

4과목-프로그래밍 언어 활용

(Part 2. 프로그래밍 언어 활용-2)

프로그래밍 언어 활용 총 파트

프로그래밍 언어 활용 4과목은 총 3Part로 이루어져 있다.

1장 서버 프로그램 구현(0.69%)

2장 프로그래밍 언어 활용(44.83%)

3장 응용 SW 기초 기술 활용(54.48%)

프로그래밍 언어 활용

프로그래밍 언어 활용 Part는 17개의 섹션으로 구성되어 있다.

001 데이터 타입

002 변수

003 연산자

004 데이터 입·출력

005 제어문

006 반복문

007 배열과 문자열

008 포인터

009 Python의 기초

010 Python의 활용

011 절차적 프로그래밍 언어 012 객체지향프로그래밍 언어

013 스크립트 언어

014 선언형 언어

015 라이브러리

016 예외 처리

017 프로토타입

4. 프로그래밍 언어 활용-SEC_03(연산자)

1) 산술 연산자

; 산술 연산자는 가, 감, 승, 제 등의 산술 계산에 사용되는 연산자를 말한다.

- 산술 연산자에는 일반 산술식과 달리 한 변수의 값을 증가하거나 감소시키는 증감 연산자가 있다.

연산자	의미	비고
+	덧셈	
-	뺄셈	
*	곱셈	
/	나눗셈	
%	나머지	
++	증가 연산자	• 전치 : 변수 앞에 증감 연산자가 오는 형태로 먼저 변수의 값을 증감시킨 후 변수를 연산에 사용한다(++a, --a). • 후치 : 변수 뒤에 증감 연산자가 오는 형태로 먼저 변수를 연산에 사용한 후 변수의 값을 증감시킨다(a++, a--).
--	감소 연산자	

4. 프로그래밍 언어 활용-SEC_03(연산자)

1) 산술 연산자

문제1) 다음에 제시된 산술 연산식의 결과를 적으시오.

번호	산술 연산식	결과
①	$10 + 15$	
②	$15 - 10$	
③	$3 * 5$	
④	$15 / 3$	
⑤	$15 \% 2$	
⑥	$3 - 7 \% 8 + 5$	
⑦	$-4 * 3 \% -5 / 2$	

결과 ① 25 ② 5 ③ 15 ④ 5 ⑤ 1 ⑥ 1 ⑦ -1

산술 연산자의 연산 우선 순위(높음 -> 낮음) : 증감 연산자 -> 산술 연산자(* / %) -> 산술 연산자(+ -)
산술 연산자 중 * / %는 우선순위가 같아 왼쪽에서 오른쪽 방향으로 놓인 순서대로 계산된다.

4. 프로그래밍 언어 활용-SEC_03(연산자)

1) 산술 연산자

문제2) 다음에 제시된 산술 연산식의 결과를 적으시오(단 정수형 변수 a=2, b=3, c=4, d=5와 같이 선언되었다고 가정한다.)

번호	산술 연산식	결과
①	b = ++b --c;	
②	c = ++b / b++;	
③	d = 10 % c++;	
④	b = 10 + ++a;	
⑤	c = 10 --d;	
⑥	c = ++a * b++;	

결과 : ①1 ②1 ③2 ④13 ⑤6 ⑥9

연산자 우선 순위(높음 -> 낮음)

증감 연산자 -> 산술 연산자(* / %) -> 산술 연산자(+ -) -> 시프트 연산자 -> 관계 연산자(< <= > >=) ->

관계 연산자(==, !=) -> 비트 연산자(& -> ^ -> |) -> 논리 연산자(&& -> ||) -> 조건(삼항) 연산자 -> 대입 연산자

4. 프로그래밍 언어 활용-SEC_03(연산자)

2) 관계 연산자

; 관계 연산자는 두 수의 관계를 비교하여 참(true) 또는 거짓(false)을 결과로 얻는 연산자이다.

- 거짓은 0, 참은 1로 사용되지만 외의 모든 숫자도 참으로 간주된다.

연산자	의미
==	같다
!=	같지 않다
>	크다
>=	크거나 같다
<	작다
<=	작거나 같다

관계 연산자는 왼쪽을 기준으로 "왼쪽이 크다", "왼쪽이 크거나 같다"로 해석하면 된다.

4. 프로그래밍 언어 활용-SEC_03(연산자)

2) 관계 연산자

문제) 다음 관계 연산식의 결과를 적으시오(단 정수형 변수 $a=5$, $b=10$ 으로 선언되었다고 가정한다.)

번호	관계 연산식	결과
①	$a == 10$	
②	$b != 10$	
③	$a > 10$	
④	$b >= 10$	
⑤	$a < 10$	
⑥	$b <= 10$	

결과 ① 0 ② 0 ③ 0 ④ 1 ⑤ 1 ⑥ 1

4. 프로그래밍 언어 활용-SEC_03(연산자)

3) 비트 연산자

; 비트 연산자는 비트별(0, 1)로 연산하여 결과를 얻는 연산자이다.

연산자	의미	비고
&	and	모든 비트가 1일 때만 1
^	xor	모든 비트가 같으면 0, 하나라도 다르면 1
	or	모든 비트 중 한 비트라도 1이면 1
~	not	각 비트의 부정, 0이면 1, 1이면 0
<<	왼쪽 시프트	비트를 왼쪽으로 이동
>>	오른쪽 시프트	비트를 오른쪽으로 이동

4. 프로그래밍 언어 활용-SEC_03(연산자)

3) 비트 연산자

문제) 다음 비트 연산식의 결과를 적으시오(단 정수형 변수 $a=5$, $b=7$ 으로 선언되었다고 가정한다.)

번호	비트 연산식	결과
①	$a \& b$	
②	$a b$	
③	$a \wedge b$	
④	$\sim b$	
⑤	$a \gg 1$	
⑥	$b \ll 3$	

결과 ① 5 ② 7 ③ 2 ④ -8 ⑤ 2 ⑥ 56

4. 프로그래밍 언어 활용-SEC_03(연산자)

4) 논리 연산자

; 논리 연산자는 두 개의 논리 값을 연산하여 참(true) 또는 거짓(false)을 결과로 얻는 연산자이다. 관계 연산자와 마찬가지로 거짓은 0, 참은 1이다.

연산자	의미	비고
!	not	부정
&&	and	모두 참이면 참
	or	하나라도 참이면 참

문제) 다음 논리 연산식의 결과를 적으시오(단 정수형 변수 a=2, b=3, c=0, d=1, e=1 로 선언되었다고 가정한다.)

번호	논리 연산식	결과
①	a > 3 && b > 2	
②	a > 3 b > 2	
③	!c	
④	a == 2 && b != 3	
⑤	a & b && c	
⑥	++d && --e	

결과 ① 0 ② 1 ③ 1 ④ 0 ⑤ 0 ⑥ 0

4. 프로그래밍 언어 활용-SEC_03(연산자)

5) 대입 연산자

; 연산 후 결과를 대입하는 연산식을 간략하게 입력할 수 있도록 대입 연산자를 제공한다. 대입 연산자는 산술, 관계, 비트, 논리 연산자에 모두 적용할 수 있다.

연산자	예	의미
<code>+=</code>	<code>a += 1</code>	<code>a = a + 1</code>
<code>-=</code>	<code>a -= 1</code>	<code>a = a - 1</code>
<code>*=</code>	<code>a *= 1</code>	<code>a = a * 1</code>
<code>/=</code>	<code>a /= 1</code>	<code>a = a / 1</code>
<code>%=</code>	<code>a %= 1</code>	<code>a = a % 1</code>
<code><<=</code>	<code>a <<= 1</code>	<code>a = a << 1</code>
<code>>>=</code>	<code>a >>= 1</code>	<code>a = a >> 1</code>

4. 프로그래밍 언어 활용-SEC_03(연산자)

6) 조건(삼항) 연산자

; 조건 연산자는 조건에 따라 서로 다른 수식을 수행한다.

형식

조건 ? 수식1 : 수식2; '조건'의 수식이 참이면 '수식1'을, 거짓이면 '수식2'를 실행한다.

문제) 다음 조건 연산식의 결과를 적으시오(단 정수형 변수 $a=1$, $b=2$, $c=3$, $d=4$ 와 같이 선언되었다고 가정한다.)

번호	조건 연산식	결과
①	$b *= a > b ? a : b;$	
②	$c -= a < b ? a - b : b - a;$	
③	$d \% = c < d ? c++ : d++;$	
④	$c += b < b ? ++a : b++;$	
⑤	$d /= d \% 3 ? a * b : d \% c;$	
⑥	$a += ++a \% b++ ? c * d : b / c;$	

결과 ① 4 ② 4 ③ 1 ④ 5 ⑤ 2 ⑥ 3

4. 프로그래밍 언어 활용-SEC_03(연산자)

7) 기타 연산자

연산자	의미
sizeof	자료형의 크기를 표시한다.
,(콤마)	<ul style="list-style-type: none">• 콤마로 구분하여 한 줄에 두 개 이상의 수식을 작성하거나 변수를 정의한다.• 왼쪽에서 오른쪽으로 순서대로 수행되며, 순서 연산자라 부르기도 한다.
(자료형)	<ul style="list-style-type: none">• 사용자가 자료형을 다른 자료형으로 변환할 때 사용하는 것으로, cast(캐스트) 연산자라고 부른다.• 변환할 자료형을 괄호 안에 넣어서 변환할 값이나 변수명 앞에 놓는다. 예 a = (int)1.3 + (int)1.4; 1.3을 정수형으로 변환한 값 1과 1.4를 정수형으로 변환한 값 1이 더해진 2가 a에 저장된다.

4. 프로그래밍 언어 활용-SEC_03(연산자)

8) 연산자 우선순위

- 한 개의 수식에 여러 개의 연산자가 사용되면 기본적으로 아래 표의 순서대로 처리된다.
- 아래 표의 한 줄에 가로로 나열된 연산자는 우선순위가 같기 때문에 결합규칙에 따라 <-는 오른쪽에 있는 연산자부터, ->는 왼쪽에 있는 연산자부터 차례로 계산된다.

대분류	중분류	연산자	결합규칙	우선 순위
단항 연산자	단항 연산자	!(논리 not) ~ (비트 not) ++ (증가) -- (감소) sizeof (기타)	←	높음 ↑
이항 연산자	산술 연산자	* / %(나머지)	→	
		+ -		
	시프트 연산자	<< >>		
	관계 연산자	< <= >= >		
		== (같다) != (같지 않다)		
	비트 연산자	& (비트 and) ^ (비트 xor) (비트 or)		
논리 연산자		&& (논리 and) (논리 or)		
삼항 연산자	조건 연산자	? :	→	↓ 낮음
대입 연산자	대입 연산자	= += -= *= /= %= <<= >>= 등	←	
순서 연산자	순서 연산자	,	→	

4. 프로그래밍 언어 활용-SEC_03(연산자)

8) 연산자 우선순위

문제) 다음 연산식의 결과를 적으시오(단 정수형 변수 $a=3$, $b=4$, $c=5$, $d=6$ 로 선언되었다고 가정한다.)

번호	연산식	결과
①	$a * b + c >= d \ \&\& \ d / a - b != 0$	
②	$d \% b + ++a * c -- \ \ c -- -a >= 10$	

결과 ① 1 ② 1

프로그래밍 언어 활용-SEC_03(연산자) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(연산자)

1. 다음 JAVA 프로그램이 실행되었을 때의 결과는?

```
public class Operator {
    public static void main(String[] args) {
        int x=5, y=0, z=0;
        y = x++;
        z = --x;
        System.out.print(x + ", " + y + ", " + z);
    }
}
```

- ① 5, 5, 5 ② 5, 6, 5
- ③ 6, 5, 5 ④ 5, 6, 4

증감 연산자의 위치에 따른 값을 파악을 할 수 있어야 한다.

2. C언어에서 산술 연산자가 아닌 것은?

- ① % ② *
- ③ / ④ =

산술 연산자의 종류

+ : 덧셈, - : 뺄셈, * : 곱셈, / : 나눗셈, % : 나머지

++ : 증가 연산자, -- : 감소 연산자

증감 연산자에는 전위(전치)는 변수 앞에 증감 연산자가 오는 형태로 먼저 변수의 값을 증감 시킨 후 연산에 사용한다.

3. 다음 C언어 프로그램이 실행되었을 때의 결과는?

```
#include <stdio.h>
int main(int argc, char *argv[] ) {
    int a = 4;
    int b = 7;
    int c = a | b;

    printf("%d", c);
    return 0;
}
```

- ① 3 ② 4
- ③ 7 ④ 10

위의 문제는 비트 or(|)문제이며 비트 or는 두 비트 중에서 한 비트라도 1이라면 1이 되는 비트 연산자이다.

4. 다음 C언어 프로그램이 실행되었을 때의 결과는?

```
#include <stdio.h>
int main(void) {
    int a = 3, b = 4, c = 2;
    int r1, r2, r3;

    r1 = b <= 4 || c == 2;
    r2 = (a > 0) && (b < 5);
    r3 = !c;

    printf("%d", r1+r2+r3);
}
```

프로그래밍 언어 활용-SEC_03(연산자) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(연산자)

5. 다음 중 JAVA에서 우선순위가 가장 낮은 연산자는?

- ① -- ② %
③ & ④ =

연산자의 우선순위

단항 연산자 : !(논리, not), ~(비트, not), ++(증가), --(감소), sizeof(기타)

산술 연산자 : *, /, %(나머지) -> +, -

시프트 연산자 : <<, >>

관계 연산자 : <, <=, >, >= -> ==(같다), !=(같지 않다)

비트 연산자 : &(비트 and) -> ^(비트 xor) -> |(비트 or)

논리 연산자 : &&(논리 and) -> ||(논리 or)

조건 연산자 : ? :

대입 연산자 : =, +=, -=, *=, /=, %=, <<=, >>=

순서 연산자 : ,

6. C언어에서 연산자 우선순위가 높은 것에서 낮은 것으로 바르게 나열된 것은?

- ㉠ () ㉡ == ㉢ <

7. C언어에서 비트 논리 연산자에 해당하지 않는 것은?

- ① ^ ② ?
③ & ④ ~

비트 연산자의 종류

&(and) : 모든 비트가 1일 때만 1

^(xor) : 모든 비트가 같으면 0, 하나라도 다르면 1

|(or) : 모든 비트 중 한 비트라도 1이면 1

~(not) : 각 비트의 부정, 0이면 1, 1이면 0

<<(왼쪽 시프트) : 비트를 왼쪽으로 이동

>>(오른쪽 시프트) : 비트를 오른쪽으로 이동

8. 다음 JAVA 프로그램이 실행되었을 때의 결과는?

```
public class ovr {  
    public static void main(String[] args) {  
        int a = 1, b = 2, c = 3, d = 4;  
        int mx, mn;  
        mx = a < b ? b : a;  
        if (mx == 1) {  
            mn = a > mx ? b : a;  
        }  
        else {  
            mn = b < mx ? d : c;  
        }  
        System.out.println(mn);  
    }  
}
```

프로그래밍 언어 활용-SEC_03(연산자) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(연산자)

9. C언어에서 정수 변수 a, b에 각각 1, 2가 저장되어 있을 때 다음 식의 연산 결과로 옳은 것은?

```
a < b + 2 && a << 1 <= b
```

- ① 0 ② 1
③ 3 ④ 5

10. 다음 C언어 프로그램이 실행되었을 때, 실행 결과는?

```
#include <stdio.h>
int main(int argc, char* argv[ ]) {
    int n1 = 1, n2 = 2, n3 = 3;
    int r1, r2, r3;
    r1 = (n2 <= 2) || (n3 > 3);
    r2 = !n3;
    r3 = (n1 > 1) && (n2 < 3);
    printf("%d", r3 - r2 + r1);
    return 0;
}
```

- ① 0 ② 1
③ 2 ④ 3

11. 다음 프로그램의 실행 결과에 의해 변수 a와 b에 저장된 값은?
(단, "<<"는 왼쪽 시프트, ">>"는 오른쪽 시프트를 의미한다.)

```
int a = 16, b = 64;
a = a >> 2;
b = b << 2;
```

- ① a = 4, b = 256 ② a = 8, b = 128
③ a = 32, b = 32 ④ a = 64, b = 16

X << 2 : X * 2의 n승

X >> 2 : X / 2의 n승

12. 다음 C언어 프로그램이 실행되었을 때, 실행 결과는?

```
#include <stdio.h>
int main(int argc, char* argv[ ]) {
    int a = 5, b = 3, c = 12;
    int t1, t2, t3;
    t1 = a && b;
    t2 = a || b;
    t3 = !c;
    printf("%d", t1 + t2 + t3);
    return 0;
}
```

- ① 0 ② 2

프로그래밍 언어 활용-SEC_03(연산자) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(연산자)

13. C언어에서 두 개의 논리 값 중 하나라도 참이면 1을, 모두 거짓이면 0을 반환하는 연산자는?

- ① || ② &&
- ③ ** ④ !=

논리 연산자의 종류와 의미

!(not) : 부정

&&(and) : 모두 참이면 참이며 하나라도 거짓이면 거짓을 리턴

||(or) : 하나라도 참이면 참, 둘 다 거짓이면 거짓을 리턴

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

1) C언어의 표준 입·출력 함수의 개요

; 표준 입출력 함수(Input-Output Functions)란 키보드로 입력 받아 화면으로 출력할 때 사용하는 함수로, 대표적으로 scanf(), getchar(), gets(), printf(), putchar(), puts() 등이 있다.

2) scanf() 함수

; scanf() 함수는 C언어의 표준 입력 함수로, 키보드로 입력 받아 변수에 저장하는 함수이다.

형식

scanf(서식 문자열, 변수의 주소)	<ul style="list-style-type: none">• 서식 문자열 : 입력받을 데이터의 자료형을 지정한다.• 변수의 주소 : 데이터를 입력받을 변수를 적는다. 변수의 주소로 입력 받아야 하기 때문에 변수에 주소연산자 &를 붙인다.
-----------------------	---

예) scanf("%3d", &a);

- ▶ % : 서식 문자임을 지정
- ▶ 3 : 입력 자릿수를 3자리로 지정
- ▶ d : 10진수로 입력
- ▶ &a : 입력 받은 데이터를 변수 a의 주소에 저장

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

2) scanf() 함수

특징

- 입력 받을 데이터의 자료형, 자릿수 등을 지정할 수 있다.
- 한 번에 여러 개의 데이터를 입력 받을 수 있다.
- 서식 문자열과 변수의 자료형은 일치해야 한다.

예) `scanf("%d %f", &i, &j);` -> %d와 i, %f와 j는 자료형이 일치해야 한다.

서식 문자열

; 서식 문자열은 `printf()`함수로 출력할 때도 동일하게 적용된다.

서식 문자열	의미
%d	정수형 10진수를 입·출력하기 위해 지정한다.
%u	부호 없는 정수형 10진수를 입·출력하기 위해 지정한다.
%o	정수형 8진수를 입·출력하기 위해 지정한다.
%x	정수형 16진수를 입·출력하기 위해 지정한다.
%c	문자를 입·출력하기 위해 지정한다.
%s	문자열을 입·출력하기 위해 지정한다.
%f	소수점을 포함하는 실수를 입·출력하기 위해 지정한다.
%e	지수형 실수를 입·출력하기 위해 지정한다.
%ld	long형 10진수를 입·출력하기 위해 지정한다.

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

2) scanf() 함수

서식 문자열

서식 문자열	의미
%lo	long형 8진수를 입·출력하기 위해 지정한다.
%lx	long 형 16진수를 입·출력하기 위해 지정한다.
%p	주소를 16진수로 입·출력하기 위해 지정한다.

3) JAVA에서의 표준 입력

; JAVA에서 키보드로 입력 받은 값을 변수에 저장하려면 먼저 Scanner 클래스를 이용해 키보드로부터 값을 입력 받는 객체(참조) 변수를 생성한 후 이를 사용해야 한다.

형식

```
❶ Scanner scan01 = new Scanner(System.in);  
❷ inNum = scan01.nextInt( );
```

① 객체(참조) 변수 생성

- Scanner : 입력에 사용할 객체(참조) 변수를 생성할 때 사용하는 클래스 이름이다. 그대로 적어준다.
- scan01 : 객체(참조) 변수명이다. 사용자 임의로 적어준다.
- new : 객체(인스턴스) 생성 예약어이다. 그대로 적어준다.

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

3) JAVA에서의 표준 입력

① 객체(참조) 변수 생성

- Scanner() : 클래스의 이름입니다. ()를 붙여 그대로 적어준다.
- System.in : 표준 입력장치, 즉 키보드를 의미한다. 키보드로부터 값을 입력 받는 객체(참조) 변수를 생성할 것이므로 그대로 적어준다.

② 객체 변수 활용

- inNum : 입력 받은 값을 저장할 변수이다. 이 변수는 미리 선언되어 있어야 한다.
- scan01.nextInt()
 - scan01 : 입력에 사용할 객체(참조) 변수 이름이다. 객체 변수 생성 시 사용한 객체 변수 이름과 동일해야 한다.
 - nextInt() : 입력 받은 값을 정수형으로 반환한다.

Scanner 클래스의 입력 메소드

- next() : 입력 값을 문자열로 반환(공백을 만나면 한 단어만 반환)
- nextLine() : 입력 받은 라인 전체를 문자열로 반환(공백을 만나도 전체 다 반환)
- nextInt() : 입력 값을 정수형으로 반환
- nextFloat() : 입력 값을 실수형으로 반환

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

3) JAVA에서의 표준 입력

문제) scanf()함수를 이용하여 다음과 같이 데이터를 입력할 경우 변수에 기억되는 결과를 쓰시오.

번호	코드	입력 데이터	결과
①	scanf("%d", &i);	20	
②	scanf("%2d", &i);	125	
③	scanf("%4f", &i);	12.123	
④	scanf("%c", &a);	KOREA	
⑤	char b[8]; scanf("%4c", b);	KOREA!FIGHTING	
⑥	char b[8]; scanf("%s", b);	KOREA!FIGHTING	
⑦	char b[8], c[8]; scanf("%s %s", b, c);	KOREA FIGHTING	
⑧	scanf("%3d %5f", &i, &j);	123456789	

배열명은 배열의 시작 주소를 의미하므로 배열명 앞에는 &를 붙이지 않는다.

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

4) printf() 함수

; printf() 함수는 C언어의 표준 출력 함수로, 인수로 주어진 값을 화면에 출력하는 함수이다.

형식

printf(서식 문자열, 변수)

- 서식 문자열 : 변수의 자료형에 맞는 서식 문자열을 입력한다.
- 변수 : 서식 문자열의 순서에 맞게 출력할 변수를 적는다. scanf()와 달리 주소 연산자 &를 붙이지 않는다.

예) printf("%-8.2f", 200.2);

(V는 빈 칸을 의미함)

결과 : 200.20VV

- ▶ % : 서식 문자임을 지정
- ▶ - : 왼쪽부터 출력
- ▶ 8 : 출력 자릿수를 8자리로 지정
- ▶ 2 : 소수점 이하 2자리로 지정
- ▶ f : 실수로 출력

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

4) printf() 함수

주요 제어문자

- 제어문자란 입력 혹은 출력 내용을 제어하는 문자이다.

문자	의미	기능
\n	new line	커서를 다음 줄 앞으로 이동한다.
\b	backspace	커서를 왼쪽으로 한 칸 이동한다.
\t	tab	커서를 일정 간격 띄운다.
\r	carriage return	커서를 현재 줄의 처음으로 이동한다.
\0	null	널 문자를 출력한다.
\'	single quote	작은따옴표를 출력한다.
\"	double quote	큰따옴표를 출력한다.
\a	alert	스피커로 벨 소리를 출력한다.
\\	backslash	역 슬래시를 출력한다.
\f	form feed	한 페이지를 넘긴다.

예) printf("%d\n", a); -> a의 값을 정수형 10진수로 출력한 후 다음 줄로 이동한다.

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

5) JAVA에서의 표준 출력

; JAVA에서 값을 화면에 출력할 때는 System클래스의 서브 클래스인 out클래스의 메소드 print(), println(), printf() 등을 사용하여 출력한다.

형식 1 : 서식 문자열에 맞게 변수의 내용을 출력한다.

System.out.printf(서식 문자열, 변수)

- printf() 메소드는 C언어의 printf()함수와 사용법이 동일하다.

예) System.out.printf("%-8.2f", 200.2);

(V는 빈 칸을 의미함)

결과 : 200.20VV

- ▶ % : 서식 문자임을 지정
- ▶ - : 왼쪽부터 출력
- ▶ 8 : 출력 자릿수를 8자리로 지정
- ▶ 2 : 소수점 이하를 2자리로 지정
- ▶ f : 실수로 출력

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

5) JAVA에서의 표준 출력

형식 2 : 값이나 변수의 내용을 형식 없이 출력한다.

System.out.print()

- 문자열을 출력할 때는 큰따옴표로 묶어줘야 한다.
- 문자열 또는 문자열 변수를 연속으로 출력할 때는 +(연결 연산자)를 이용한다.
- '숫자 + 숫자'는 두 숫자를 합한 값을 출력하지만, '문자열 + 숫자' 또는 '숫자 + 문자열'과 같이 문자열과 숫자가 섞인 경우에는 모두 문자열로 인식되므로 값이 붙어서 출력된다.

예1) System.out.print("abc123" + "del");

결과 : abc123def

예2) System.out.print("abc" + 12 + 34);

결과 : abc1234

예3) System.out.print("abc" + (12 + 34));

결과 : abc46

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

5) JAVA에서의 표준 출력

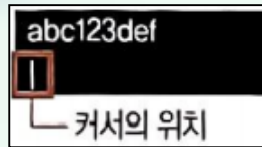
형식 3 : 값이나 변수의 내용을 형식 없이 출력한 후 커서를 다음 줄의 처음으로 이동한다.

`System.out.println()`

- `println()` 메소드는 출력 후 다음 줄로 이동한다는 것을 제외하면 `print()` 메소드와 사용법이 동일하다.

예) `System.out.println("abc123" + "def");`

결과 :



4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

문제1) printf()함수를 이용하여 다음과 같이 데이터를 출력할 경우 결과를 쓰시오.(V는 빈칸을 의미함)

번호	코드	결과
①	printf("%d", 2543);	
②	printf("%3d", 2543);	
③	printf("%6d", 2543);	
④	printf("%-6d", 2543);	
⑤	printf("%06d", 2543);	
⑥	printf("%f", 245.2555);	
⑦	printf("%.3f", 245.2555);	
⑧	printf("%.8,2f", 245.2555);	
⑨	printf("%e", 25.43);	
⑩	printf("%.3s", "help me");	
⑪	printf("%3s", "help me");	
⑫	printf("%.6s", "help me");	
⑬	printf("%-8.6s", "help me");	
⑭	printf("250은 10진수로 %d\ 8진수로 %o\n", 250, 250);	
⑮	printf("a=%8.2f\ b=%e\n", 125.23f, 3141.592e-1);	
⑯	printf("\A\는 문자로 %c, 아스키코드로 %d\n", 'A', 'A');	

25.43을 정규화하면 정수 부분이 한 자리만 남도록 소수점의 위치를 조절한 후 소수점이 이동한 자릿수 만큼 e뒤에 표현한다.
25.43을 정규화하면 2.543인데 기본적으로 소수점 자리는 6자리로 표현하므로 2.543000이며 이는 25.43에서 소수점 자리가 왼쪽으로 한 자리 이동하였기에 2.543000e+01로 표현한다.

결과: ① 2543 ② 2543 ③ V V 2543 ④ 2543 V V ⑤ 002543 ⑥ 245.255500 ⑦ 245.256
⑧ V V 245.26 ⑨ 2.543000e+01 ⑩ hel ⑪ help me ⑫ V V help m ⑬ help m V V
⑭ 250은 10진수로 250 8진수로 372 ⑮ a=V V 125.23 b=3.141592e+02
⑯ 'A'는 문자로 A, 아스키코드로 65


4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

문제2) 다음과 같이 scanf()함수로 값을 입력 받아 printf()함수로 출력할 경우 결과를 쓰시오.(V는 빈칸을 의미함)

번호	코드	입력 데이터	코드	결과
①	char a[5]; scanf("%d %e %s", &i, &j, a);	5468 3.483E-2 GOOD	printf("%4d %f %2s", i, j, a);	
②	scanf("%e", &i);	123.45E-1	printf("%f\\t %e\\n", i, i);	
③	scanf("%d", &i);	300	printf("[%5d], [%-5d], [%05d]", i, i, i);	
④	scanf("%2d \\n \\t %3d", &i, &j);	12345678	printf("i=%d j=%d\\n", i, j);	
결과 ① 5468 0.034830 GOOD ② 12.345000 1.234500e+01 ③ [V V 300], [300V V], [00300] ④ i=12 j=345				

4. 프로그래밍 언어 활용-SEC_04(데이터 입·출력)

6) 기타 표준 입·출력 함수

입력	getchar()	키보드로 한 문자를 입력받아 변수에 저장하는 함수
	gets()	키보드로 문자열을 입력받아 변수에 저장하는 함수로,  를 누르기 전까지를 하나의 문자열로 인식하여 저장함
출력	putchar()	인수로 주어진 한 문자를 화면에 출력하는 함수
	puts()	인수로 주어진 문자열을 화면에 출력한 후 커서를 자동으로 다음 줄 앞으로 이동하는 함수

문제) 다음의 코드를 실행하여 데이터를 입력할 경우 출력되는 결과를 쓰시오.

번호	코드	입력 데이터	코드	결과
①	a = getchar();	SUN DAY	putchar(a);	
②	putchar('G');			
③	putchar('G'+1);			
④	gets(b);	SUN DAY	puts(b);	
⑤	puts ("SUN DAY");			

결과 ① S ② G ③ H ④ SUN DAY ⑤ SUN DAY

프로그래밍 언어 활용-SEC_04(데이터 입·출력) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(데이터 입·출력)

1. Java에서 사용되는 출력 함수가 아닌 것은?

- ① System.out.print()
- ② System.out.println()
- ③ System.out.printing()
- ④ System.out.printf()

JAVA에서 값을 화면에 출력할 때는 System클래스의 서브 클래스인 out클래스의 메소드인 print(), println(), printf() 등을 사용하여 출력한다.

형식 1 : 서식 문자열에 맞게 변수의 내용을 출력한다.

System.out.printf(서식 문자열, 변수);

- printf() 메소드는 C언어의 printf()함수와 사용법이 동일하다.

형식 2 : 값이나 변수의 내용을 형식 없이 출력한다.

System.out.print()

- 문자열을 출력할 때는 큰따옴표로 묶어줘야 한다.
- 문자열 또는 문자열 변수를 연속으로 출력할 때는 +(연결 연산자)를 이용한다.

3. 다음 C언어 프로그램이 실행되었을 때의 결과는?

```
#include <stdio.h>
int main(int argc, char *argv[ ]) {
    char a;
    a = 'A' + 1;
    printf("%d", a);
    return 0;
}
```

- ① 1 ② 11
- ③ 66 ④ 9

'A'라는 문자는 메모리에 저장될 때 문자로 저장되는 것이 아니라 해당 문자의 아스키 코드 값으로 저장된다. 다시 말하면, 'A'는 'A'에 해당하는 아스키 코드 값인 65가 저장되는 것이다. 그러므로 a에는 'A'의 아스키 코드 값인 65에 1을 더한 값인 66을 저장하고 있는 것이다. 그리고 a에 저장된 66은 형식 지정자 "%d"로 출력하면 정수 66이 출력되고 형식 지정자가 "%c"로 지정하면 'A'의 다음 문자인 'B'가 출력된다.

4. 다음 중 C언어에서 문자열을 출력하기 위해 사용되는 것은?

- ① %x ② %d
- ③ %s ④ %h

프로그래밍 언어 활용-SEC_04(데이터 입·출력) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(데이터 입·출력)

5. 다음 C언어 프로그램을 실행한 결과 출력되는 값은?

```
#include<stdio.h>
void main (void)
{
    int a;
    a = 7;
    printf("%d", a+a);
}
```

- ① 11 ② 12
③ 13 ④ 14

6. 다음은 C언어 프로그램이다. 실행 결과는?

```
main( )
{
    printf("%2.1f\n", 123.45);
}
```

- ① 123.4 ② error
③ 123.5 ④ 23.4

'%2.1f'는 자리를 2자리를 확보하여 소수점 이하는 1자리까지 표시하라는 의미이다. 지정한 자릿수보다 출력할 값이 클 경우에는 정수 부분은 모두 표시하고, 소수점 이하 부분은 반올림

7. C언어에서 정수형 변수 a에 256이 저장되어 있다. 이를 7자리로 잡아 왼쪽으로 붙여 출력하려고 할 때, 적절한 printf() 내의 % 변환 문자 사용으로 알맞은 것은?

- ① %7f ② %7d
③ %-7d ④ %-7i

서식 변환 문자임을 나타내려면 %를 사용해야 하고, 왼쪽에 붙여 출력하려면 -을 사용, 7자리를 확보하려면 7을 기재하고 정수를 출력하려면 d를 차례로 지정하면 된다.

8. 다음에서 C언어의 입·출력 명령이 잘못 표현된 것은?

- ① getchar("%c", i);
② printf("%3d", i);
③ putchar(a);
④ scanf("%5d", &i);

getchar() 함수는 서식 문자열을 사용하지 아니한다.

기타 입·출력 함수의 종류

입력 부분

getchar() : 키보드로 한 문자를 입력 받아 변수에 저장하는 함수

gets() : 키보드로 문자열을 입력 받아 변수에 저장하는 함수로 Enter

프로그래밍 언어 활용-SEC_04(데이터 입·출력) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(데이터 입·출력)

9. 다음 중 잘못 구성된 것은?

```
main()  
{  
    int i;  
    float j;  
    char string[10];  
    scanf("%f %f\n", &i, &j);  
    scanf("%s\n", string);  
}
```

- ① float j;
- ② char string[];
- ③ scanf("%f %f\n", &i, &j);
- ④ scanf("%s\n", string);

scanf()함수에서는 서식 문자열과 저장할 변수의 자료형이 반드시 같아야 한다. i는 정수형 변수이기 때문에 'f'가 아닌 'd'로 서식 문자열을 지정해야 한다.

10. 다음 중, C언어 제어문자에서 커서를 다음 줄 앞으로 이동 하는 제어문자는 무엇인가?

- ① \b ② \t

11. JAVA언어의 Scanner클래스를 이용하여 정수를 입력 받고자 할 때 사용되는 메소드는?

- ① next() ② nextLine()
- ③ nextInt() ④ nextFloat()

JAVA에서 키보드로 입력 받은 값을 변수에 저장하려면 먼저 Scanner 클래스를 이용해 키보드로부터 값을 입력 받는 객체(참조) 변수를 생성한 후 .(접근 연산자)로 멤버 메소드를 사용해야 한다.

형식

```
int inNum = 0;
```

```
Scanner sc = new Scanner(System.in);
```

```
inNum = sc.nextInt();
```

```
sc.close();
```

① 객체(참조) 변수 생성

- Scanner : 표준 입력에 사용할 객체(참조) 변수를 생성할 때 사용하는 클래스 이름이다. 그대로 적어준다.

- sc : 객체(참조) 변수명이다. 주소를 저장하고 있다.

- new : 객체(인스턴스)를 메모리 공간 heap에 생성하는 예약어이다.

- Scanner() : 클래스의 생성자 호출하는 코드이다.

4. 프로그래밍 언어 활용-SEC_05(제어문)

1) 제어문의 개념

; 컴퓨터 프로그램은 명령어가 서술된 순서에 따라 무조건 위에서 아래로 실행되는데, 조건을 지정해서 진행 순서를 변경할 수 있다. 이렇게 프로그램의 순서를 변경할 때 사용하는 명령문을 제어문이라고 한다.

- 제어문의 종류에는 if문, 다중 if문, switch문, goto, 반복문 등이 있다.

2) 단순 if문

; if문은 조건에 따라서 실행할 문장을 달리하는 제어문이며, 단순 if문은 조건이 한 개일 때 사용하는 제어문이다.

- 조건이 참일 때만 실행할 문장을 지정할 수도 있고, 참과 거짓에 대해 각각 다른 실행문을 지정할 수도 있다.
- 형식1 : 조건이 참일 때만 실행한다.
 - 조건이 참일 때 실행할 문장이 하나인 경우

if(조건)

실행할 문장;

if는 조건 판단문에 사용되는 예약어이므로 그대로 적는다.
조건은 참(1) 또는 거짓(0)이 결과로 나올 수 있는 수식을 () 안에 입력한다.
조건이 참일 경우 실행할 문장을 적는다.

4. 프로그래밍 언어 활용-SEC_05(제어문)

2) 단순 if문

- 형식1 : 조건이 참일 때만 실행한다.
 - 조건이 참일 때 실행할 문장이 두 문장 이상인 경우

```
if(조건)
{
    실행할 문장1;      {} 사이에 조건이 참일 경우 실행할 문장을 적는다.
    실행할 문장2;
    :
}
```

예제1) a가 10보다 크면 a에서 10을 빼기

```
#include <stdio.h>
main() {
    int a = 15, b;
    if (a > 10)    //①      a가 10보다 크면 ②번 문장을 실행하고,
        b = a - 10; //②      아니면 ③번 문장으로 이동해서 실행을 계속한다.
    printf("%d\n", b); //③  여기서는 ①번의 조건식이 거짓일 경우 실행할 문장이
                           //없다. 조건 판단문을 벗어나면 무조건 ③번으로 온다.
}
```

결과 : 5

#include <stdio.h> : 표준 입·출력과 관련된 함수를 정의해 놓은 파일의 이름이다. 헤더 파일이라고 하는데, 사용하는 함수에 따라 포함시켜야 할 헤더 파일이 다르다. 여기서는 printf() 함수를 사용하기 때문에 포함시킨 것이다.

4. 프로그래밍 언어 활용-SEC_05(제어문)

2) 단순 if문

- 형식2 : 조건이 참일 때와 거짓일 때 실행할 문장이 다르다.
 - 조건이 참일 때 실행할 문장이 두 문장 이상인 경우

if(조건)	
실행할 문장1;	조건이 참일 경우 실행할 문장을 적는다. 참일 경우 실행할 문장이 두 문장 이상이면 {}를 입력하고 그 사이에 문장을 적는다.
else	
실행할 문장2;	조건이 거짓일 경우 실행할 문장을 적는다. 두 문장 이상인 경우 {}를 입력하고 그 사이에 문장을 적는다.

예제2) a가 b보다 크면 'a - b', 아니면 'b - a'를 수행하기

```
#include <stdio.h>
main() {
    int a = 10, b = 20, cha = ' ';
    if (a > b)
        cha = a - b;
    else
        cha = b - a;
    printf("%d\n", cha);
}
```

결과 10

4. 프로그래밍 언어 활용-SEC_05(제어문)

3) 다중 if문

; 다중 if문은 조건이 여러 개 일 때 사용하는 제어문이다.

● 형식1

if(조건1)	
실행할 문장1;	조건1이 참일 경우 실행할 문장을 적는다.
else if(조건2)	
실행할 문장2;	조건2가 참일 경우 실행할 문장을 적는다.
else if(조건3)	
실행할 문장3;	조건3이 참일 경우 실행할 문장을 적는다.
⋮	
else	
실행할 문장4;	앞의 조건이 모두 거짓일 경우 실행할 문장을 적는다.

4. 프로그래밍 언어 활용-SEC_05(제어문)

3) 다중 if문

예제1) 점수에 따라 등급 표시하기

```
#include <stdio.h>
main() {
    int jum = 85;
    if (jum >= 90)
        printf("학점은 A입니다.\n");
    else if (jum >= 80)
        printf("학점은 B입니다. \n");
    else if (jum >= 70)
        printf("학점은 C입니다.\n");
    else
        printf("학점은 F입니다.\n");
}
```

결과 학점은 B입니다.

4. 프로그래밍 언어 활용-SEC_05(제어문)

3) 다중 if문

- 형식2 : if문 안에 if문이 포함된다.

if(조건1)	
{	조건1이 참일 경우 실행할 문장의 시작점이다.
if(조건2)	
실행할 문장1;	조건2가 참일 경우 실행할 문장을 적는다.
else	
실행할 문장2;	조건2가 거짓일 경우 실행할 문장을 적는다.
}	
else	
실행할 문장3;	조건1이 거짓일 경우 실행할 문장을 적는다.

4. 프로그래밍 언어 활용-SEC_05(제어문)

3) 다중 if문

예제2) 홀수, 짝수 판별하기

```
#include <stdio.h>
main() {
    int a = 21, b = 10;
    if (a % 2 == 0)
        if (b % 2 == 0)
            printf("모두 짝수\n");
        else
            printf("a : 짝수, b : 홀수\n");
    else
        if(b % 2 == 0)
            printf("a:홀수, b: 짝수\n");
        else
            printf("모두 홀수\n");
}
```

결과 a:홀수, b:짝수

4. 프로그래밍 언어 활용-SEC_05(제어문)

4) switch문

; switch문은 조건에 따라 분기할 곳이 여러 곳인 경우 간단하게 처리할 수 있는 제어문이다.

● 형식

switch(수식) ❶	<ul style="list-style-type: none">• switch는 switch문에 사용되는 예약어로 그대로 입력한다.• 수식: '레이블' ~ '레이블n'의 값 중 하나를 도출하는 변수나 수식을 입력한다.
{ ❷	❷~❸번이 switch문의 범위이다. '('로 시작해서 ')'로 끝난다. 반드시 입력해야 한다.
case 레이블1: ❸	<ul style="list-style-type: none">• case는 switch문에서 레이블을 지정하기 위한 예약어로 그대로 입력해야 한다.• 레이블1: ❶번 식의 결과가 될 만한 값 중 하나를 입력한다. 결과가 '레이블1'과 일치하면 이곳으로 찾아온다. 식의 결과가 5종류로 나타나면 case문이 5번 나와야 한다.
실행할 문장1; break;	❶번 식의 결과가 ❸번의 '레이블1'과 일치할 때 실행할 문장이다. switch문을 탈출하여 ❹번으로 간다.
case 레이블2: ❹	❶번의 식의 결과가 '레이블2'와 일치하면 찾아오는 곳이다.
실행할 문장2; break;	❶번의 식의 결과가 ❹번의 '레이블2'와 일치할 때 실행할 문장이다.
:	switch문을 탈출하여 ❺번으로 간다.
default:	❶번의 식의 결과가 '레이블1' ~ '레이블n'에 해당하지 않는 경우 찾아오는 곳이다.
실행할 문장3;	
} ❺	

4. 프로그래밍 언어 활용-SEC_05(제어문)

4) switch문

- case문의 레이블에는 한 개의 상수만 지정할 수 있으며, int, char, enum형의 상수만 가능하다.
- case문의 레이블에는 변수를 지정할 수 없다.
- break문은 생략이 가능하지만 break문이 생략되면 수식과 레이블이 일치할 때 실행할 문장부터 break문 또는 switch문이 종료될 때까지 모든 문장이 실행된다.

```
예) switch(2) {  
    case 3 : printf("1");  
    case 2 : printf("2");  
    case 1 : printf("3");  
}
```

결과 23

4. 프로그래밍 언어 활용-SEC_05(제어문)

4) switch문

예제) 점수(jum)에 따라 등급 표시하기

```
#include <stdio.h>
main() {
    int jum = 85;
    switch (jum / 10) {
        case 10:
        case 9:
            printf("학점은 A입니다. \n");
            break;
        case 8:
            printf("학점은 B입니다.\n");
            break;
        case 7:
            printf("학점은 C입니다.\n");
            break;
        case 6:
            printf("학점은 D입니다.\n");
            break;
        default:
            printf("학점은 F입니다.\n");
    }
}
```

결과 학점은 B입니다.

4. 프로그래밍 언어 활용-SEC_05(제어문)

5) goto문

; goto문은 프로그램 실행 중 현재 위치에서 원하는 다른 문장으로 건너뛰어 수행을 계속하기 위해 사용하는 제어문이다.

- goto문은 원하는 문장으로 쉽게 이동할 수 있지만 많이 사용하면 프로그램의 이해와 유지 보수가 어려워져 거의 사용하지 않는다.
- 형식

goto 레이블;	레이블로 이동한다.
레이블:	goto문의 주소값이다. '레이블' 형태로 작성되며, 레이블명은 사용자가 원하는 이름을 임의로 지정할 수 있다.
실행할 문장	

C언어로 실무 프로그램을 작성할 때는 goto문을 거의 사용하지 않지만 시험을 위한 C언어 문법의 출제 범위가 명확하지 않아 포함한 내용이니 가볍게 읽어 두자.

4. 프로그래밍 언어 활용-SEC_05(제어문)

5) goto문

예제) 10보다 큰 값을 입력할 때까지 입력하기

```
#include <stdio.h>
main() {
    int a = 0;
again :
    scanf("%d", &a);
    if (a <= 10)
        goto again;
    else
        printf("%d는 10보다 큽니다.", a);
}
```


프로그래밍 언어 활용-SEC_05(제어문) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(제어문)

1. 다음 C언어 프로그램 실행 후, 'c'를 입력하였을 때 출력 결과는?

```
#include <stdio.h>
main() {
    char ch;
    scanf("%c", &ch);
    switch (ch) {
        case 'a':
            printf("one ");
        case 'b':
            printf("two ");
        case 'c':
            printf("three ");
            break;
        case 'd':
            printf("four ");
            break;
    }
}
```

two

- ③ three ④ one two three four

switch문은 한정된 값, 정해진 값을 if문에 비교하여 좀 더 가독성이 좋게 하기 위해서 만들어진 제어문이다.

모든 if문은 switch문으로 변경이 불가능하지만 모든 switch문은 if문으로 변경이 가능하다.

3. 다음 C 프로그램의 실행 결과는?

```
#include<stdio,h>
main()
{
    int a = 10;
    if (a == 10)
        printf("a는");
        printf("%d입니다.", a);
    else
        printf("a는");
        printf("%d이 아닙니다.", a);
}
```

- ① a는 10입니다.
② error 발생
③ a는 10이 아닙니다.
④ 10입니다.

if문에서 조건이 참이거나 거짓일 때 실행할 문장이 두 문장 이상이면 중괄호({ })로 반드시 감싸야 하며, 그 사이에 실행할 문장을 입력해야 한다. 위 소스는 중괄호가 없기에 else문에서 중괄호로 감싸라고 하는 error가 발생한다. 만약에 else문이 없다면 error가 발생하지 않고 첫 번째 printf()문이 if문 소속으로 실행되고 두 번째 printf()문이 main() 블록에 소속되어 실행되어지는 것이다.

프로그래밍 언어 활용-SEC_05(제어문) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(제어문)

5. 'switch(a)~'로 시작하는 명령문이 있다고 할 때, 다음 설명 중 틀린 것은?

- ① a의 값에 따라 실행할 명령문이 달라진다.
- ② a의 값으로 실수형도 가능하다.
- ③ a와 일치하는 레이블이 없으면 default 이하의 명령을 실행 한다.
- ④ switch문 안에 switch문을 또 쓸 수 있다.

a에는 int, char, enum(열거형)상수만 값으로 사용할 수 있다.

6. 다음 중, 프로그램 실행 중 현재 위치에서 원하는 다른 문장 으로 건너뛰어 수행을 계속하기 위해 사용하는 제어문은?

- ① if문 ② switch문
- ③ else문 ④ goto문

goto문은 프로그램 실행 중 현재 위치에서 원하는 다른 문장으로 건너뛰어 수행을 계속하기 위해 사용하는 제어문이다.

- goto문은 저급언어의 잔재이며 원하는 문장으로 레이블(주소)을 이용하여 쉽게 이동할 수는 있지만 남용하면 프로그램의 가독성과 유지 보수가 어려워져 거의 현재는 사용하지 않는다. 많이 쓰면 1개

4. 프로그래밍 언어 활용-SEC_06(반복문)

1) 반복문의 개념

; 반복문은 제어문의 한 종류로 일정한 횟수를 반복하는 명령문을 말한다. 보통 변수의 값을 일정하게 증가시키면서 정해진 수가 될 때까지 명령이나 명령 그룹을 반복 수행한다.

- 반복문에는 for, while, do~while 문이 있다.

4. 프로그래밍 언어 활용-SEC_06(반복문)

2) for문

; for문은 초기값, 최종값, 증가값을 지정하는 수식을 이용해 정해진 횟수를 반복하는 제어문이다.

- for문은 초기값을 정한 다음 최종값에 대한 조건이 참이면 실행할 문장을 실행한 후 초기값을 증가값만큼 증가시키면서 최종값에 대한 조건이 참인 동안 실행할 문장을 반복 수행한다.

- 형식

for(식1; 식2; 식3)

실행할 문장;

- for는 반복문을 의미하는 예약어로 그대로 입력한다.
- 식1: 초기값을 지정할 수식을 입력한다.
- 식2: 최종값을 지정할 수식을 입력한다.
- 식3: 증가값으로 사용할 수식을 입력한다.

식2가 참일 동안 실행할 문장을 입력한다. 실행할 문장이 두 문장 이상일 경우 { }를 입력하고 그 사이에 처리할 문장들을 입력한다.

- 초기값, 최종값, 증가값 중 하나 이상을 생략할 수 있고, 각각의 요소에 여러 개의 수식을 지정할 수도 있다.

예1) for(a = 1; sum <= 30;)

sum += a;

- 증가값을 생략하고 실행할 문장에서 증가값을 만들

4. 프로그래밍 언어 활용-SEC_06(반복문)

2) for문

예2) `for(a = 0; sum <= 10; a++, sum += a)`

- 증가값(`a++`, `sum += a`)을 두 개 지정함

예3) `for(a = 0, b = 5; a <= 5, b >= 0; a++, b--)`

- 초기값, 최종값 증가값을 모두 두 개씩 지정함

- 단, 싱글 루프이건 더블 루프(내부 루프 제외)이건 초기값은 무조건 1번만 수행한다.
- `for(; ;)`와 같이 조건에 참여하는 수식을 모두 생략하면 for문은 무한 반복한다. 하지만 현업에서는 무한 루프를 통해서 프로그램을 작성할 때는 보통 for문 보다 while문을 선호한다.
- for문은 처음부터 최종값에 대한 조건식을 만족하지 못하면 한 번도 수행하지 않는다.
- 문자도 for문을 수행할 수 있다.

예) `for(char a = 'A'; a <= 'Z'; a++)`

a에 'A, B, C ~ X, Y, Z' 순으로 저장함

4. 프로그래밍 언어 활용-SEC_06(반복문)

2) for문

예제) for문을 이용하여 1~5까지의 합을 더하는 다양한 방법이다.

코드	설명
<pre>int a, hap = 0; for (a = 1; a <= 5; a++) hap += a; ❶</pre>	반복 변수 a가 1에서 시작(초기값)하여 1씩 증가(증가값)하면서 5보다 작거나 같은 동안(최종값) ❶번을 반복하여 수행한다. 그러니까 'hap += a'를 5회 실행하며, 결과는 15이다.
<pre>int a, hap = 0; for (a = 0; a < 5; a++, hap += a);</pre>	for문의 마지막에 ';'이 있으므로 실행할 문장 없이 for문만 반복 수행한다.
<pre>int a = 1, hap = 0; for (; a <= 5; a++) hap += a; ❶</pre>	초기값을 지정하지 않았지만 변수 a 선언 시 1로 초기화 했기 때문에 a가 1부터 5보다 작거나 같은 동안 ❶번을 반복하여 수행한다.
<pre>int a, hap = 0; for (a = 0; a++ < 5;) hap += a; ❶</pre>	증가값을 지정하지 않았지만 최종값에서 'a++;'를 수행하기 때문에 a가 0부터 5보다 작은 동안 ❶번을 반복하여 수행한다.
<pre>int a = 1, hap = 0; for (; a <= 5;) { ❶ hap += a; a++; ❷ } ❸</pre>	초기값과 증가값을 지정하지 않았지만 변수 a 선언 시 1로 초기화 했고, ❷번을 수행하면서 a가 1씩 증가하기 때문에 a가 1부터 5보다 작거나 같은 동안 ❶~❸번 사이의 실행할 문장을 반복하여 수행한다. 실행할 문장이 2개 이상인 경우 중괄호 ({ })로 묶어준다.

4. 프로그래밍 언어 활용-SEC_06(반복문)

3) while문

; while문은 조건이 참인 동안 실행할 문장을 반복 수행하는 제어문이다.

- while문은 조건이 참인 동안 실행할 문장을 반복 수행하다가 조건이 거짓이면 while문을 끝낸 후 다음 코드를 실행한다.
- while문은 조건이 처음부터 거짓이면 한 번도 수행하지 않는다.
- 형식

while(조건)	<ul style="list-style-type: none">• while은 반복문에 사용되는 예약어로 그대로 입력한다.• (조건) : 참이나 거짓을 결과로 갖는 수식을 '조건'에 입력한다. 참(1)*을 직접 입력할 수도 있다.
실행할 문장;	조건이 참인 동안 실행할 문장을 입력한다. 문장이 두 문장 이상인 경우 ()를 입력하고 그 사이에 처리할 문장들을 입력한다.

- while(1)과 같이 조건을 1로 지정하면 while문을 무한 반복한다. 아울러 통상 현업에서는 무한 반복을 실행할 때는 while문을 주로 이용한다. 하지만 무한 반복을 하는 코드를 작성하면 반드시 무한 반복을 빠져나가는 코드가 필요하다.(중요)

4. 프로그래밍 언어 활용-SEC_06(반복문)

3) while문

예제) 다음은 1~5까지의 합을 더하는 프로그램이다. 결과를 확인하시오.

```
#include <stdio.h>
main( )
{
    int a = 0, hap = 0;
    while (a < 5) ❶
    { ❷
        a++; ❸
        hap += a; ❹
    } ❺
    printf("%d, %d\n", a, hap); ❻ 결과 5, 15
}
```

a가 5보다 작은 동안 ❶~❺번 문장을 반복하여 수행한다.
❷~❺번까지가 반복문의 범위이다. 반복문에서 실행할 문장이 하나인 경우는 { }를 생략해도 된다.
'a = a + 1;'과 동일하다. a의 값을 1씩 증가시킨다.
'hap = hap + a'와 동일하다. a의 값을 hap에 누적시킨다.
반복문의 끝이다.
a가 5가 되었을 때 5를 hap에 누적한 다음 while 문을 벗어나기 때문에 a는 5로 끝난다.

4. 프로그래밍 언어 활용-SEC_06(반복문)

4) do~while문

; do~while문은 조건이 참인 동안 정해진 문장을 반복 수행하다가 조건이 거짓이면 반복문을 벗어나는 while문과 같은 동작을 하는데, 다른 점은 do~while문은 실행할 문장을 무조건 한 번 실행한 다음 조건을 판단하여 탈출 여부를 결정한다는 것이다. 그 이유는 조건식이 뒤에 있기 때문이다.

● 형식

do	do는 do~while문에 사용되는 예약어로, do~while의 시작 부분에 그대로 입력한다.
실행할 문장;	조건이 참인 동안 실행할 문장을 입력한다. 문장이 두 문장 이상인 경우 { }를 입력하고 그 사이에 실행할 문장들을 입력한다.
while(조건);	<ul style="list-style-type: none">• while은 do~while문에 사용되는 예약어로, do~while의 끝 부분에 그대로 입력한다.• (조건) : 참이나 거짓을 결과로 갖는 수식을 '조건'에 입력한다. 참()을 직접 입력할 수도 있다.

4. 프로그래밍 언어 활용-SEC_06(반복문)

4) do~while문

예제) 다음은 1부터 10까지 홀수의 합을 더하는 프로그램이다. 결과를 확인하시오.

```
#include <stdio.h>
main() {
    int a = 1, hap = 0;
    do {
        hap += a;
        a += 2;
    } while (a < 10);
    printf("%d, %d\n", a, hap);
}
```

결과 11, 25 -> a가 9가 되었을 때 9를 hap에 누적한 다음 a에 2를 더해 a가 11이 되었을 때 do~while문을 벗어나기 때문에 a는 11로 끝난다.

4. 프로그래밍 언어 활용-SEC_06(반복문)

5) break, continue

; switch문이나 반복문의 실행을 제어하기 위해 사용되는 예약어이다.

- break : switch문이나 반복문 안에서 break가 나오면 블록을 벗어난다.

- continue

- continue 이후의 문장을 실행하지 않고 제어를 반복문의 처음으로 옮긴다.
- 반복문에서만 사용된다.

예제) 다음은 1~5까지의 합을 더하되 2의 배수는 배제하는 프로그램이다. 결과를 확인하시오.

```
#include <stdio.h>
main() {
    int a = 0, hap = 0;
    while (1) {
        a++;
        //무한 반복문을 탈출하는 코드
        if (a > 5)
            break;
        //a를 2로 나눈 나머지가 0이면 다시 반복문의 조건으로
        //이동한다.
        if (a % 2 == 0)
            continue;
        hap += a; //값을 누적한다.
    }
    printf("%d, %d\n", a, hap);
}
```

결과 6, 9

프로그래밍 언어 활용-SEC_06(반복문) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(반복문)

1. 다음 C 프로그램의 결과 값은?

```
main(void) {  
    int i;  
    int sum = 0;  
    for(i = 1; i <= 10; i = i + 2)  
        sum = sum + i;  
    printf("%d", sum);  
}
```

- ① 15 ② 19
③ 25 ④ 27

for문은 초기값, 조건식(최종값), 증감식을 지정하는 수식을 이용해서 정해진 횟수를 반복하는 반복문이다.

- for문은 초기값을 정한 다음 조건식에 대한 조건이 참이면 실행할 문장을 실행한 후 초기값을 증감식에 따라 증감시키면서 조건식이 참인 동안 실행할 문장을 반복 수행한다.

- 단, 싱글 루프이건 더블 루프이건(내부 루프는 제외) 초기값은 무조건 1번만 수행한다.

- for(;;)와 같이 조건에 참여하는 수식을 모두 생략하면 for문이 무한 반복한다. 하지만 현업에서는 무한 루프를 통해서 프로그램을 작성할

3. 다음 JAVA 프로그램이 실행되었을 때의 결과는?

```
public class array1 {  
    public static void main(String[] args) {  
        int cnt = 0;  
        do {  
            cnt++;  
        } while (cnt < 0);  
        if(cnt==1)  
            cnt++;  
        else  
            cnt = cnt + 3;  
        System.out.printf("%d", cnt);  
    }  
}
```

- ① 2 ② 3
③ 4 ④ 5

do~while문은 조건이 참인 동안 정해진 문장을 반복 수행하다가 조건이 거짓이면 반복문을 벗어나는 while문과 같은 동작을 하는데, 단 차이점은 do~while문은 실행할 문장을 무조건 한 번 실행한 다음 조건을 판단하여 탈출 여부를 결정한다는 것이다. 그 이유는 조건식이 뒤에 있기 때문이다.

4. 다음 C언어 프로그램이 실행되었을 때, 실행 결과는?

```
int main(int argc, char* argv[]) {
```

프로그래밍 언어 활용-SEC_06(반복문) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(반복문)

5. C언어에서 반복 처리를 위한 명령문이 아닌 것은?

- ① for ② while
- ③ continue ④ do~while

continue문은 continue 이후의 문장을 실행하지 않고 제어를 반복문의 처음으로 옮긴다. 반복문에서만 사용된다.

break문은 switch문이나 반복문 안에서 break가 나오면 가장 가까운 블록을 벗어난다.

continue, break문은 반복문을 제어하는 명령문이다.

6. 프로그램 언어의 문장 구조 중 성격이 다른 하나는?

- ① while(expression) statement;
- ② for(expression-1; expression-2; expression-3)
statement;
- ③ if(expression) statement-1; else statement-2;
- ④ do {statement;} while(expression);

if~else 구문은 제어문 중 분기문에 속한다.

7. C언어의 do~while문에 대한 설명 중 틀린 것은?

- ① 문의 조건이 거짓인 동안 루프처리를 반복한다.
- ② 문의 조건이 처음부터 거짓일 때도 문을 최소 한번은 실행한다.
- ③ 무조건 한 번은 실행하고 경우에 따라서는 여러 번 실행하는 처리에 사용하면 유용하다.
- ④ 문의 맨 마지막에 ";"이 필요하다.

do~while문은 조건이 참인 동안 루프 처리를 반복한다.

8. C 언어의 제어 구조(Control Structure)에서 문장을 실행한 다음, 조건을 검사하여 반복 실행의 여부를 결정하는 문은?

- ① for문 ② while문
- ③ do~while문 ④ switch~case 문



감사합니다.