c++를 이용한 Student Database Management system 프로그램의 분석

대구가톨릭대학교 컴퓨터소프트웨어학부 사이버보안전공

배채윤

01

서론

이 프로그램은 학생 정보에 대한 리스트를 쉽게 볼 수 있도록 하는 프로그램이며, 기록을 추가, 나열, 수정 ,삭제 등 파일에 저장된 정보를 선택할 수 있도록 한다.

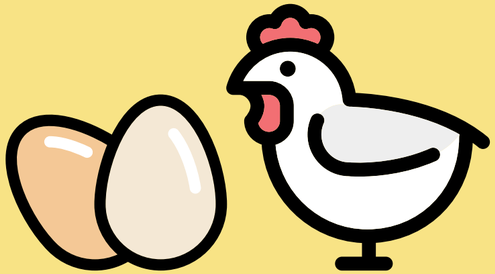
(1) 구조체

구조체는 정수나 문자, 실수나 포인터 그리고 이들의 배열 등을 묶어 하나의 자료형으로 이용하는 것이다.
즉, 연관성이 있는 서로 다른 개별적인 자료형의 변수들을 하나의 단위로 묶은 새로운 자료형을 구조체(structure)라 한다. 구조체는 연관된 멤버로 구성되는 통합 자료형으로 대표적인 유도 자료형이다. 즉 기존 자료형으로 새로이 만들어진 자료형을 유도 자료형이라 한다.

구조체를 자료형으로 사용하려면 먼저 구조체를 정의해야 한다.
구조체를 사용하려면 먼저 구조체를 만들 구조체 틀을 정의하여야 한다.

<구조체 틀:정의>

```
struct lecture {
    char name[20];
    int credit;
    int hour;
};
```



(1) 구조체-1

구조체를 정의하는 방법은 키워드 'struct' 다음에 구조체 태그이름을 기술하고 중괄호를 이용하여 원하는 멤버를 여러 개의 변수로 선언하는 구조다.
구조체를 구성하는 하나 하나의 항목을 구조체 멤버 또는 필드라 한다.
구조체 정의는 변수의 선언과는 다른 것으로 변수선언에서 이용될 새로운 구조체 자료형을 정의하는 구문이다.

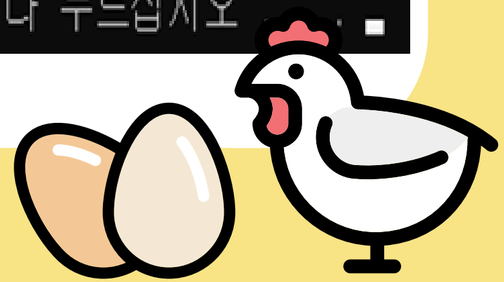
구조체 내부의 멤버 선언 구문은 모두 하나의 문장이므로 반드시 세미콜론으로 종료해야 하며, 각 구조체 멤버의 초기값을 대입할 수 없다. 마지막 멤버 선언에도 반드시 세미콜론이 빠지지 않도록 주의해야 한다.
구조체 멤버로는 일반 변수, 포인터 변수, 배열, 다른 구조체 변수 및 구조체 포인터도 해당된다. 선언된 구조체형 변수는 접근연산자 .를 사용하여 멤버를 참조할 수 있다.
실제 구조체의 크기는 멤버의 크기의 합보다 크거나 같다

```cpp
#include<iostream>
using namespace std;
//구조체의 선언
struct Person {
    char name[30];
    char phone_num[20];
    char job[10];
    int age;
};
int main() {
    //구조체 변수
    struct Person person {
        "chaeyoon", "010-1234-5678", "student", 20
    };
    cout << "Name:" << person.name << endl;
    cout << "Phone_num:" << person.phone_num << endl;
    cout << "Job:" << person.job << endl;
    cout << "age:" << person.age << endl;

    return 0;
}
```

구조체 예시 코드 결과

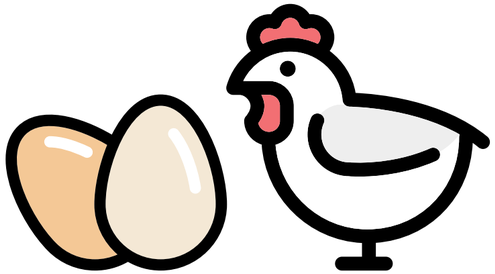
```
Name:chaeyoon
Phone_num:010-1234-5678
Job:student
age:20
계속하려면 아무 키나 누르십시오 . . .
```



(2) 배열

배열(array)은 여러 변수들이 같은 배열이름으로 일정한 크기의 연속된 메모리에 저장되는 구조이다. 배열을 이용하면 변수를 일일이 선언하는 번거로움을 해소할 수 있고, 배열을 구성하는 각각의 변수를 참조하는 방법도 간편하며, 반복 구문으로 쉽게 참조할 수 있다.

배열은 동일한 자료 유형이 여러 개 필요한 경우에 유용한 자료 구조 이다.
즉 배열은 한 자료유형의 저장공간인 원소를 동일한 크기고 지정된 배열크기만큼 확보한 연속된 저장공간이다. 배열은 변수이므로 배열마다 고유한 배열이름을 갖는다. 배열을 구성하는 가각의 항목을 배열의 원소라 한다. 그러므로 배열에서 중요한 요소는 배열이름, 원소 자료유형, 배열 크기이다. 배열원소는 첨자(index) 번호라는 숫잦를 이용해 쉽게 접근할 수 있다.



(2) 배열-1

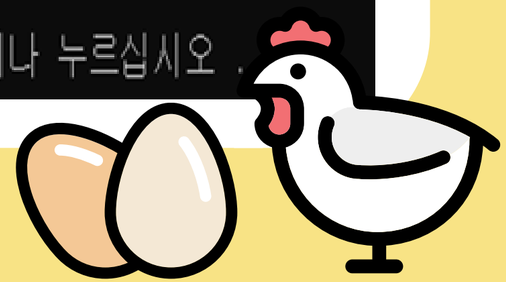
배열선언은 int data[10];과 같이 원소자료유형 배열이름[배열크기];로 한다. 배열선언 시 초기값 지정이 없다면 반드시 배열크기는 양의 정수로 명시 되어야 한다.
배열의 크기를 지정하는 부분에는 양수 정수로 리터럴 상수와 매크로 상수 또는 이들의 연산식이 올 수 있다. 변수와 const 상수로는 배열의 크기를 지정할 수 없다.
배열 선언 후 배열원소에 접근하려면 배열이름 뒤에 대괄호 사이 첨자(index)를 이용한다.
배열에서 유효한 첨자의 범위는 0부터 (배열크기-1)까지이며, 첨자의 유효 범위를 벗어나 원소를 참조하면 문법오류 없이 실행오류가 발생한다.
배열선언 시 대괄호 안의 수는 배열크기이다.
일반 배열 선언과 다르게 배열크기는 생략할 수 있으며, 생략하면 자동으로 중괄호 사이에 기술된 원소 수가 배열크기가 된다.

원소값을 나열하기 위해 콤마를 사용하고 전체를 중괄호로 묶는다.
만일 배열크기가 초기값 원소 수보다 크면 지정하지 않은 원소의 초기값은 자동으로 모두 기본값으로 저장된다. 즉 정수형은 0, 실수형은 0.0 그리고 문자형은 '₩0'인 널문자가 자동으로 채워진다. 배열크기가 초기값 원소 수보다 작으면 배열 저장공간을 벗어나므로 문법오류가 발생한다.

```c
int main() {
    int a[5];//배열 선언
    int b[5] = { 1,2,3,4,5 };//배열 초기화
    for (int i = 0; i <= 5; i++)
        printf("b[%d]=%d₩n", i, b[i]);
    return 0;
```

배열 예시 코드 결과

```
b[0]=1
b[1]=2
b[2]=3
b[3]=4
b[4]=5
b[5]=-858993460
계속하려면 아무 키나 누르십시오 . . .
```

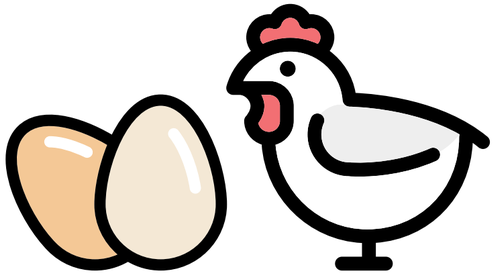


(3) 반복문 while

반복(repetition)은 말 그대로 같거나 비슷한 일을 여러 번 수행하는 작업이다.
반복과 같은 의미로 순환(loop)이라는 표현도 함께 사용한다.
먼저 반복문에는 3가지 함수가 존재한다.
for, while, do while문이 있다. 반복문에는 항상 조건이 존재하며 조건이 참일 경우에만 괄호안에 있는 소스코드가 실행이된다.

반복조건을 검사하여 반복을 수행하는 while구문, 제일 나중에 반복 조건 검사하여 반복을 수행하는 do while구문, 초기화와 반복조건, 그리고 증감연산의 세부분으로 나누어 일정한 횟수의 반복에 적합한 for문으로 구분할 수 있다.

반복조건을 만족하면 일정하게 반복되는 부분을 반복몸체(repetition body)라 한다.
반복문 while은 단순한 숫자의 반복이 아니라 반복할 때마다 조건을 따지는 반복문으로, 조건식이 반복몸체 앞에 위치한다.



반복문의 종류

```
while (<반복조건>)
{//반복몸체(loop body);
    <해야할 일>;
}
```

while 반복

```
do {
    //반복몸체(loop body);
    <해야할 일>;
} while (<반복조건>);
```

do while 반복

```
for (<초기화>; <반복조건>; <증감>)
{
    //반복몸체(loop body);
    <해야할 일>;
}
```

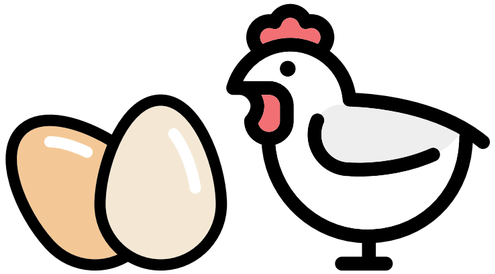
for 반복

(4) 포인터

먼저 주소개념에 대해 설명하자면, 메모리 공간은 8비트인 1바이트마다 고유한 주소가 있다. 메모리 주소는 0부터 바이트마다 1씩 증가한다. 메모리 주소는 저장 장소인 변수이름과 함께 기억장소를 참조하는 또 다른 방법이다.

주소는 변수이름과 같이 저장장소를 참조하는 하나의 방법이다.

&가 피연산자인 변수의 메모리 주소를 반환하는 주소연산자이다. 변수의 주소값은 형식제어문자 %u 또는 %d로 직접 출력할 수 있다. 그러나 최근 비주얼 스튜디오에서는 경고가 발생하니 주소값을 int 또는 unsigned로 변환하여 출력한다. 만일 16진수로 출력하려면 형식제어문자 % p를 사용한다. 주소연산자 &는 상수나 표현식에는 사용할 수 없다.



(4) 포인터-1

변수의 주소도 포인터 변수에 저장할 수 있다. 변수의 주소값은 반드시 포인터 변수에 저장해야 한다. 포인터 변수는 주소값을 저장하는 변수로 일반 변수와 구별되며 선언방법이 다르다. 포인터 변수 선언에서 자료형과 포인터 변수이름 사이에 연산자 *를 삽입한다.

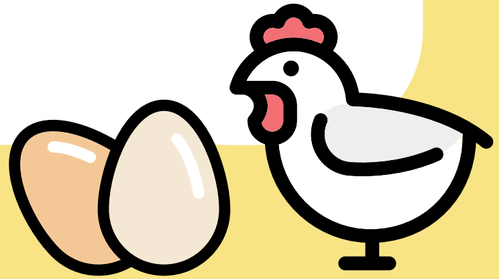
변수 자료형이 다르면 그 변수의 주소를 저장하는 포인터의 자료형도 달라야 한다. 어느 변수의 주소값을 저장하려면 반드시 그 변수의 자료유형과 동일한 포인터 변수에 저장해야 한다. 포인터 변수선언에서 포인터를 의미하는 *는 자료형과 변수이름 사이에만 위치하면 된다.

예를 들어 문장 ptrint=&data;는 포인터 변수 ptrint에 변수 data의 주소를 저장하는 문장이다.
이러한 관계를 '포인터변수 ptrint는 변수 data를 가리킨다' 또는 '참조한다'라고 표현한다.
포인터 변수는 가리키는 변수의 종류에 관계없이 크기가 모두 4바이트이다.

```
#include<iostream>
using namespace std;
int main() {
    int abc = 1000;
    int*p = &abc;
    cout << abc << endl;
    cout << "*p is:" << *p << endl;
}
```

포인터 예시 코드 결과

```
1000
*p is:1000
계속하려면 아무 키나 누르십시오 . . .
```

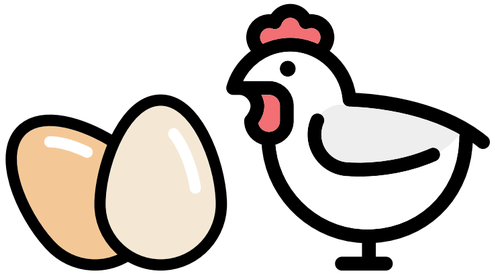


(5) if문과 if else문

문장 if는 조건에 따른 선택을 지원하는 구문이다.
if문의 형태는 if(cond) stmt;;이다. if문에서 조건식 cond가 0이 아니면 (참) stmt를 실행하고, 0이면(거짓) stmt를 실행하지 않는다. 문장 if의 조건식 (cond)는 반드시 괄호가 필요하며, 참이면 실행되는 문장은 반드시 들여쓰기를 하도록 한다.

if문은 조건이 만족되면 특정한 문장을 실행하는 구문이다. 반대로 조건이 만족되지 않은 경우에 실행할 문장이 있다면 else를 사용한다. 조건문 if에서 키워드 else를 사용하여 조건 연산값이 0(거짓)이면 else 이후의 문장을 실행하는 구문을 만들 수 있다.

조건문 if(cond) stmt1; else stmt2;는 조건 cond를 만족하면 stmt1를 실행하고, 조건 cond를 만족하지 않으면 stmt2를 실행하는 문장이다.
조건문 if else는 stmt1과 stmt2 둘 중의 하나를 선택하는 구문이다.
예를 들어 정수 n이 짝수인지 아니면 홀수인지 판단할 수 있는 조건식으로 (n%2==0) 또는 (n%2)이 주로 사용될 수 있다.

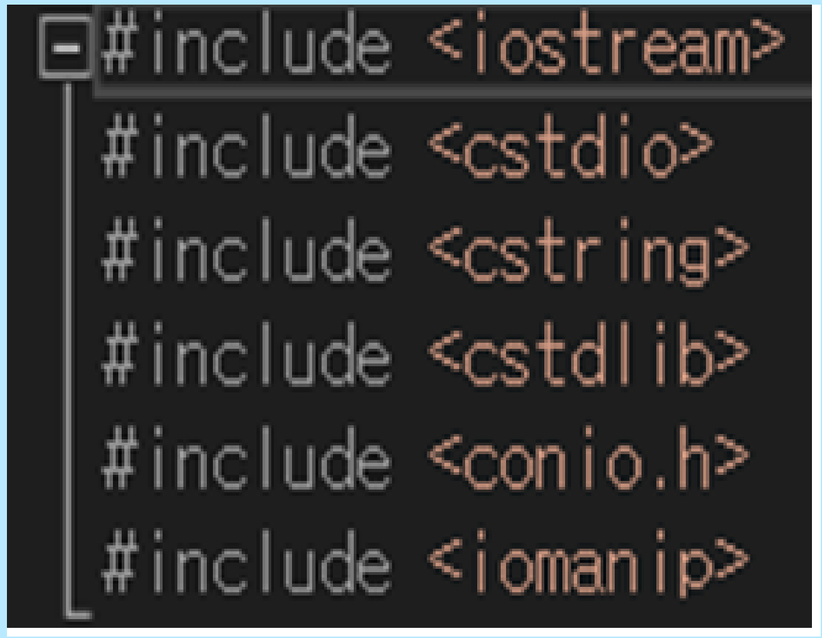


02

프로그램 구성

표준 헤더 파일을 자신의 소스에 포함시키기 위해 #include 전처리기를 사용하였다.
main()함수에는 구조체를 정의하고 상속받아 배열선언을 하였다.
while 반복문과 , if문이 사용되었다.

설명



#include <iostream> // iostream는 input+output+stream의 약자로써 입출력이라는 명령을 할 때 필요한 문장이다.

#include <cstdio> // C스타일 입.출력 함수

#include <cstring> // 문자열 클래스 string을 쓰기 위함.

#include <cstdlib> // 범용 유틸리티:프로그램 통제, 동적 메모리 할당, 난수, 정렬 및 탐색

#include <iomanip> // 형식 또는 입력 및 출력을 제어하는 도우미 기능

설명

```c
int main()
{
    FILE *fp, *ft;
    char another, choice;

    struct student
    {
        char first_name[50], last_name[50];
        char course[100];
        int section;
    };

    struct student e;
    char xfirst_name[50], xlast_name[50];
    long int recsize;
```

FILE *fp, *ft // 파일 포인터 선언

struct student // 구조체 student 선언

char first_name[50], last_name[50]; // char형 멤버 함수 선언, 배열 선언

char course[100] // char형 멤버 함수 선언, 배열 선언

int section // int형 멤버 함수 선언

Thank you

대구가톨릭대학교 컴퓨터소프트웨어학부 사이버보안전공

배채윤