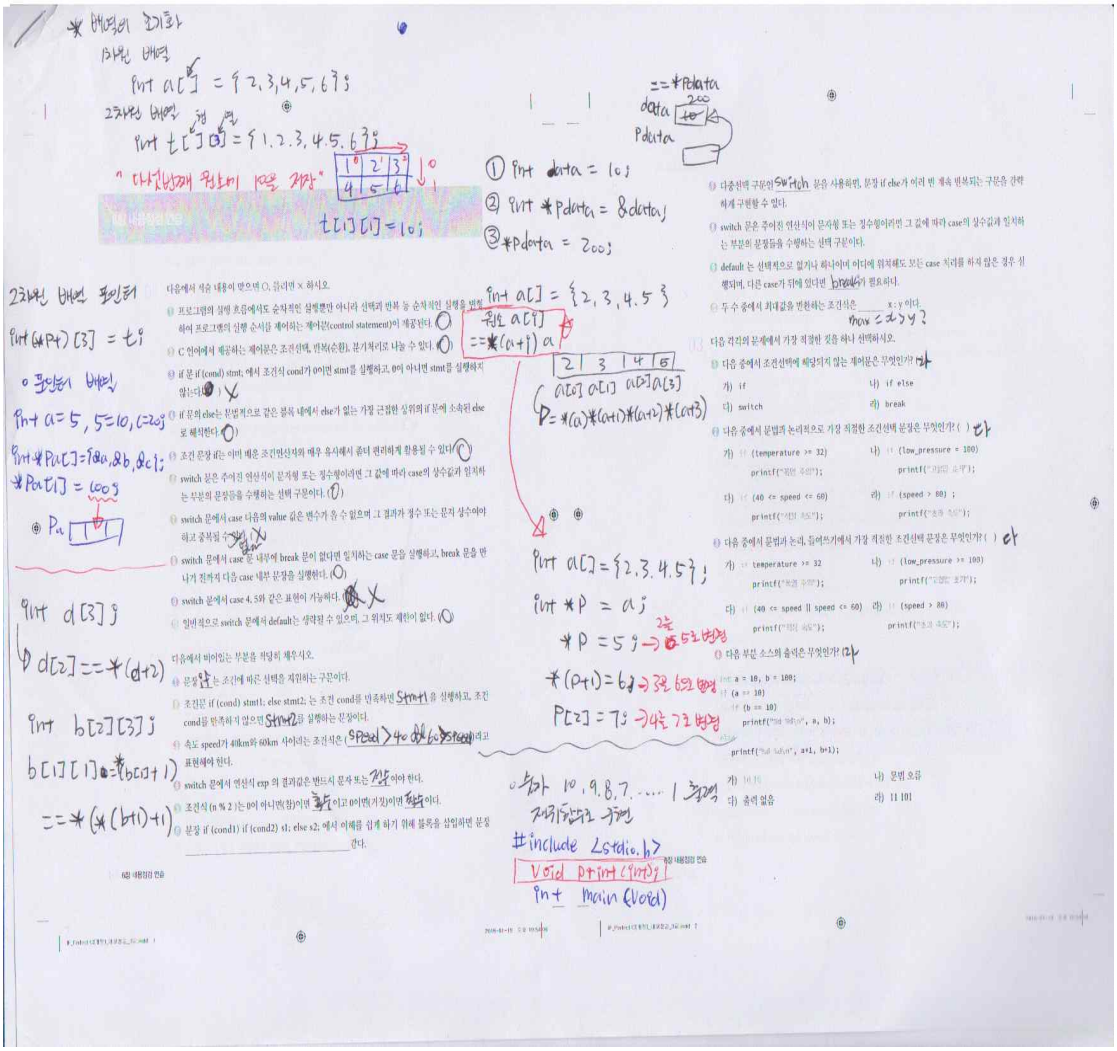
C언어 포트폴리오

20160755 김지원



Date. Page.

06 조건

6-1 제어문의 종류

비 순차적 실행의 제어문

- 프로그램의 실행 흐름에서도 순차적인 실행뿐만 아니라 선택과 반복 등 순차적인 실행을 변형하여 프로그램의 실행 순서를 제어하는 제어문(control statement)이 제공

제어문 종류

- 조건선택 구문이란 두 개 또는 여러 개 중에서 한 개를 선택하도록 지원하는 구문이다.

- 반복 또는 순환 구문이란 정해진 횟수 또는 조건을 만족하면 정해진 몇 개의 문장을 여러 번 실행하는 구문이다.

- 분기 구문은 작업을 수행 도중 조건에 따라 반복이나 선택을 빠져 나가거나(break), 인정구문은 실행하지 않고 다음 반복을 실행하게나(continue), 지정된 위치로 이동하거나(goto) 또는 작업 수행을 마치고 이전 위치로 돌아가는(return) 구문이다.

6-2 조건에 따른 선택 구문

if 문장

- 문장 if는 위에서 살펴본 조건에 따른 선택을 지원하는 구문.
- 형태는 if(cond) Stmt;이다. if 문에서 조건식 cond가 0이 아니면(참) Stmt를 실행하고, 0이면(거짓) Stmt를 실행하지 않는다.

Date. Page.

if else 문장

- 조건문 if (cond) Stmt1; else Stmt2; 는 조건 (cond)을 만족하면 Stmt1을 실행하고, 조건 (cond)을 만족하지 않으면 Stmt2를 실행하는 문장이다.

- 결국 조건문 if else는 Stmt1과 Stmt2 둘 중의 하나는 선택하는 구문이다.

- 정수 n이 짝수인지 아니면 홀수인지 판단할 수 있는 조건식으로 (n % 2 == 0) 또는 (n % 2)이 주로 사용될 수 있다.

if 문에 사용하는 조건식

(n != 0) n이 0이 아니어야 참인
(n) 연산식으로 연산식 (n)과 같음.

ex) if (n % 2 != 0)
printf("홀수");
else
printf("짝수");

(n == 0) n이 0이어야 참인
(!n) 연산식으로 연산식 (!n)과 같음.

ex) if (n % 2 == 0)
printf("짝수");
else
printf("홀수");

중첩된 if

if문 내부의 if

- if문 내부에 if문이 존재하면 중첩된 if문이라 한다.

블록 표시와 else

- else는 문법적으로 같은 블록 내에서 else가 없는 가장 근접한 상위의 if문에 포함된 else로 구별된다.
- 들여쓰기는 첫 if의 조건식 (age >= 20)에 else로 되어 있으나 실제 문법적으로 else는 두 번째 if (age >= 65)의 else이다.

Date. Page.

6-3 다양한 선택 Switch 문

정수 또는 문자 선택

- 용광 if else가 여러 번 계속 반복되는 것을 좀 더 간략하게 구현

- Switch 문은 주어진 연산식이 문자형 또는 정수형이라면 그 값에 따라 case 의 상수값과 일치하는 부분의 문장들을 수행하는 선택 구문이다.

```
switch (exp) {
  case 상수1:
    stmt1;
    break;
  default:
    stmt2;
    break;
}
```

- 식의 결과는 문자형 또는 정수형이어야 한다.
- 상수 1은 정수 또는 문자형의 상수이어야 한다.
- break 안나오면 switch 문 종료
- 위의 case 값과 일치하지 않으면 default 이하의 문장 stmt2를 실행한다.

break의 적절한 사용

- Switch 문에서 주의할 것 중 하나로 case 마다 정수 상수 콜마로 구분하여 여러 개 나열할 수 없다.

- case 문 내부에 break 문이 없다면 일치하는 case 문을 실행하고, break 문을 만나기 전까지 다음 case 내부 문장을 실행한다.

default의 위치

- 일반적으로 Switch 문에서 ~~default~~ default는 생략 될 수 있으며 또 그 위치도 제한이 없다.

- default를 위치시킨 이후에 다른 케이스가 없다면 break 반드시 안적

Date Page

07 반복

반복의 개념과 구문 종류

- 반복은 말 그대로 같거나 비슷한 일을 여러 번 수행하는 것이다.
- 반복과 같은 의미로 순환 (loop, 루프) 이라는 표현도 함께 사용
- C 언어는 while, do while, for 세 가지 종류의 반복 구문을 제공
- 반복조건을 만족하면 일정하게 반복되는 부분을 반복몸체라고 한다.

while문 구조와 제어흐름

- 문장 while (cond) statement 은 반복조건인 cond를 평가하여 0이 아니면 (참) 반복몸체인 statement를 실행하고 다시 반복조건 cond를 평가하여 while 문 종료시까지 반복한다.
  - 반복이 실행되는 statement를 반복몸체라 부르며, 필요하면 블록으로 구성

등의 횟수만 3개로 고정된 while

```
int count = 1;
while (count <= 3)
{
    printf("C 언어 재미있어요!\n");
    count++;
}
```

=> 출력 3번

상수 3은 최대 반복회수를 지정하는 상수

반복몸체에서 제어변수 count 회수만큼 반복을 위해 count를 1 증가시키는 count++ 문장이 반드시 필요하다.

morning glory

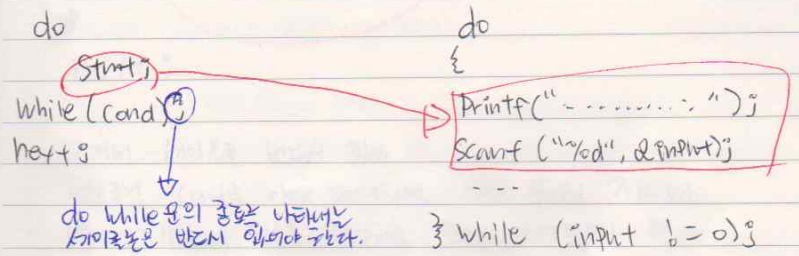
Date. Page.

7-2

do while문 구조와 제어흐름

- while문은 반복전에 반복조건을 평가한다.
  이와 다르게 do while문은 반복몸체 수행 후에 반복조건을 검사한다.

- 일반 do Stmt; while (cond); 는 가장 먼저 Stmt를 실행한 이후 반복조건인 cond를 평가하여 0이 아니면 (참) 다시 반복몸체인 Stmt;를 실행하고, 0이면 (거짓) do while 문을 종료한다.



do while문의 종료후 다음에는 세미콜론은 반드시 있어야 한다.

- 특히 반복횟수가 정해지지 않고 입력 받은 자료값에 따라 반복 수행의 여부를 결정하는 구문에 유용하다.

- 반복몸체에 특별히 할 일이 없는 경우, do while의 몸체는 적어도 한 번은 실행되는 특징이 있다.

센티넬 값 검사에 유용

-입력 후에 반복 검사를 진행하는 처리 과정으로 do while문과 적합

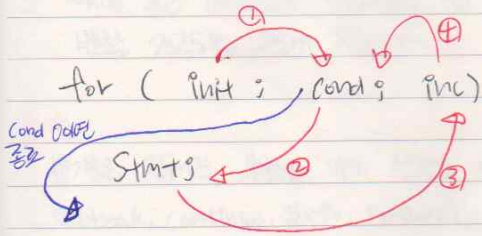
- do while 반복문은 이러한 센티넬 값 검사에 유용하게 사용된다.

morning glory

Date. Page.

for문 구조와 제어흐름

- 반복문 for (init; cond; inc) Stmt; 에서 init 에서는 주로 초기화가 이루어지며, cond 에서는 반복조건을 검사하고, inc 에서는 주로 반복을 결정하는 제어변수의 증감을 수행한다.

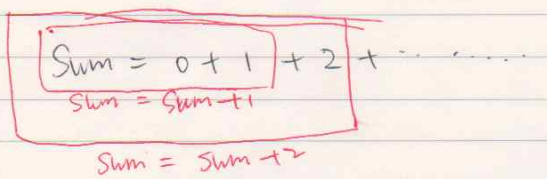


- 2개의 세미콜론 반드시 필요
- 반복조건 cond를 아예 지정하면 반복은 무한히 계속된다.
- 보통이 반복의 횟수는 제어하는 변수를 제어변수라 한다.

for 문의 한 주기

- 초기하는 제어변수 i 값은 계속 증가하며 변수 Sum에 누적

- for (i=1, Sum=0; i<10; i++)와 같이 초기화 부분에 콤마연산자를 이용하여 바꿈



morning glory

Date. Page.

for 문과 while 문의 비교

- for 문은 주로 반복회수를 제어하는 제어변수를 사용해서 초기화와 증감식이 있는 반복문에 적합하다.

- while 문은 반복회수가 정해지지 않고 특정한 조건에 따라 반복을 결정하는 구문에 적합하다.

분기문

- 분기문은 정해진 부분으로 바로 실행을 이동하는 기능을 수행한다.
  (break, continue, goto, return)

- break
  ① 반복내부에서 반복을 종료하려면 break 문장 사용
  ② 중첩된 반복에서의 break는 자신이 속한 가장 근접한 반복에서 반복을 종료

중첩된 for 문

- 반복문 내부에 반복문이 또 있는 구문을 중첩된 반복문이라 한다.

내부반복과 외부반복의 의존

- 외부반복에서 변수 i는 1에서 5까지 반복한다.

- 내부반복에서 제어변수 j는 1에서 외부반복의 제어변수 i까지 반복

Date Page

08 포인터 변수 기초

주소개념

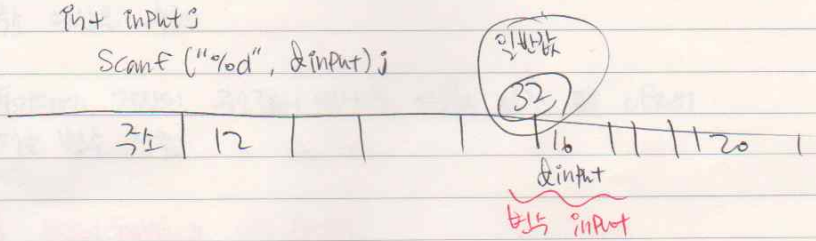
- 메모리 공간은 8비트인 1 바이트마다 고유한 주소(address)가 있다.

- 메모리 주소는 1바이트마다 1씩 증가한다.
  메모리 주소는 저장 장소인 변수이름과 함께 기억 장소를 참조하는 또 다른 방법이다.

- 주소는 변수이름과 같이 저장장소를 참조하는 하나의 방법이다.

- 바로 &가 피연산자인 변수의 메모리 주소를 반환하는 주소연산자

```
int input;
scanf("%d", &input);
```



- 변수의 주소값은 형식지정자 %u 또는 %d로 직접 출력할 수 있다.
  그러나 최근 비주얼 스튜디오에서는 결과가 반환하는 주소값은 int 또는 unsigned로 변환하여 출력한다.
  만일 16진수로 출력하려면 %p를 사용한다.

포인터 변수 선언

- 자료형과 포인터 변수 이름 사이에 연산자 *를 삽입한다.

Date Page

간접참조자 *

- 포인터 변수가 갖는 주소로 그 주소의 위치에 변수를 참조

- 포인터 변수가 가리키고 있는 변수를 참조하려면 간접참조자 *를 사용한다.

(Ex)
```
int data1 = 100, data2;
int *p;
Printf("간접참조 출력 : %d \n", *Printf);

*Printf = 2008
```

주소 연산

- 포인터 변수는 간단한 더하기와 뺄셈 연산으로 이웃한 변수의 주소 연산은 수행

- 포인터에 저장된 주소값의 연산으로 이웃한 이전 혹은 이후의 다른 변수 참조

8-3 포인터 형변환과 다중 포인터

명시적 형변환

- 포인터 변수는 동일한 자료끼리만 대입이 가능하다. 만일 대입문에서 포인터의 자료형이 다르면 경고가 발생한다.

다중 포인터

- 포인터 변수의 주소값을 갖는 변수를 이중 포인터라 한다.
- 포인터의 포인터를 또 다중 포인터 라고 한다.

```
int i = 20;
int *pi = &i;
int **dPi = &Pi;
```
이중 포인터

morning glory

Date. Page.

04 배열

배열의 특징성

- 배열은 여러 변수들이 가진 배열이름과 인덱스 값이 연속된 메모리에 저장되는 구조이다.

- 배열을 이용하면 변수를 일일이 선언하지 않아도 관리할 수 있고, 배열을 구성하는 각각의 변수를 참조하는 방법을 간편하여, 반복 구문으로 쉽게 참조할 수 있다.

배열 선언구문

- 배열선언은 int data[10]과 같이 원소자료형 배열이름[배열크기]; 이다.

- 배열선언 시 초기값 지정이 없다면 반드시 배열크기는 양의 정수로 명시.

배열 선언 최적화

- 배열을 선언하면서 동시에 원소값을 순차적 저장하는 배열선언 최적화 방법 제공

- 배열선언 최적화 구문은 배열선언을 하면서 대입연산자를 이용하여 중괄호 사이에 여러 원소값을 쉼표로 구분하여 기술하는 방법이다.

이차원 배열 선언

- 2개의 대괄호 필요
- 첫 번째 부터 행의 크기 지정

morning glory

Date. Page.

이차원 배열 선언 초기화

- 이차원 배열을 선언하면서 초기값을 지정하는 방법은 중괄호를 중첩하게 이용하는 방법과 일차원 배열 같이 하나의 중괄호를 사용하는 방법이 있다.

일차원 배열과 포인터

- 배열 score 에서 배열이름 score 자체가 배열 첫 원소의 주소값인 상수이다.

```
int score[] = {89, 90, 76};
```

- 배열이름 score는 &score[0] 와 같다.

```
*score = score[0]
```

배열 포인터 선언

- 열이 4인 이차원 배열 ary[][4]의 주소를 저장하려면 배열 포인터 변수 ptr는 반드시 int (*ptr)[4]; 로 선언해야 한다.

원소자료형 *변수이름;
변수이름 = 배열이름;
또는
원소자료형 *변수이름 = 배열이름;

```
int a[] = {8, 2, 8, 1, 3};
int *p = a;
```

배열크기 계산방법

- (sizeof(배열이름) / sizeof(배열원소)) 의 결과는 배열크기이다.

morning glory

Date Page

10 함수 기초

함수 개념

- 함수는 필요한 입력을 받아 특정한 연산 기능을 수행한 후 결과를 반환하는 프로그램 단위

함수정의 구문

- 함수정의, 함수헤더, 함수몸체로 구성된다.

지역 특성

- 함수내에서 자신 함수를 호출하는 함수를 재귀 함수라 한다.
  메모리 사용도 많고 시간도 오래걸림!!

함수 rand()

- 특정한 나열 순서나 규칙을 가지지 않는 연속적인 임의의 수를 나타내 한다.
  ↳ 0에서 32767 사이의 정수 중에서 임의의 하나의 정수 반환.

함수 srand()

- 1970년 1월 1일 이후 현재까지 경과된 시간을 초 단위로 반환하는 함수이다.
  ↳ 함수 time(NULL)

morning glory