eng  Math  Sum  Avg
================================
=> 검색된 데이터 없음!
```

```c
#include <stdio.h>
#include <string.h>

                                    …
void searchName(Score *s, int count){
 int scnt = 0;
 char search[20];

 printf("검색할 이름? ");
 scanf("%s", search);

 printf("\nNo Name Kor Eng Math Sum Avg\n");
 printf("================================\n");
 for(int i =0; i <count ; i++){
   if(s[i].kor == -1) continue;
   if(strstr(s[i].name, search)){
        printf("%2d ", i+1);
        readScore(s[i]);
        scnt++;
   }
 }
 if(scnt == 0) printf("=> 검색된 데이터 없음!");
 printf("\n");
}
```

Library

1. 필요한 기능을 함수로 구현하고 모아 집합 파일로 만든 후 여러 프로그램에서 재사용 가능
2. 미리 컴파일 되어 있어 컴파일 시간 단축됨
3. 링크될 수 있도록 컴파일된 형태의 목적 코드(Object code)형태로 존재
4. 소스 외부 유출 방지 효과
5. 종류
   - 정적 라이브러리(*.lib, *.a) : 프로그램을 컴파일하는 과정(link)에서 라이브러리 파일을 실행 바이너리에 포함
   - 동적 라이브러리(*.dll) : 프로그램을 실행할 때 메모리에 로드 후 동적으로 사용

정적 라이브러리 실습

라이브러리 생성

1. 헤더파일 생성(*.h)
2. 라이브러리에 포함되는 함수 구현 코드(*.c)
3. 라이브러리 컴파일(*.o)
4. 정적라이브러리 생성(*.a)

라이브러리 사용

5. 라이브러리를 이용할 main함수 구현
6. Object 파일을 사용한 컴파일 및 실행 (Object파일의 모든 코드 포함)
7. 정적라이브러리를 이용한 컴파일 및 실행(정적 라이브러리에서 필요한 코드만 포함)

정적 라이브러리 실습

```c
#include <stdio.h>

int add(int n1, int n2);
int subtract(int n1, int n2);
int multiply(int n1, int n2);
int divide(int n1, int n2);

void printResult(int n1, int n2, int r, char op);
```

```c
int main(){
  int n1,n2, result;

  char op;

  printf("두 수를 입력 :");
  scanf("%d %d", &n1, &n2);
  getchar();
  printf("연산자 입력(+, -, *, /) :");
  scanf("%c", &op);

  if(op == '+') result = add(n1, n2);
  else if(op == '-') result = subtract(n1, n2);
  else if(op == '*') result = multiply(n1, n2);
  else if(op == '/') result = divide(n1, n2);
  else{
    printf("=> 잘못 입력!\n");
    return 0;
  }
  printResult(n1,n2,result,op);
  return 0;
}
int add(int n1, int n2){return n1 + n2;}
int subtract(int n1, int n2){return n1 - n2;}
int multiply(int n1, int n2){return n1 * n2;}
int divide(int n1, int n2){return n1 / n2;}
void printResult(int n1, int n2, int r, char op){
  printf("=> %d %c %d = %d\n",n1,op,n2,r);
}
```


정적 라이브러리 실습

1. 헤더파일 생성(calculator.h)

```c
int add(int n1, int n2);
int subtract(int n1, int n2);
int multiply(int n1, int n2);
int divide(int n1, int n2);
void printResult(int n1, int n2, int r, char op);
```

2. 라이브러리에 포함할 함수 생성(calculator.c)

```c
#include <stdio.h>
#include "calculator.h"

int add(int n1, int n2){return n1 + n2;}
int subtract(int n1, int n2){return n1 - n2;}
int multiply(int n1, int n2){return n1 * n2;}
int divide(int n1, int n2){return n1 / n2;}
void printResult(int n1, int n2, int r, char op){
  printf("=> %d %c %d = %d\n",n1,op,n2,r);
}
```

정적 라이브러리 실습

3. 라이브러리 컴파일(*.o)

```
User1-ui-Macmini inc % gcc -c calculator.c
User1-ui-Macmini inc % ls
calculator.c    calculator.h    calculator.o
```

4. 정적라이브러리 생성(*.a) : 여러 Object 파일을 하나로 묶음

```
User1-ui-Macmini inc % ar rv libcal.a calculator.o
ar: creating archive libcal.a
a - calculator.o
User1-ui-Macmini inc % ls
calculator.c    calculator.h    calculator.o    libcal.a
```


정적 라이브러리 실습

5. 라이브러리를 이용하여 main함수 구현 (mycal.c)

```c
#include <stdio.h>
#include "calculator.h"

int main(){
  int n1,n2, result;
  char op;

  printf("두 수를 입력 :");
  scanf("%d %d", &n1, &n2);
  getchar();
  printf("연산자 입력(+, -, *, /) :");
  scanf("%c", &op);

  if(op == '+') result = add(n1, n2);
  else if(op == '-') result = subtract(n1, n2);
  else if(op == '*') result = multiply(n1, n2);
  else if(op == '/') result = divide(n1, n2);
  else{
    printf("=> 잘못 입력!\n");
    return 0;
  }
  printResult(n1,n2,result,op);
  return 0;
}
```

정적 라이브러리 실습

6. Object파일을 이용한 컴파일 및 실행 파일 생성
   - Object파일의 모든 코드를 가져와 실행파일 생성

```
User1-ui-Macmini 5_lab % gcc ./inc/calculator.o mycal.c I./inc -o mycal2
User1-ui-Macmini 5_lab % ls mycal2
mycal2
User1-ui-Macmini 5_lab % ./mycal2
두 수를 입력 :10 20
연산자 입력(+, -, *, /) :+
=> 10 + 20 = 30
```


정적 라이브러리 실습

7. 정적 라이브러리를 이용한 컴파일 및 실행 파일 생성
   - 필요한 코드만 가져와 하나의 실행파일로 생성함

```
User1-ui-Macmini 5_lab % ls ./inc
calculator.c    calculator.h    calculator.o    libcal.a
User1-ui-Macmini 5_lab % gcc mycal.c -o mycal -I./inc -L./inc -lcal
User1-ui-Macmini 5_lab % ls mycal
mycal
```

- -o mycal mycal 실행파일 생성
- -I./inc 현재 폴더 아래 inc 폴더에서 헤더파일을 찾는 옵션(calculator.h)
- -L./inc 현재 폴더 아래 inc폴더에서 라이브러리를 찾는 옵션(libcal.a)
- -lcal 정적라이브러리인 libcal.a와 링크하라는 옵션

요약

1. 파일 입출력 과정의 이해
2. 파일 입출력에 필요한 함수 사용법
3. 검색 함수의 이해(strstr)
4. 정적라이브러리 vs 동적라이브러리
5. 정적라이브러리 생성법
6. Object 파일을 이용한 실행파일 생성
7. 정적라이브러리를 이용한 실행파일 생성