

자바 프로그램 설치방법

- ① java 앱축풀기 → jdk - 8u261 - windows - x64 실행
- ② 협업에서는 8비전 사용
- ③ JRE : 자바프로그램을 실행하기 위한 프로그램, 일반 사용자가 사용.
ex) Photoshop 실행하기 위한 adobe 같은 것.
- ④ JDK : 개발자 사용하는 프로그램

(언어: Java 8비전) *검은창에 'java' - version 검색하면 설치한 프로그램 버전 확인.
개발툴(= id) : 이클립스

이클립스 설치.

- ① 폴더만들기 → 새폴더 이름: src → 하위 새폴더: Java
- ② 이클립스 앱축풀기 → workspace의 소스위치 → 폴더 'src' → 하위폴더 'Java' 선택.
- ③ window → Show view → other → **Package** 검색 → open → 왼쪽에 끌고가기
- ④ File → New → Other → Java **Project** 선택
→ Project 이름 name → Java 001 첫글자 - 대문자 → Finish → open
- ⑤ 왼쪽에 > 클릭 → JRE → Src (소스) → **Package** 선택
→ Package 이름 name → j001 첫글자 - 소문자 → Finish
- ⑥ File → New → Other → **Class** 선택
→ class 이름 → H 001 첫글자 - 대문자, 알파벳만, 특수문자x, → Public Static void 처리 → Finish

이클립스 더 알기.

- 컨트롤키 누르고 플러스/マイ너스 : 글씨 크기 조정
- ★ ◦ Class : { } 10행 까지
- ★ ◦ 메소드(함수) : { } 8행까지 ★ 들여쓰기 잘하기 ★
 - // 주석 : 설명하는 것, 명령문이 아님. ★ 메소드에 맞춰 들여쓰기 해야 함 ★
 - 컨트롤키 누르고 F11 : 출력값, Console 창이 뜸
 - Sysout 쓰고 컨트롤키 누르고 Space bar : 자동완성됨.

24 - 02 - 19 [변수][값][변수명][초기값][연산자/대입연산자][자료형]

[기술정의]

1. 변수

: 값을 저장하는 저장공간, 저장된 값은 변할 수 있다.

2. 값

: 저장한 내용

3. 변수명

: 변수의 이름

4. 초기값

: 선언문에서 처음 변수에 지정하는 값

5. 선언문

: 변수를 만든다.

6. 구현

: 선언문에서 만든 변수로 사용한다.

7. 대입연산자

: $A = B \rightarrow B$ 의 결과를 A에 대입한다. 같다는 의미가 아니다.

순서 1. B 2. A

8. 연산자

: + - * %. → / : 몫의 값

→ % : 나머지의 값

9. 자료형

: 변수가 저장할 값의 type.

① 예시 - (1)

int a = 20; → 선언문
자료형 변수명 초기값

a = 50; : 50 → 구현

a = a+30; : 80 → 구현

a = "abc"; 자료형의 int와 맞지 않음.

Sysout (a + 70) : 150 출력

Sysout (a) : a 출력

② 예시 - (2)

Q. 철수는 자신의 나이를 10으로 저장

int a = 10; → 선언문

int KKK = 10; 네이밍규칙에 어긋남. (네이밍규칙: 두글자 합칠 때 두번째글자 대문자로)

ex) eatFood

String name = "철수"; → 선언문

자료형: 문자 문자

name = "김철수" → 구현

② 예제 - (1)

Q1. String a = "홍길동";
자료형 변수 문자

: 변수 a를 선언한다. a는 문자열의 값을 저장할 수 있다.
초기값으로 홍길동을 대입한다.

Q2. a = a + "사랑해"

: 변수 a에 저장된 값과 문자열 사랑해를 연결하여
변수 a에 대입

Q3. a = a + 20;

: False → Java에 따라 다르다.

: 변수 a의 자료형은 문자열이다. 따라서 숫자 20이 연결될 수 없다

Q4. String b = 20;

: False → Java에 따라 다르다.

: 변수 b의 자료형은 문자열이다. 따라서 숫자 20이 초기값으로 선언될 수 없다.

Q5. c = b + 30;

: False

: 변수 c가 선언되지 않았다 + b는 문자열이다.

Q6. b = b + 50;

: False

: 변수 b가 선언되지 않았다. + b는 문자열이다.

Q7. String C = "70";

: C라는 변수를 선언한다. C는 문자열의 값을 저장할 수 있다.
초기값으로 70을 대입한다.

④ 예제 - (2)

Q1. int a = 30;

: a라는 변수를 선언한다. a는 숫자 int의 값을 가질 수 있고 초기값은 30이다.

Q2 int b = a;

: b라는 변수를 선언한다. b는 a가 초기값이다.

Q3 int c = a + b;

: c라는 변수를 선언한다. c는 a+b가 초기값이다.

Q4. Sysout (a);

: 30

Q5. Sysout (b);

: 30

Q6. Sysout (c);

: 60

Q7. Sysout (a == b);

: True

Q8. a = b + 30;

: a라는 변수에 b+30을 대입한다 : 60

Q9. Sysout (a + 30);

: a변수에 30을 더한 값을 출력한다. a의 값은 변하지 않는다.

Q10. Sysout (a);

: Q8에서 a의 값이 60으로 바뀌었음 : 60

Q11 int d = a - b;

: d라는 변수를 선언한다. d는 숫자 int만 가지고 초기값은 a(60) - b(30)이다 : 30

Q12 Sysout (d % 2);

: d라는 변수를 2로 나누고 나머지는 0이다 : False

Q13. Sysout (a+b+c) >= 100;

: (a(60)+b(30)+c(60)) = 150 : True

⑤ 예제 - (3)

```
Package j001;
Public class H001 {
    Public static void main (String [ ] args) {
        System.out.println ("프로그램 시작");
        String title = "나의점심";
        String Content = "오늘은 자장면";
        System.out.println ("제목 :" + title);
        System.out.println ("본문 :" + Content);
        title = "점심메뉴";
        System.out.println (title);
        int count = 0;
        Content = Content + "짬뽕도 먹음"; // 글자추가 (기존글 + 추가글)
        count = 10;
        System.out.println ("제목 :" + title);
        System.out.println ("내용 :" + Content);
        System.out.println ("조회수 :" + count);
```

```
int randomInt = 30; // 30번을 임의의 숫자로 뽑았다고 가정.
if (randomInt % 2 == 0) // 30을 2로 나눈 나머지가 0이다
    System.out.println ("짝"); // 맞으면 "짝"
else {
    System.out.println ("홀"); } // 아니면 "홀"
```

[기초정의]

1. 연산자

: 변수의 값으로 연산을 한다.

① 종류

높음

• 단항연산자 : 항이 1개

ex) $-a$, $+a$, $++a$: $a = a + 1 \rightarrow a$ 값 변함, $--a$: $a = a - 1 \rightarrow a$ 값 변함

\downarrow 전위연산자 ←

$a++$: $a = a + 1$

\downarrow 후위연산자 ←

$a--$: $a = a - 1$

int a = 10;

System.out (-a); -10

System.out (a); 10

System.out (++a); 11 $\rightarrow a = a + 1 \rightarrow$ 대입연산자 사용 $\rightarrow a$ 변수의 값 변경

System.out (a); 11

$a = -a;$

우선순위

System.out (--a); 9 $\rightarrow a = a - 1 \rightarrow$ 대입연산자 사용 $\rightarrow a$ 변수의 값 변경

System.out (a); 9

• 산술연산자 : 항이 여러개

ex) + - * / %

int a = 30;

System.out (++a + 20); 51 $\rightarrow 31 + 20$

System.out (++a + (-a)); 0 $\rightarrow 32 - 32$

int a = 10;

$a = -a; -10$

System.out (--a + a); 142 $\rightarrow -11 -11$

• 비교연산자 : 연산결과는 참 (true), 거짓 (false)

ex) ==, >, <, !=

낮음

• 대입연산자 : 뒤의 결과를 앞에 대입

ex) =, (=, -=) 참고만.

int a = 30;

int b = 0;

b = $++b + 2 * 3 + (-a)$; -23

System.out (b); -23

System.out (a + b * 2 > 50); False $\rightarrow -12 > 50$

② 예제

`int a = 10; // 선언문 → 변수선언 → 선언은 자료형, 변수명을. ⇒ 초기값`

`a = a + 10; // 구현문 → 변수사용 → 연산자 이용 or 명령문 이용` 자료형을 이해해내야 함
숫자 + 숫자 → 더하기
문자 + 문자 → 연결.

`Sysout (10); // 구현문 → 메서드(함수) → 특성한기능: 콘솔 출력`

`Sysout (a); // 구현문 → "`

`int a = 10;` : 변수 a를 선언한다. 변수 a에 10을 대입한다.

`a = a + 10;` : +연산자는 대입연산자보다 우선순위이므로 a+10을 먼저 계산, 변수에 대입.

`Sysout (10);` : Sysout 메서드에 10을 입력하고 콘솔에 10을 출력

`Sysout (a);` : Sysout 메서드에 변수 a을 입력하고 콘솔에 변수 a의 값을 출력.

함수 1. 기능
2. 입력값
3. 출력값



Java의 함수: Sysout (a) 1. 콘솔 출력
= 메서드 2. 변수 a 입력
3. a의 값 출력

DB의 함수
= function

2. 자료형

: 변수에 저장되는 값의 유형

① 대표적인 자료형

- 정수 `int` (소문자)
- 실수 `double` (소문자)
- 문자 `char` (소문자) : 1글자, ‘ ’ (작은 따옴표) → 아스키코드표, 유니코드표
- 문자열 `String` (대문자) : 1글자 이상, “ ” (큰 따옴표)

3. 연산자

: 변수는 무엇인가 하려고.

① 종류

- 단항연산자 `+ - ++ -- !` (부정의 의미)
- 산술연산자 `+ - * / %`
- ★ ◦ 비교연산자 `true or false` → 조건이 있다는 뜻 `> < == !=`
- 대입연산자 `변수의 값 변경` `= += -= *= /=`
 ex) $a += 1$ ex) $a -= 10$ → 아래보다 컴퓨터가 처리하는
 $\rightarrow a = a + 1$ $\rightarrow a = a - 10$ 속도가 더 빠르다.
 ex) $a += 10$
 $\rightarrow a = a + 10$

예제.

int a = 20;	\rightarrow a는 20
int b = 20 + 30;	\rightarrow b는 50
int c = a % 2;	\rightarrow C는 (20을 2로 나눈 나머지값이므로) 0
int d;	\rightarrow
String e = "Kim";	\rightarrow e를 선언하지 않음.
a = a + b;	\rightarrow 20 + 50을 a에 대입.
Sysout (a);	\rightarrow 70
b += 10;	\rightarrow b = b + 10 이므로 50 + 10을 b에 대입.
Sysout (b);	\rightarrow 60
++c;	\rightarrow C = 0 + 1 이므로 1을 C에 대입
Sysout (c + 20);	\rightarrow 21
Sysout (c);	\rightarrow 1
d = ++a + (-b);	\rightarrow d에 (<u>a+1</u> + (-b)) \rightarrow 71 - 60 = 11을 대입
Sysout (a >= 100);	\rightarrow 위에서 전위연산으로 인해 a의 값이 71로 바뀜 $71 \geq 100$ 은 False
Sysout (! (a >= 100));	\rightarrow True 위에서 !부정이 있으므로 $71 \geq 100$ 이 아닙니다.
Sysout (a + b + c / 2);	\rightarrow 131 순서대로 C/2는 ½이므로 0.5이지만 C의 자료형은 int여서 0
	순서 ② a + b 이므로 $71 + 60 = 131$

int k = 8 ;

```
int f = 9 ;
```

Sysout (K/2); → 4 8÷2는 4

`System.out.println(f/2);` → 4 $9 \div 2$ 는 4.5 이지만 자료형이 `int`(정수) 이므로 소수점 뒤는 x

double KK = 9; → KK 변수를 선언한다. KK는 실수의 자료형을 가진다.

SYSOUT (KK/2); → 4.5 9÷2 = 4.5 자료형이 double이어서 소수점까지 0

$KK = f_{1/2} + k_{1/2}; \rightarrow 8_{1/2} + 9_{1/2} \rightarrow 4 + 4 \rightarrow 8$ 을 뺀다 KK 에 대입한다.

Sysout KK → 8.0

* 캐스팅 *

• 145

: 변수의 자료형을 임시 바꿔. ex) $KK = \frac{(double) + 12}{\sin(1)} \frac{1}{\sin(1)} \rightarrow 9.0 / 2 \rightarrow 4.5$

Char CC = 'A'

$$\rightarrow \text{문자 A 선언} KK = \frac{(\text{double})}{\text{순서 } ②} \left(\frac{f/2}{\text{순서 } ①} \right) \rightarrow 9/2 \rightarrow 4 \rightarrow 4.0$$

$CC = CC + 1$

→ 92

Sysout (cc);

→ A

String CCC = "A"; → 변수 CCC를 선언한다. 문자열을 가진다.

$ccc = ccc + "f";$ → 큰따옴표는 문자열로 인식하기 때문에 들을 연결한다.

Skout (ccc); → A1

$S \% 2 == 0 \rightarrow$ True $8 \% 2$ 의 나머지는 0 이므로 같나는 비교연산자의 답은 T.

* 무자 *

: ASCII, 유니코드 → 'A'의 아스키코드는 65 → 65를 문자로 표현하면 'A'

sys (cc++) → 66 → 문자표 표현하면 B.

4. 단항 연산자

① 전위 연산자

: $++a$, $--a$ → 연산자 $++$ 가 피연산자 a 보다 전에 위치할 때.
→ 1 증가된 값 $\rightarrow a = \boxed{a+1}$

② 후위 연산자

: $a++$, $a--$ → 연산자 $++$ 가 피연산자 a 보다 후에 위치할 때.
→ 1 증가하기 전 값 $\rightarrow a = \boxed{a+1}$

예제

```
int a = 0;  
System.out(a); 0  
System.out(a+a); 0 → 0+0  
System.out(++a+a); 2 → 1+1  
System.out(a+a++); 0 → 0+0 → 하지만 System.out(a);의 값은 1  
System.out(++a+a-a); 1 → (1+1)-1  
System.out(a+a++-a); -1 → (0+0)-1  
System.out(a+a++-a++); 1 → (0+0)-1 → 하지만 System.out(a);의 값은 2  
System.out(a); 2
```

컴퓨터는 0과 1로만 표현한다.

숫자는 2진수로 저장한다.

(때문에 number(-)는 표현 생략 가능)

0	1	0	0	0	0	0	1
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

$64 + 1 = 65 \rightarrow$ 문자 'A'

[기출정의]

1. 배열

: 여러개의 값의 나열.

① 배열선언

0 1 2 3 4 5 ← index (불변)

int [] a = { 37, 2, 8, 17, 25, 28 } ← values

자료형 배열 변수의 배열

→ 자료형 정수를 저장하는 배열.

② 분석

길이는 6

index는 0 ~ 5

특징은 길이는 선언 후 변경이 불가능 = ★ 고정길이 → 이해하는 것이 삼화과정
= ★ 연속적 할당

③ 표현법

a[2] 출력값은 8. → 변수 a의 인덱스 번호 2인 값 → System.out (a[2]);
변수명 인덱스 배열의 2번 index의 value 값을 의미함.

④ 예시 - (1)

int [] b = new int [7] ; // 선언문
길이 : 7

인덱스 : 0 ~ 6

값 : 기본값 0

⑤ 예시 - (2)

a[2] = 10 ; 변수를 사용하여 구현한 것.

숫자 10을 인덱스번호 2인 값에 대입하여 변수의 값 바꿈.

⑥ 문제1. 철수가 뽑은 4번째 번호를 1로 수정하라.

→ a[3] = 1 ;

문제2. 철수가 짙은 번호를 모두 합하시오.

index 1, 3, 5 → 두번쨰, 네번쨰, 여섯번쨰

→ a[1] + a[3] + a[5]
 System.out

⑦ 문제3. a가 60점 이상이면 합격을 출력하시오. 60점 이하면 불합격으로 출력하시오

①

②

→ 반드시 조건기! ③

① a가 60점 이상

$\rightarrow a >= 60$

② 합격을 출력하시오.

$\rightarrow \text{SYSOUT} ("합격");$

③ 불합격을 출력

SYSOUT ("불합격");

int a = 70 ; 일때.

조건문

```
if (a >= 60) {  
    tab → SYSOUT ("합격");  
}  
else {  
    tab → SYSOUT ("불합격");  
}
```

* } 대괄호

: 대괄호 안에는 계속 추가 가능.

⑧ 문제4. 60점 ~ 80점만 합격을 출력하시오.

\rightarrow 문제 3번에서 한 줄더만 수정하면 된다.

\rightarrow if (조건) \rightarrow 60점 ~ 80점 수정

$\rightarrow A\text{조건1 and } B\text{조건2} : A \&& B$

조건문.

```
조건1      그리고      조건2  
if (a >= 60 && a <= 80) {  
    SYSOUT ("합격");  
}  
else {  
    SYSOUT ("불합격");  
}
```

&& 그리고

|| 또는

if (조건) {
(조건이 참일때 실행);

조건: True인지 False인지 판별

\rightarrow 비교연산자가 나오야 한다.

2024 - 02 - 21

1. Java Skill

- ① 변수를 선언한다.
- ② 변수를 가지고 연산자 / 명령문으로 구현한다.
- ③ If 문으로 참/거짓을 판별한다.
- ④ 배열을 사용해서 같은 자료형을 연속적으로 저장 · 관리할 수 있다.

활용

- ① → 변수는 값을 저장. → 출력할 수 있다.
- ② → 변수 \Rightarrow 연산자 / 명령문을 변수에 대입할 수 있다.
- ③ → 참/거짓 판별 \Rightarrow 조건이 있다는 것 \Rightarrow 조건이 있는 비교연산자가 있다는 것.
- ④ → 변수는 배열변수

예제 - 0220 오후과제

1. 포인트를 저장하는 변수를 만들고 테스트값으로 80을 입력한다. 포인트를 출력하세요.

$\rightarrow \text{int point} = 80;$
 $\text{System.out.println(point);}$

2. ① 숫자 2개를 저장하는 변수를 만들고, 첫번째변수는 10을 두번째 변수는 4를 입력하시오.

첫번째 변수를 두번째 변수로 나눈 몫을 구하시오. 몫은 소수점도 나와야 합니다.

$\rightarrow \text{① int a} = 10;$ $\text{System.out.println((double)a / b);}$ 캐스팅 (double)
 $\text{int b} = 4;$ $\text{System.out.println(a / b);}$

\rightarrow 실수가 정수보다 표현범위가 넓다.

실수 \div 정수 \rightarrow 결과값이 자동적으로 실수가 된다.

3. 이름을 저장하는 변수를 만들고 초기값은 null을 입력하시오. - ①

홍길동 이름을 변수에 저장하고 출력할 때는 이름 뒤에 님이라고 불리시오. - ②

$\rightarrow \text{① String name} = \text{null};$
② name = "홍길동";
 $\text{System.out.println(name + "님")}$ \rightarrow 문자+문자 : 연결.

4. ① 영어점수 90, 수학점수 87, 컴퓨터점수 100점을 저장하는 변수를 만들고 평균을 구하시오.
점수는 정수만 입력되어야 하며 평균은 소수점까지 나와야 합니다.

$\rightarrow \text{① int a} = 90; \text{ int b} = 87; \text{ int c} = 100;$
② $\text{System.out.println((double)(a+b+c / 3));}$ // 결과값을 캐스팅
 $\text{System.out.println((a+b+c) / 3.0);}$ // 자동캐스팅 ↓ 이렇게 써도 된다.
다른방법)

$\text{double avg} = (a+b+c) / 3.0;$ 새로운 변수 선언, 초기화하는 방법, 변수에 값을
 $\text{System.out.println(avg)}$ 저장해놓는게 좋다.

5. 평균을 저장하는 변수를 만들고 89.2라고 저장하시오.

평균이 80점 이상이면 합격, 80점 미만이면 불합격이라고 출력하시오.

→ double avg = 89.2;

if (avg >= 80) {

 System.out ("합격"); //

}

if (avg < 80) {

 System.out ("불합격");

}

6. level이 1이면 포인트에 30을 더하고 level이 1이 아니면 포인트에 10을 더하시오.

System을 한 번만 사용하여 최종 포인트를 출력하시오.

→ if (level == 1) {

 Point = Point + 30; // True

} else {

 Point = Point + 10; // False

}

System.out (Point);

7. 로또번호는 int [] lotto = {6, 12, 33, 4, 5, 26}; 3번째 값에 80을 더하시오.

→ int [] lotto = {6, 12, 33, 4, 5, 26}

lotto[2]

lotto[2] = lotto[2] + 80;

8. 로또번호는 위와 같다. 4번째 번호가 홀수이면 홀수, 짝수이면 짝수라고 출력하시오.

lotto[3]

조건문

if (lotto[3] % 2 == 0) {

 System.out ("짝수"); // true - 짝수일때.

} else {

 System.out ("홀수"); // False - 홀수일때.

}

10. `int [] lotto = { 6, 12, 33, 4, 5, 26 }` 로또번호이다.

짝수인 로또번호는 모두 몇개인가.

풀이과정 1. 자료형은 `int`, `[]`은 배열, `lotto`는 변수이다.

운 주인의 값이 초기값이고 자료형은 정수의 배열이다.

배열의 길이는 6이다. 인덱스번호는 0~5이다.

2. 짝수는 모두 몇개인가? 배열값에서 짝수를 찾아낸다. → 조건으로 하들이 가능.
3. 모든 배열의 값이 짝수인지 판별해야 한다. → 모든 배열의 값을 조건문으로 판별
4. 모두 몇개인가? ①선언문 ②연산자 ③조건문 중에서
구운팅은 저장하는 것 → 선언문과 가깝다.

문제풀이. `int count = 0;`

`if (lotto[0] % 2 == 0) {`

`count ++;`

`}`

`if (lotto[1] % 2 == 0) {`

`count ++`

`}`

`if (lotto[2] % 2 == 0) {`

`count ++`

`}`

`if (lotto[3] % 2 == 0) {`

`count ++`

`}`

`if (lotto[4] % 2 == 0) {`

`count ++`

`}`

`if (lotto[5] % 2 == 0) {`

`count ++`

`}`

`SYSOUT ("짝수는 모두 " + count + "개 있습니다.");`

출력결과. 짝수는 모두 4개 있습니다.

9. `int[] lotto = {6, 12, 33, 4, 5, 26};`로 정의된다.
홀수인 로또번호의 합을 구하시오.

- 풀이과정
1. 자료형은 정수의 배열. 배열: 길이, 인덱스 0~5.
 2. 홀수인 로또번호의 합. → 홀수판별은 조건을 사용 → if를 사용해야 한다.
 3. if문은 어떻게 사용해야 하나 → 모든 배열의 값에 if문 적용.
 4. 합을 구하라 → 선언문 → 변수를 선언해서 합의 값을 저장해야 해서.

문제풀이

1. `int sum = 0;`
2. `if (lotto[0] % 2 == 1) {
 sum = sum + lotto[0];
}

if (lotto[1] % 2 == 1) {
 sum = sum + lotto[1];
}

if (lotto[2] % 2 == 1) {
 sum = sum + lotto[2];
}

if (lotto[3] % 2 == 1) {
 sum = sum + lotto[3];
}

if (lotto[4] % 2 == 1) {
 sum = sum + lotto[4];
}

if (lotto[5] % 2 == 1) {
 sum = sum + lotto[5];
}

 sysout(sum);`

예제 - 0221

1. 30, 40, 50, 60, 55를 저장하고 모든 배열의 값을 출력하시오.

```
int[] a = {30, 40, 50, 60, 55}; // 저장  
System.out (a[0]);  
System.out (a[1]);  
System.out (a[2]);  
System.out (a[3]);  
System.out (a[4]); // 출력
```

2. Kim, Lee, Park 3개의 값을 배열에 저장하고 출력하시오.

```
String[] b = {"Kim", "Lee", "Park"}; // 저장  
System.out (b[0]);  
System.out (b[1]);  
System.out (b[2]); // 출력.
```

3. 1번 문제에서 배열의 값이 짝수일 경우 +10을 하여 값에 저장하시오.

$a[\text{인덱스번호}] \% 2 == 0 \rightarrow$ 모든 배열에 적용

```
if (a[0] % 2 == 0) {  
    a[0] = a[0] + 10;
```

```
}  
if (a[1] % 2 == 0) {  
    a[1] = a[1] + 10;
```

```
}  
if (a[2] % 2 == 0) {  
    a[2] = a[2] + 10;
```

```
}  
if (a[3] % 2 == 0) {  
    a[3] = a[3] + 10;
```

```
}  
if (a[4] % 2 == 0) {  
    a[4] = a[4] + 10;
```

```
}
```

4. 40이 4의 배수이면 합격이라고 출력하시오.

$40 \% 4 == 0 \rightarrow$ 4의 배수 \rightarrow % 연산자는 나머지를 구하는 연산자.

```
int a = 40;  
if (a % 4 == 0) {  
    System.out ("합격");
```

5. 15가 3의 배수이면서 5의 배수인지 확인하시오.

조건① ↓ 조건②

조건①과 조건②를 둘 다 만족해야 한다.

```
int b = 15;  
방법 1.  
if (b % 3 == 0 && b % 5 == 0) {  
    System.out ("3과 5의 배수 맞음");  
}
```

방법 2.

```
if (b % 3 == 0) {  
    if (b % 5 == 0) {  
        System.out ("3과 5의 배수가 맞음");  
    }
```

(방법 1. 중첩 if 사용

: 앞의 if가 참이라면 뒤의 if 실행

ex) if (조건) { if (조건) }

(방법 2. && 사용

: 그리고라는 뜻, 조건 1 그리고 조건 2

ex) if (조건1 && 조건2) { }

→ 조건 1과 조건 2를 다 참일 때 { } 실행.

6. 배열의 모든 값의 합을 구하시오.

합을 저장할 공간 필요 \rightarrow 변수를 선언한다.

```
int sum = 0;  
int [] aa = {30, 40, 50, 60, 55};  
sum = aa[0] + aa[1] + aa[2] + aa[3] + aa[4];  
System.out (sum);
```

7. 배열에서 50이상인 숫자는 모두 몇개 입니까?

카운팅 \rightarrow 카운팅 저장할 공간 필요 \rightarrow 변수선언.

```
int cnt = 0;  
if (aa[0] >= 50) { cnt++; } → {++ cnt;}로 바꿔도  
if (aa[1] >= 50) { cnt++; } sysout (cnt) 하면 값을 같다.  
if (aa[2] >= 50) { cnt++; }  
if (aa[3] >= 50) { cnt++; }  
if (aa[4] >= 50) { cnt++; }
```

```
System.out (cnt);
```

9. int Point = 92; 수우미양가로 출력하시오.

Point가 90점 이상이면 수 80이상은 70점이상은 60점이상 양그 이하 가

```
if (Point >= 90) {  
    Sysout ("수");  
}  
else if (Point >= 80) {  
    Sysout ("우");  
}  
else if (Point >= 70) {  
    Sysout ("미");  
}  
else if (Point >= 60) {  
    Sysout ("양");  
}  
else {  
    Sysout ("가");  
}
```

```
① if (Point >= 90) {  
    Sysout ("수");  
}  
② if (Point >= 80) {  
    Sysout ("우");  
}  
③ if (Point >= 70) {  
    Sysout ("미");  
}  
④ if (Point >= 60) {  
    Sysout ("양");  
}
```

출력: 수

이유: if문이 1개이다.

조건 ① (90이상)이 참일경우.

출력되고 끝나기 때문.

출력: 수

우

미

양

가

이유: if문이 4개이기 때문에 92점은
4번 해당되어어서 값은 4개.

★ 1개발자의 형태를 보자. 정답이 아니고 방법 중 하나이다.

예제. 가장 큰 값은?

```
int[] a = {30, 45, 3, 90, 32}
```

int maxValue = a[0]; ← Value 값은 30이 들어감.
Camel 표기법

if ($\text{maxValue} < \text{a}[i]$) 을 \leftarrow 다음 인덱스 번호의 value 값이
더 크다면 다음 인덱스 번호의 값을
 maxValue 에 대입하라

```
if (maxValue < a[2]) {  
    maxValue = a[2]  
}
```

반복하면 가장 큰 maxValue 값이 저장되게 한다.

2024 - 02 - 22.

[반복문 for]

[기능정의]

선언문 : 변수를 할당 \rightarrow 컴퓨터에 메모리에.

연산자 : 값을 계산

조건문 : 조건에 따라 참 / 거짓 실행

★ 반복문 : 지정된 수행 횟수만큼 코드반복 \rightarrow 시작값이 있어야 함.
조건이 있어야 함.

프로그래밍 공부

1. 문법공부

2. 아키텍처 이해.

3. 유통시스템 이해

증가 / 감소 값 (= 증감식) 이 있어야 함.

① 횟수 카운팅에 필요한 수식.

★ ② 동작위치: 반복코드를 실행할 때.

1. 반복문.

① 시작값 \rightarrow 이복터. \rightarrow 시작값은 한번만 실행.

② 조건 \rightarrow 한 번에 하나씩. $i--$ 여도 무한루프

③ 증감식 \rightarrow 0 1 2 3 4 \rightarrow 업다운 무한루프 \rightarrow 계속 i는 0이나가.

for (int 횟수(i=0; i<5; i++)) {
 시작값 조건 증감식

}

\rightarrow i++ 를 해서 조건에 맞지 않을 때까지 반복실행. \rightarrow i<5 가 아닐 때까지 i++ 반복

2. 반복문 실행 [for]

① 초기값 확인 $i = 0$

② 조건문 확인. [true 일 때 : } 블록의 코드 실행. \rightarrow 증감식 \rightarrow 다시 조건문으로
False 일 때 : for 문 종료.

* i 값을 초기에 어떻게 설정하느냐에 따라
코드가 간결해진다.

예제 0222 - Step. 1

출력값

1. for (int i=0; i<5; i++) {
 Sysout (i + "개 먹고");
 } → 0개 먹고
 1개 먹고
 2개 먹고
 3개 먹고
 4개 먹고

2. for (int i=0; i<5; i++) {
 Sysout (i+1 + "개 먹고");
 } → 1개 먹고
 2개 먹고
 3개 먹고
 4개 먹고
 5개 먹고
 i의 값은 위와같이 0, 1, 2, 3, 4
 괄호로 묶어 +1으로 계산되기
 때문에 1, 2, 3, 4, 5가 된다.

3. for (int i=0; i<5; i++) {
 Sysout ((i++) + " 개 먹고");
 } → 0개먹고
 2개먹고
 4개먹고

4. for (int i=5; i>=0; i--) {
 Sysout (" 안녕");
 } → 안녕
 안녕
 안녕
 안녕
 안녕
 안녕
 안녕

5. for (int i=5; i<0; i--) {
 Sysout (" 안녕");
 } → 출력되지 않음

6. 다음을 조건문으로 작성해서 해결해보시오.

질수야 얼마나 내 마음을 너에게 보여줘야하니?

질수 : 사랑해 10번만 해줘!

횟수

→ for (int i= 0; i<10; i++) {
 Sysout ("사랑해");
 }

7. int [] a = {32, 34, 24, 22, 3, 8}

모든 배열의 값을 출력하기 위해서는 다음과 같아야 한다.

Sysout (a[0]);

Sysout (a[1]);

Sysout (a[2]);

Sysout (a[3]);

Sysout (a[4]);

이것을 반복문으로 만든다면?

→ for (int i=0; i<5; i++) {
 Sysout (a[i]);
 }

예제 0222 - Step. 2

1. 다음 결과를 예측해보세요.

```
for (int i=0; i<5 i++) {
    sysout (i);
}
```

출력값
0
1
2
3
4

2. for (int i=5; i>=0; i--) {
 sysout (i)
}

5
4
3
2
1
0

3. for (int i=0; i<100 i++) {
 if (i%2 == 0) {
 sysout (i)
 }
}

0
2
4
6
8
:
98

4. for (int i=0; i%2==0; i++) {
 sysout (i);
}

0 → $i=1$ 일 때 $i \% 2 == 0$ 이 조건에
 충족되지 않으므로 for문 종료됨.

5. for (int i=0; i%2==0 || i<10; i++) {
 sysout(i);
}

0
1
2
3
4
5
:
10

$"||"$ 는 '또는' 라는 뜻으로 둘 중 하나만 T
여도 충족된다.
0, 2, 4, 6, 8, 10은 $\% 2 == 0$ True.
1, 3, 5, 7, 9는 $i < 10$ True.
나머지는 False이어서 for문 종료.

6. for (int i=0; i%2==0 && i<10; i++) {
 sysout (i);
}

0 → $\&\&$ 는 '그리고'라는 뜻으로
 둘 다 만족해야 True.
 $\% 2 == 0$ 그리고 $i < 10$ 둘다
만족하는 것은 0이다.

7. int sum = 0;

```
for (int i=0; i<5; i++) {
    sum = sum+1;
}
```

5 → sum = sum + 1 을 5번 반복한다.
① 0+1
② 1+1
③ 2+1
④ 3+1
⑤ 4+1

SYSOUT (sum);

1. 1부터 100까지 숫자 중 홀수는 더하고 짝수는 뺀 결과값 1개를 출력하시오.

int result = 0; \hookrightarrow 반복할 것. \hookrightarrow sysout (결과값)

result = result + i; \hookrightarrow 연산자, 값 저장 필요

result = result - 2; \hookrightarrow int result = 0;

result = result + 3;

result = result - 4;

\vdots

result = result - 100;

\hookrightarrow 규칙이 있다 : result = result (연산자) (숫자)

\hookrightarrow 반복 규칙.

int result = 0;

for (int i=1; i<=100; i++) { \rightarrow 1부터 100까지

if (i%2 == 0) { \rightarrow 짝수일 때

 result = result - i ; \rightarrow 반복 규칙

} else { \rightarrow 홀수일 때

 result = result + i ; \rightarrow 반복 규칙

$\}$

$\}$

sysout (result)

2. 1부터 100까지 숫자 중 홀수는 홀수끼리 더한 값을 짝수는 짝수끼리 더한 값을 각각 출력하시오.

\hookrightarrow 반복문 사용.

\hookrightarrow 반복할 것.

sysout (oddsum)

\hookrightarrow 연산자, 값 저장 2개 필요.

sysout (evensum)

① 홀수끼리 더한 값 저장할 변수 : int oddsum = 0;

② 짝수끼리 더한 값 저장할 변수 : int evensum = 0;

int oddsum = 0;

int evensum = 0;

for (int i=0; i<=100; i++) {

 if (i%2 == 0) {

 evensum += i ;

} else {

 oddsum += i ;

$\}$

$\}$

sysout (oddsum);

sysout (evensum);

3. 1부터 100까지 숫자 중 홀수는 모두 몇개인가?

↳ 반복문 사용 ↳ 연산자 ↳ 값 저장: int Cnt = 0;

↳ 몇개인가? → 출력 → sysout (Cnt);

```
int Cnt = 0;  
for ( int i=0; i<=100; i++ ) {  
    if ( i%2 != 0 ) {  
        Cnt ++      → Cnt = Cnt + 1  
    }  
}  
sysout ( Cnt );
```

4. 100부터 999까지의 숫자의 모든 숫자의 합을 구하시오. 4+5+3로 합한 값의 모든 증합
↳ 반복문 사용. ↳ 반복할 것. Int sum = 0

```
int sum = 0;  
for ( int i = 100; i <= 999; i++ ) {  
    int a = i / 100;  
    int b = i % 100 / 10;  
    int c = i % 100 % 10;  
    sum = sum + a + b + c;  
}  
sysout ( sum );
```

5. 범인은 100부터 999까지 숫자에 숨어있다. → 반복문 사용.

범인은 십의자리에 있고 3의 배수이다. 범인의 숫자를 모두 출력하시오.

ex) 437의 십의자리 ↳ 연산자.

→ 437 % 100 / 10 → 3 → (숫자) % 3 == 0

for (int i = 100; i < 1000; i++) {

int ii = (i % 100 / 10);

→ ii를 선언한 이유: 컴퓨터의 CPU를 위해.

if (ii == 0 && ii % 3 == 0) {

→ ii 대신 (i % 100 / 10) 를 넣으면

sysout (i);

계산을 2번해서 CPU 성능 ↓

}

}

6. 범인을 찾으라. 0부터 100까지 숫자를 합한다. ①

0부터 123 순서대로 합한 값을 누적한다. ②

합한 값이 44를 넘어가게 하는 숫자 1개가 범인이다. 범인의 숫자를 출력하시오. ③

① 반복문 사용.

② 합한 값의 변수 선언 \rightarrow int sum = 0; \rightarrow sum = sum + i;

③ 44를 넘어간다면 출력 \rightarrow if (sum > 44) sysout(i);

```
for ( int i = 0; i <= 100; i++ ) {
```

```
    sum += i;
```

```
    if ( sum > 44 ) {
```

```
        sysout (i);
```

```
        i = 101; → 무한루프를 막기위해 쓰임
```

```
    }
```

→ break;로 사용하면 실무.

```
}
```

방법 2)

for의 조건을 sum > 44로 바꿔도 된다.

```
int i = 0;
```

```
for ( ; sum > 44; i++ ) { → while (sum > 44);
```

```
    sum = sum + i;
```

```
}
```

```
    sysout (i);
```

```
int i = 0;
```

```
while (sum > 44);
```

```
    sum = sum + i;
```

```
    i++;
```

```
}
```

```
    sysout (i);
```

7. 다음과 같이 출력하게 하시오.

1+2+3+4+5+6+7+8+9+10 = 결과값.

```
int sum = 0;
```

```
for (int i = 0; i <= 10; i++ ) {
```

```
    sum += i;
```

```
    if (i < 10) {
```

sysout. Print (i + "+"); → Println에서 ln을 빼면 줄바꿈을 하지 않는 코드이다.

```
} else {
```

만약 Println으로 하고 출력한다면

0+

1+

2+

3+

4+

5+

6+

7+

8+

9+

10=

```
    sysout (sum);
```

$\rightarrow 0+1+2+3+4+5+6+7+8+9+10 = 55$

8. 피보나치 수열 구하기.

i 는 피보나치 수의 개수이다. 1 2 3 4 5 6 7 8 총 8개의 피보나치 수를 구하고 합한 값 출력.

결과 : $1 + 1 + 2 + 3 + 5 + 8 + 13 + 21 = \text{합한 값}$

int $f_i = 1;$

int $f_{i1} = 0;$

int $f_{i2} = 0;$

int $\text{Sum} = 0;$

for (int $i = 1; i < 8; i++$) {

 System.out.println(f_i);

 Sum += f_i ;

 int temp = f_{i1} ;

$f_{i1} = f_i$;

$f_i = temp + f_{i1}$;

}

자료형 Scanner

① 예제

Scanner in = new Scanner (System.in);

자료형 변수명 = 값

SYSOUT ("숫자를 입력하세요");

자료형 int nowNum = in.nextInt();

in.nextLine();

SYSOUT (nowNum + "을 입력하였습니다");

숫자를 입력하세요

10

10을 입력하였습니다.

SYSOUT ("이름을 입력하세요");

자료형 String name = in.nextLine();

SYSOUT (name + "을 입력하였습니다");

이름을 입력하세요

손

손을 입력하였습니다.

② 예제

Q1. 키보드로 숫자를 입력받아 홀과 짝을 구분하시오.

Scanner in = new Scanner (System.in); → 위에서 선언한 변수 중복이지만

int KK = in.nextInt();

이해를 위해 다시 쓴다.

in.nextLine();

if (KK % 2 == 0) {

SYSOUT ("짝");

} else {

SYSOUT ("홀");

}

③ 예제

Q2. 키보드로 숫자를 입력받아 그 숫자만큼 10을 더하시오.

Scanner in = new Scanner (System.in);

int maxCnt = in.nextInt();

in.nextLine();

int sum = 0;

for (int i=0; i < maxCnt; i++) {

sum += 10;

}

SYSOUT (sum);

④ 예제

Q3. 키보드로 사람의 이름을 입력받고, 점수를 입력받고, ABCDE로 출력해보자.

Scanner in = new Scanner (System.in);

int inpoint = in.nextInt();

in.nextLine();

String name = in.nextLine();

if (inpoint >= 90) {

 Sysout ("A");

} else if (inpoint >= 80) {

 Sysout ("B");

} else if (inpoint >= 70) {

 Sysout ("C");

} else if (inpoint >= 60) {

 Sysout ("D");

} else {

 Sysout ("E");

}

0226 - Step. 4

1. 100부터 999까지의 숫자 중에서 홀수만 출력하시오.

```
for ( int i=100; i<1000; i++ ) {  
    if ( i%2 != 0 ) {  
        sysout ( i )  
    }  
}
```

2. 1부터 100까지 숫자 중에서 홀수의 총합을 구하시오.

```
int sum = 0;  
for ( int i=0; i<=100; i++ ) {  
    if ( i%2 != 0 ) {  
        sum += i  
    }  
}
```

3. 다음과 같이 출력하시오. 10 9 8 7 6 5 4 3 2 1

```
for ( int i=10; i>=1; i-- ) {  
    sysout.print ( i + " " );  
}
```

↳ 없으면 출력시 줄바꿈 없음

↑ 띄어쓰기 (빈칸)을 위해.

4. 범인은 100부터 999까지 숫자 중 여러 명이다. 범인의 특징은 숫자의 각 자리수를 더한 값이 짝수이다. 범인의 숫자를 모두 찾으시오.

백의자리 + 십의자리 + 일의자리

```
for ( int i=100; i<1000; i++ ) {  
    int a = ( i/100 ) + ( i%100 /10 ) + ( i%10 )  
    if ( a%2 == 0 ) {  
        sysout ( i )  
    }  
}
```

5. 키보드로 숫자를 입력받아 짝수를 판별하시오.

② ④

① Sysout ("숫자를 입력하시오.")

② Scanner in = new Scanner (System.in); // 스캐너 변수 in 선언.
int num = in.nextInt(); // int 변수 num의 값을 ()
in.nextLine();

③ Sysout (num + "을 입력하였습니다.");

④ if (num % 2 == 0) {
 Sysout ("짝수입니다")
} else {
 Sysout ("홀수입니다")
}

6. 키보드로 숫자 2개를 입력받아 서로의 차를 구하는 프로그램을 작성하시오.

① Sysout ("숫자를 하나 더 입력하시오")

② Scanner in = new Scanner (System.in); // 스캐너 변수 in 선언
int num1 = in.nextInt();
in.nextLine();

③ Sysout ("숫자를 하나 더 입력하시오")

④ int num2 = in.nextInt();
in.nextLine();

→ 위에 Scanner in 변수를 선언을 이미 했기 때문에
밑에서 또 in을 참조.

Sysout ("입력하신 두 숫자의 차 입니다")

Sysout (num1 - num2);

7. 키보드로 숫자를 입력받아 숫자의 자리수에 4가 하나라도 있다면 잘못된 숫자로 판별하시오.

Sysout ("숫자를 하나 더 입력하시오")

Scanner in = new Scanner (System.in);

int num = in.nextInt();

in.nextLine();

Sysout (num + "을 입력하였습니다")

if (num < 1000) { → 999 까지 이내 백의자리수 까지만 생각하면 됨.

if (num / 100 == 4 || num % 100 / 10 == 4 || num % 10 == 4) } → 각각 백의자리, 십의자리, 일의자리

Sysout ("잘못된 숫자입니다")

}

}

선생님풀이 ↴

7. 키보드로 숫자를 입력받아 숫자의 자릿수에 따라 하위로 있다가, 잘못된 숫자라고 판별하시오.
→ 숫자가 몇 번째 자리의 수인지 모르니 두에서부터 차운다.

① 숫자 임의로 지정하고 test

```
for (int i = 47425; i > 0; i = i / 10) {  
    System.out(i);
```

}

출력값: 47425

4742

474

47

4

↓

② 위 출력값에서 뒤의 숫자만 뽑아내는 것 = 숫자.하나씩 (뒤에서부터) 보는 것.

$$47425 \% 10 = 5$$

$$4742 \% 10 = 2$$

$$474 \% 10 = 4$$

$$47 \% 10 = 7$$

$$4 \% 10 = 4$$

```
for (int i = 47425; i > 0; i = i / 10) {
```

```
    System.out(i % 10);
```

출력값:
5
2
4
7
4

③ 각 자리의 숫자에 '4'가 들어있으면 "잘못된 숫자"

```
for (int i = 47425; i > 0; i = i / 10) {
```

```
    if (i % 10 == 4) {
```

```
        System.out("잘못된 숫자");
```

```
        break;
```

}

}

④ 키보드 입력받기 (Scanner 추가)

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

```
int a = in.nextInt();
```

```
in.nextLine();
```

```
for (int i = a; i > 0; i = i / 10) {
```

```
    if (i % 10 == 4) {
```

```
        System.out("잘못된 숫자");
```

```
        break;
```

}

}

7. 번 심화.

각 자리에 4가 몇개인지 아는 방법은? for문이 끝나고 세어보기 → Cnt 변수 선언 → Cnt 출력

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

```
int a = in.nextInt();
```

```
in.nextLine();
```

```
for (int i = a; i > 0; i = i / 10) {
```

```
    if (i % 10 == 4) {
```

```
        Cnt++;
```

```
}
```

```
}
```

```
if (Cnt > 0) {
```

```
    System.out ("잘못된 숫자" + Cnt + "개 있다")
```

```
}
```

```
→ 2
```

8번. 키보드로 입력받아 업다운 게임하기.

```
System.out ("게임을 시작합니다")
```

```
int Com = 45; // 컴퓨터는 나누라고 가준 숫자를 미리 정의한다. → 정답입력하면 for문 종료.
```

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

```
int nowNum = in.nextInt();
```

반복할 것. → 정답 입력할 때까지 반복해야 함.

```
System.out ("1~100 사이의 숫자를 입력하세요")
```

위 코드를 이용하여 게임을 완성하세요.

```
System.out ("게임을 시작합니다")
```

```
int Com = 45;
```

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

```
System.out ("1~100 사이의 숫자를 입력하세요");
```

```
for ( ; ; ) { → 무한루프
```

[
int nowNum = in.nextInt();
in.nextLine();] → 반복해서 입력받기.

```
if (nowNum > Com) {
```

```
    System.out ("업");
```

```
else if (nowNum < Com) {
```

```
    System.out ("다운");
```

```
else if (nowNum == Com) {
```

```
    System.out ("정답");
```

```
    break; → 무한루프 멈춤
```

```
}
```

```
}
```

```

9번 String test = "abcdeabce";
for (int i=0; i<9; i++) {
    Char aaa = test.charAt(i); → test를 참조하는 매개변수 charAt()
    Sysout( aaa ); 참고
}

```

위 코드를 분석하여 a 문자열에 a가 몇개 있는지 구하는 프로그램을 작성하세요.

매개변수 ① 어떤기능 : 입력값의 위치출력.

② 입력값 (매개변수)

③ 리턴값

변수 ① 값을 저장한다

② scope (변수의 사용범위)

변수 선언된 블록 안에서만 사용 가능.

① test

```

String test = " abcdeabce"
for (int i=0; i<9; i++) {
    Char aaa = test.charAt(i); →
    Sysout( aaa );
}

```

a
b
c
d
e
a
b
c
e
index 번호 0번~8번인
문자 9개가 각각의 위치
→ 쪼개서 출력.

② 문자열에 a의 개수 세기.

```

int Cnt = 0;
String test = " abcdeabce";
for (int i=0; i<9; i++) {
    Char aaa = test. charAt(i)
    if (aaa == 'a') {
        Cnt = Cnt + 1;
    }
}
Sysout( Cnt ); → 2

```

③ 참고!

만약 e라는 글자 하나만 출력하고 싶다면?

: for문이 끝나고 난 뒤 Sysout(test.charAt(8)); → e

10번. 가장 긴 터널의 알파벳 이름과 숫자를 찾으세요.

String ttt = "aabbbccc aaaaaddbbb aaaaa";
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 → 각자의 자리의 문자
1 2 1 2 3 1 2 3 1 2 3 4 1 2 3 1 2 3 1 2 3 4 5
0의 자리값과 1의 자리값이 자리값이 서로 다르다면
같다면 ts = ts + 1 1로 초기화

```
int ms = 0; // 가장 긴 터널길이
char t = 'n'; // 가장 조신의 터널번호
Char ct = 'n'; // 가장 긴 터널이름
int ts = 0; // 터널 사이즈
```

```
for (int i = 0; i <= 21; i++) {
    Char a = ttt.charAt(i); → 각 자릿수
    if (t != a) {
        sysout ("현재터널" + t + " i인덱스" + i + " 사이즈" + ts);
        t = a; → 각 자릿수 터널번호 조신의 터널번호에 대입
        ts = 1; → 1로 초기화
    }
    else { → t = a라면
        ts++; → +1 카운팅
        if (ms < ts) {
            ms = ts;
            ct = t;
        }
    }
}
sysout ("가장 긴 터널" + ct + " 터널 사이즈" + ms);
```

예제 0227

1. int [] arr = {45, 23, 25, 64, 3, 24, 48}

```
for (int i=0; i<=6; i++) {
    sysout (arr [i])
}
```

→ 45
23
25
64
3
24
48

2. 1번 배열에서 배열이 짝수인 것만 출력하시오.

```
for (int i=0; i<=6; i++) {
    if (arr [i] % 2 == 0) {
        sysout (arr [i])
    }
}
```

→ 64
24
48

3. 1번 배열에서 배열의 값이 홀수인 곳의 인덱스만 출력하시오.

```
for (int i=0; i<=6; i++) {
    if (arr [i] % 2 != 0) {
        sysout (i)
    }
}
```

→ 0
1
2
4

4. int [] arr = {45, 23, 25, 64, 3, 24, 48}

배열의 값을 모두 더한 총합을 구하시오.

```
int sum = 0;
for (int i=0; i<=6; i++) {
    sum = sum + arr [i];
}
sysout (sum);
```

→ 232

5. int [] arr = { 45, 23, 25, 64, 3, 24, 48 }

가장 큰 배열의 값을 출력하십시오.

```
int maxValue = 0;  
for (int i=0; i<7; i++) {  
    if (maxValue < arr[i]);  
        maxValue = arr[i];  
    }  
    }  
    Sysout (maxValue);
```

6. int [] arr = { 45, 25, 23, 64, 3, 24, 48 }

짝수는 모두 몇개인가?

```
int cnt = 0;  
for (int i=0; i<7; i++) {  
    if (arr[i] % 2 == 0) {  
        cnt = cnt + 1  
    }  
    }  
    Sysout (cnt);
```

7. int [] arr = { 1, 2, 3, 0, 0, 0, 1, 2, 3, 4, 5, 2, 2, 2, 2, 0, 0, 0, 0, 0, 3, 3 }

0은 터널이다. 가장 긴 터널의 길이를 구하십시오.

```
int tunnel = 0; → 지나가는 터널의 길이 저장  
int tunSave = 0; → 가장 마지막으로 지나가는 터널의 길이를 저장하는 공간.  
int maxSize = 0; → 가장 긴 터널의 길이 저장하는 공간
```

```
for (int i=0; i<21; i++) {  
    int tn = arr[i];  
    if (tn == 0) { → arr[i]이 0일 때 (=터널일 때)  
        tunnel++; → tunnel의 길이 +1씩 카운팅  
        tunSave = tunnel; → 가장 최근에 지나간 터널의 길이 저장  
    }  
}
```

```
if (maxSize < tunSave) { → 가장 긴 터널의 길이보다 현재 저장한 터널의 길이가 더 길다면  
    maxSize = tunSave; → 현재 터널의 길이를 가장 긴 터널의 길이에 대입  
} else {  
    tunnel = 1; → 1로 초기화됨  
}  
} Sysout ("가장 긴 터널의 길이" + maxSize);
```

7. int [] arr = {1, 2, 3, 0, 0, 0, 1, 2, 3, 4, 5, 2, 2, 2, 2, 0, 0, 0, 0, 0, 3, 3}

arr[] 배열의 0번 인덱스부터 마지막까지 value를 순회한다.

arr[]의 값이 0이면 +1 카운팅, 아니면 카운팅하지 않음. → tempLength 변수 선언
maxLength를 선언해서 tempLength와 비교한다.

maxLength의 초기값은 tempLength 보다 항상 작아야 비교가 쉬우므로 초기값은 -1

① test

```
for (int i=0; i<22; i++) {  
    if (arr[i] == 0) {  
        tempLength++;  
        sysout (tempLength);  
    }  
}
```

1
2
3
4
5 → 0의 개수.
6
7
8

: 0이 누적 계산되기 때문에 12345678이 된다. 이를 해결하려면 0이 아닌 경우 (else)
tempLength를 0으로 갱신한다.

② test

```
for (int i=0; i<22; i++) {  
    if (arr[i] == 0) {  
        tempLength++;  
    } else {  
        tempLength = 0;  
    }  
}
```

1
2
3
4
5

③ 0이 3개인 것과 0이 5개인 것과 비교해서 더 큰 개수의 값을 maxLength에 대입 갱신.

```
if (maxLength < tempLength) {  
    maxLength = tempLength;  
}
```

5

이문 삽입의 위치가 else 안에 있다면 만약 배열의 값 맨 마지막에
0이 하나라도 있다면 오류가 난다.

0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18
4. int [] arr = {1000 1 1 1 0000 1 1 1 000 1 3
int size = 2;

arr 배열에서 0이 의미하는 것은 끝터.

SIZE는 건물크기. 건물을 하나만. 끝터는 연속적으로.

SIZE 설정값에 따라 건물을 지을 수 있는 위치는 모두 몇개인가?

SIZE가 2일경우 4개이다.

| 0 0 0 | | | 0 0 0 0 | | | | 0 0 0 |

끝터는 연속적으로. 건물을 하나. 건물크기는 2.

건물 개수 셀 변수선언 \rightarrow int Gunmul = 0;

앞과 뒤가 0 일때 건물의 개수 카운팅

\rightarrow arr[i] == 0 && arr[i+1] == 0

\rightarrow Gunmul = Gunmul + 1

for (int i=0; i<=18; i++)
 \quad if (arr[i] == 0 && arr[i+1] == 0) {
 $\quad \quad$ SIZE = (Gunmul = Gunmul + 1)
 $\quad \quad$ Gunmul ++ ;

}

}

SYSOUT (Gunmul);

\rightarrow 7

9. v12번. Random r = new Random(); // 랜덤숫자 출력하는 Random 선언
int num = r.nextInt(45) + 1;
// 변수 r 참조하여 nextInt 메서드 호출. 매개변수는 랜덤숫자의 범위지정.
// (45)는 0~44까지 랜덤. → +1 하여 (~45)로 보정.
// 리턴값을 변수 num에 대입.

9번. 로또번호를 자동으로 생성하는 게임을 작성하시오.

중복허용 가능, 끝번호로 6개를 배열에 저장하고, 출력해하시오.

```
int [] lotto = {1, 2, 3, 4, 5, 6}; // 임의의 숫자로 배열선언.  
for (int i=0; i<6; i++) {  
    Random r = new Random();  
    int num = r.nextInt(45) + 1; // 마찬가지로 있어야 랜덤이 반복되지  
    lotto[i] = num; // 6개 모두 다르게 나온다.  
    System.out.println(lotto[i]);  
}
```

10번. 9번에서 중복되지 않게 하시오. for문은 한 개만 사용.

[나의 정의]

~~1~~

① String 자료형

길이: .length()

② 배열[] 자료형

길이: .length.

③ for문으로 인덱스 하나하나 훑을 때.

```
for( int i=0; i<title.length(); i++ )  
= for( int i=0; i<=title.length()-1; i++ )
```

④ 배열의 각 문자열의 각 문자

ex) String [] = {"안녕", "잘가"};

index0 → index1

index0의 charAt(0) = 안 index1의 charAt(0) = 잘

index0의 charAt(1) = 녕 index1의 charAt(1) = 가

24-03-04 [이중for]

[기출정의]

1 이중for

: for 블록안에 다시 for문

① 예시 : 구구단 → i와 j의 변화, 자료형 (원시타입)

0x1	1x1	2x1	3x1
0x2	1x2	2x2	3x2
0x3	1x3	2x3	3x3
0x4	1x4	2x4	3x4

i j i j i j i j

for (int i=0; i<=3; i++) { → i는 0 1 2 3

 for (int j=1; j<=4; j++) { → j는 1 1 1 1

 i + "x" + j + "=" + (i*j); 2 2 2 2

 }

}

② 예시

i 1 2 3 4 5

1 **** *

2 ** ***

3 * * * *

for (int i=1; i<=3; i++) {

 for (int j=1; j<=5; j++) {

 System.out.print("*");

}

 System.out.println();

}

i 1 2 3

1 *

2 **

3 * * *

i=1 일때 j=1

i=2 일때 j= 1, 2

i=3 일때 j= 1, 2, 3

for (int i=1; i<=3; i++) {

 for (int j=1; j<=i; j++) {

 System.out.print("*");

j

 System.out.println();

j

j <= 3을 하면 * * *
* * *
* * *

원시 type 자료형 : 변수의 값이 실제 사용할 값	참조 type 자료형 : ○ 사용 가능 참조, 자료형이 대문자일 때
① 정수 : int ② 실수 : double ③ 문자 : char ④ 불린 : boolean	① String ex) .charAt() ② Scanner ex) .nextInt(), .nextLine() → 키보드로부터 입력받는 기능 type ③ Random ex) .nextInt() → Random 수 추출하는 기능 type

③ Random 자료형 단위테스트

→ 프로그래밍 사고를 위해 중간증거 `test`를 통해 확인하는 습관 가지자!

Random r = new Random(); → Ctrl+Shift+O → 상단에 import.java.util.Random

(1) 단위레코드

SysOut (r.nextInt(3));

변수 매커드 → 매개변수: 매개변수 값에 따라 0 ~ (매개변수 - 1) 까지의 문자를 랜덤 1개씩

* 기능별로 파일 나누기 → 나누어야 유지·보수 가능.

- ① 바이올린 for
 - ② 종복해설 for
 - ③ 출연 for

① int [] lotto = new int[6]; → 길이: 6 인덱스: 0 ~ 5

```
← for (int i=0; i<lotto.length; i++) {
```

int temp = r.nextInt(45) + 1;
lotto[i] = temp; 0~44 보정 → 1~45

② `for (int j=0; j<i; j++)` \rightarrow (=이아닌 이유: $j=0$ 일 때 j 는 있고 $j=i$ 일 때 $j=0$ 이므로)

//SYSOUT (i + " 일대 풍속체크 index" + j); → test

if (otto[i] == otto[j]) → 값이 중복일 때

‘--’ → 다시 ‘ ’의 전순서로 돌아간다. 캐릭터번호를 다시 뽑는다.

break;

130 37

$$T=1 \quad 3\pi/2$$

$$\begin{array}{r} \boxed{1} \boxed{2} \boxed{3} \boxed{1} \boxed{2} \\ \boxed{1} \boxed{2} \boxed{3} \boxed{1} \boxed{2} \end{array}$$

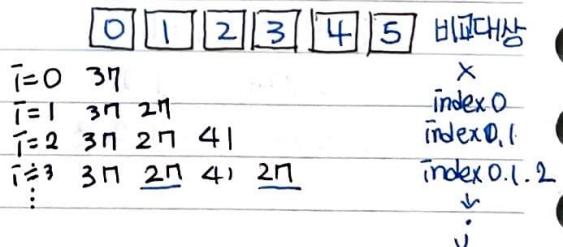
Final 50% (1st 10 weeks) $\bar{x} = 11.2$

```
int i=0; i<10000.length; i++) {  
    f[i] = 10000[i];  
}
```

```
sout (otto[i]);
```

Handwritten note: The following table summarizes the results of the study.

Digitized by srujanika@gmail.com



3. 000*

00***

0*****

분석 : ① 조건문이 필요한가? --- 조건에 따라 참 / 거짓이 나눠서 실행되는가

② 단일 반복문이 필요한가? --- i변수로 반복하여 반복되는 명령문이 있는가

③ 다중 반복문이 필요한가? --- i변수로 반복하여 반복되는 명령문을 사용할 때

반복되는 명령문에 j변수로 다시 반복을 하는가?

방법 1) 행이 4개, 열이 7개

	0	1	2	3	4	5	6
0	0	0	0	*			
1	0	0	*	*	*		
2	0	*	*	*	*	*	
3	*	*	*	*	*	*	*

기준

j는 3미만은 0, 3초과는 공백 센터값(3) + 0

j는 2미만은 0, 4초과는 공백 센터값(3) + 1

j는 1미만은 0, 5초과는 공백 센터값(3) + 2

j는 0미만은 0, 6초과는 공백 센터값(3) + 3

↓ ↓ ↓

j < 3-i 는 {0} j > 3+i 는 {공백} 나머지 : {*}}

```

for (int i=0; i<4; i++) {
    for (int j=0; j<7; j++) {
        if (j < 3-i) {
            System.out.print("0");
        } else if (j > 3+i) {
            System.out.print(" ");
        } else {
            System.out.print("*");
        }
    }
    System.out.println();
}

```

방법2) 행이 4개, 행이 0일때 3, 행이 1일때 4, 행이 2일때 5, 행이 3일때 6

	0	1	2	3	
0	0	0	0	*	4
1	0	0	*	*	*
2	0	*	*	*	*
3	*	*	*	*	*

기준

J의 끝은 3

3미만은 0

나머지칸은 *

J의 끝은 4

2미만은 0

"

J의 끝은 5

1미만은 0

"

J의 끝은 6

0미만은 0

"

↓

↓

↓

조건: $j \leq 3+i$ $j < 3-i$ 는 {0} else {*}}

```
for (int i=0; i<4; i++) {
    for (int j=0; j<=3+i; j++) {
        if (j < 3-i) {
            System.out.print ("0");
        } else {
            System.out.print ("*");
        }
    }
    System.out.println ();
}
```

[1. 1차원 배열]

1. 배열

: 연속적으로 자료저장

① 길이

: 고정, 길이는 선언 후 변경 불가

ex) `int [] a = new int [4];` 길이: 4, 인덱스: 0~3 // 선언`a = new int [7];` 길이: 7, 인덱스: 0~6 // 대입연산자로 길이 갱신

참조자료형 ↳ 길이 변경이 아닌, 새로 생성.

새로 만들 때 사용 원래 있던 값은 삭제됨.

② 인덱스

③ 자료형

ex) `String []` - 자료형: 배열

배열: 참조타입

`String`: 값의타입ex) `int []` - 자료형: 배열

배열: 참조타입

`int`: 값의타입

2. 연속적

: 메모리 (RAM)에 변수를 할당하는데, 연속적으로 저장한다. (띄엄띄엄 X)

배열은 연속적인 byte 자리가 필요하다.

1 byte = 8 bit

3. 컴퓨터 구조

① CPU

② 보조기억장치 - SSD

③ 주기억장치 - RAM (Random Access Memory)

순차적이 아니다. 랜덤이다.

언제든지 접근이 가능하다.

4. 이차원 배열

: 배열의 값이 배열

ex) `int [][] a = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}`=
$$\begin{array}{c} a[0] \quad a[1] \quad a[2] \\ \hline 0 & 1 & 2 & 3 \\ 1 & 4 & 5 & 6 \\ 2 & 7 & 8 & 9 \end{array}$$
 열`int [][] a = {{1, 2, 3}, a[0]``{4, 5, 6}, a[1]``{7, 8, 9}} a[2]`

0	1	2	3
1	4	5	6
2	7	8	9

행

① `a.length`는 3② `SYSOUT (a[2][1]);` 은 8③ `a[0].length`는 3④ `a[0][2] = 7;` → Value 3을 7로 바꿔라.⑤ `a[0]`의 의미는 `{1, 2, 3}`이고 행이다.⑥ `SYSOUT`은 `a[][]` 형태.

5. 이차원 배열 설정하기.

`int[][] b = new int[4][5];`

Q1. b 배열의 길이

`System.out (b.length) → 4` [4][5] 배열의 길이는 행의 개수

Q2. 0번째 행의 길이

`System.out (b[0].length) → 5`

Q3. 2차원배열 출력하기.

```
for (int i=0; i<b.length; i++) {  
    for (int j=0; j<b[0].length; j++) {  
        System.out.print (b[i][j]);  
    }  
}
```

`System.out();` → 줄바꿈

}

→ 00000 → 배열에 값을 대입하지 않았으므로
00000 초기값인 '0' 출력.
00000
00000

[1] [1] [1] [1] [1] [1] [1] [1] [1] [1]

1. 배열

연속적으로 자료저장

① 길이

고정, 길이는 선언 후 변경 불가

ex) int [] a = new [4]; 길이:4, 인덱스:0~3 //선택

```
a = new int [7]; 길이: 7, 인덱스: 0~6 // 대입연산자로 길이 갱신
```

참조자료형 ↳ 길이 변경이 아닌, 새로 생성.

새로 만들 때 사용 원래 있던 값은 삭제됨.

② 알데스

③ 자료형

ex) String [] - 자료형 : 배열

배열 : 참조타입

String : 값의 타입

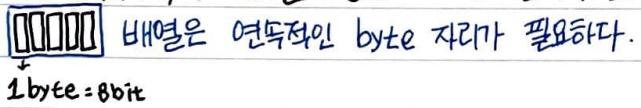
Ex) `int[]` - 자료형: 배열

배열 : 참조타입

`int` : 값의 타입.

2. 연속적

: 메모리(PAM)에 변수를 할당하는데, 연속적으로 저장한다. (찍엄찍엄X)



3. 컴퓨터 구조

① CPU

② 보조기억장치 - SSD

③ 주기억장치 - RAM (Random Access Memory)

순차적이다. 근본적이다.

언제든지 접근이 가능하다.

4. 이차원 배열

: 배열의 값이 배열

ex) `int [][] a = {{1,2,3},{4,5,6},{7,8,9}}`

- ① `a.length`는 3
 - ② `a[0].length`는 3
 - ③ `a[0]`의 의미는 `{1, 2, 3}`이고 행이다.
 - ④ `Sysout`은 `a[][]` 형태.
 - ⑤ `Sysout(a[2][1]);`은 8
 - ⑥ `a[0][2] = 7;` → Value 3을 7로 바꿔라.

5. 이차원 배열 선언하기.

`int[][] b = new int[4][5];`

Q1. b 배열의 길이

`System.out (b.length) → 4` [4][5] 배열의 길이는 행의 개수

Q2. 0번째 행의 길이

`System.out (b[0].length) → 5`

Q3. 2차원 배열 출력하기.

```
for (int i=0; i<b.length; i++) {  
    for (int j=0; j<b[0].length; j++) {  
        System.out.print (b[i][j]);  
    }  
}
```

`System.out();` → 줄바꿈

{

→ 00000 → 배열에 값을 대입하지 않았으므로
00000 초기값인 '0' 출력.
00000
00000

24-03-18 [CLASS 나누기]

[기술정의]

1. CLASS를 나누는 목적★

: 하나의 CLASS는 하나의 기능만 하는 것이 좋다.

2. CLASS 나누는 예시

(1) Project : Java 0318

(2) Package : main

(3) Class : ① Pstart → 목적 : 프로그램 시작의 기능

↳ Public static void main (String [] args)

↳ Pstart 클래스 안에 main() 만들음 ① 기능 : 프로그램의 시작

② 매개변수 : String

③ 리턴값 : 없다. void 때문.

② Room → 목적 : 방을 관리하는 기능

↳ main() 없이 생성한다. 프로그램을 실행하지 못함.

3. 프로그램 종료

: main() 가 끝나면 프로그램이 종료된다.

: main() {} → 줄괄호가 닫히면 프로그램 종료.

4. JVM

: 프로그램 자체를 실행하는 프로그램.

: main() 을 찾고 찾으면 프로그램 실행 시작.

5. 런타임

: 프로그램 실행 중 (Ctrl + F11)

: CLASS를 만든다고 런타임이 실행되는 것은 아니다. main() 가 있어야 한다.

6. 멤버드

① 가독성이 좋아진다

: 업무 효율이 높아진다.

② 재사용성이 높아진다

③ 하나의 멤버드는 하나의 기능을 하는 것이 좋다

: 세부적인 수록 좋다. 주관적이다. 디밀 책임의 원칙

④ 훼손하면 리턴해야 한다.

Public class Restart {

 Public static void main (String [] args) {

 → 프로그램 시작을 기능하는 대시트

Public class Room {

 Scanner in = new Scanner (System. in)

 String [] room = new String [5]

 Sysout ("프로그램 시작");

 new Room (); → 생성자

 Sysout ("프로그램 종료");

 } → main () 대시트가 끝나면 프로그램 종료

}

Room () { → 생성자 대시트, class의 시작,

class의 이름과 동일하다

 for (; ;) {

 Sysout ("1. 등록 2. 전체보기 3. 삭제/변경")

 int a = in.nextInt (); in.nextLine () ;

 if (a == 1) {

 insert (); → 대시트 호출

 else if (a == 2) {

 allList (); → 대시트 호출

 else if (a == 3) {

 reset (); → 대시트 호출

 else {

 break;

 } } } → Room () 종료

 Public void insert () { → 대시트 선언

 → 대시트 선언을 통해 대시트 기능을 넓리한다

 Sysout ("등록 가능");

 String a = in.nextLine ();

 for (int i = 0; i < room.length; i++) {

 Sysout (i + "번 : " + room [i]);

 } }

 Public void allList () { → 대시트 선언

 Sysout ("전체보기 가능");

 for (int i = 0; i < room.length; i++) {

 Sysout (i + "번 : " + room [i]);

 } }

 Public void rest () { → 대시트 선언

 for (int i = 0; i < room.length; i++) {

 room [i] = null; // 삭제하기

 } }

24-03-19 [Method]

[기출정의]

1. Method

: Class에서 기능을 정의하고 호출하여 사용

① 정의 : 매서드 선언 ex) Public void abc () {} → 매서드를 만들었을 뿐 사용X

② 호출 : 매서드 호출 (매개변수) \rightarrow 매서드처리 \rightarrow 리턴값 \rightarrow 매서드 사용 = (21)

2. Method 구성

① 리턴 type : 리턴값이 가질 수 있는 type (자료형)

② 매서드 이름 : 소문자, 단어 1개 이상 조합, 카멜 표기법 (앞 단어 소문자, 뒤 단어 처음 대문자)
네이밍 규칙 (앞 동사 뒤 명사)

*③ 매개변수 선언 : 매서드선언 시 받을 변수 선언

④ 리턴값 : 매서드를 호출한 위치로 리턴 값을 리턴한다.

3. Method oki

→ 이름으로 (PLUS 한다)라는 기능을 유추할 수 있게 설명해야 한다.

① public int doplus (int a, int b) { → Method() 선언
접근제어자 리턴타입 매서드명 값 받을 변수선언

return $a + b$; → 리턴값 : int

② Test() {

sysout doplus (20, 50)

→ Method() 호출

③ Console 결과값

∴ 70 → 리튬액 : 탄소

```
① Public boolean idChk ( String id ) {
```

If (id.length() >= 4) {

return true;

三

return false;

→ return을 만나면 리턴을하고
다시드 종료.

۳

해석: 아이디의 길이가 4이상이면 참. 아니면 거짓

② Test() {

sysout (idChk ("abcde")); → 길이 5

③ console 결과값

: true

① Public boolean charchk (String a, char b) {
for (int i=0; i<a.length(); i++) {
if (a.charAt(i) == b) { // a문자열 각각 Char i+ Char b와 같으면
return true;
}
}
return false;
}

② Test () {
String a ↑ char b ↑
System.out (charchk ("abcde", 'f'));

③ Console 결과값
: false

① Public int abc () {
int a = 30;
return a; // a는 리턴 값, type은 int
② abc () ; → Call, 콘솔창에는 X
System.out (abc ()); → Call, 하지만 System으로 콘솔창에 결과값이 뜬다

int b = abc ();
System.out (b); → 콘솔창에 결과값이 뜬다

③ Console 결과값

: x
30
30

[기본정의]

1. CLASS

: 기능을 수행하는 대상 → `new` → 객체를 만든다

실제로는 사용 불가능

객체만드는 명령어

`CLASS`를 객체로 만들어 참조 (.)하여 사용

- 실제 사용 가능하도록 만들음

인스턴스와 비슷한 의미

: 멤버변수와 Method로 구성됨.

① Class 예시

: `Scanner in = new Scanner (System. in) → in.nextInt();`

참조타입, class로 참조할 객체의 주소 저장

Scanner는 참조타입이어서 . 을 사용 가능.

: `int a = 10;`

classx 실제값 저장

2. 원시타입 VS 참조타입

① 원시타입

: Java에서 원시타입은 class이지만 사용편의를 위해 원시타입으로 제공한다.

: 실제값이 저장 (현금)

② 참조타입

: Java에서는 class로 정의한다.

: 참조변수의 자료형 class의 멤버변수와 대상을 참조한다.

: class의 주소 저장 (카드)

[Class 1] : Main

Public class Main {

 Public static void main (String [] args) { // 프로그램 시작 매서드 : main ()

 Word w = new Word (); // Word class 객체 생성

 w.menu (); // w 변수에 저장된 주소를 참조하여 menu 매서드 call

[Class 2] : Word

Public class Word {

 ↳ 맴버 변수 위치

 String [] word = new String [5]; // 배열 Word 객체 생성

 Scanner in = new Scanner (System.in); // 스캐너 in 객체 생성

 ↳ 매서드의 위치

 word () { // 생성자 ① 객체가 만들어질 때 처음으로 시작하는 매서드이다.

 } ② 재호출이 불가능하다.

 ③ 리턴 type 이 없다.

 ⊕ 생략 가능, 자동 생성이 가능하다.

↳ 매서드 정의하기

 Public void insert () { // 등록 매서드

 Sysout ("단어를 등록하세요");

 String inword = in.nextLine();

 for (int i=0; i<word.length; i++) {

 if (word[i] == null) { // 빈칸이라면

 word[i] = inword;

 break;

 Public void menu () { // 메뉴 매서드

 while (true) {

 Sysout ("1.등록 2.삭제 3.수정");

 Sysout ("번호선택");

 int SelNum = in.nextInt(); in.nextLine();

 if (SelNum == 1) insert(); // 매서드 호출하기

 else if (SelNum == 2) delete(); // 매서드 호출하기

 else if (SelNum == 3) update(); // 매서드 호출하기

 else {

 break;

 }

```
Public void del() { // 삭제메소드
    for (int i=0; i<word.length; i++) {
        System.out.print(i + ": " + word[i]);
    }
    System.out.println("삭제할 단어 선택");
    int delNo = in.nextInt(); in.nextLine();
    word[delNo] = null;
}
```



```
Public void update() { // 수정메소드
    for (int i=0; i<word.length; i++) {
        System.out.print(i + ": " + word[i]);
    }
    System.out.println("수정할 단어 선택");
    int delNo = in.nextInt(); in.nextInt();
}
```

```
System.out.println("수정할 단어를 입력하세요");
word[delNo] = in.nextLine();
```

□ 간단 매서드생성 → list()

```
Public void list() {
    for (int i=0; i<word.length; i++) {
        System.out.print(i + ": " + word[i]);
    }
}
```

□ 간단 매서드생성 → SelNum()

```
Public int SelNum (String title) {
    list();
    System.out.print(title + " 할 단어 선택");
    int inNo = in.nextInt();
    in.nextLine();
    return inNo;
}
```

```
Public Void del() { // 삭제마이너드  
    int delNo = SelNum ("삭제");  
    Word [delNo] = null;  
}
```

```
Public Void update () {  
    int modNo = SelNum ("수정");  
    Sysout ("수정할 단어를 입력");  
    word [modNo] = in.nextLine();  
}
```

[기술정의]

1. Method

① 정의

② 호출 (= Call)

- 매개변수에게 값 전달 → 대서드에서 처리 → 리턴값을 받음 ★

2. Class와 객체

① class

- 요구사항에 대한 정의 (설계)
 - 맹비변수와 메서드로 구성

② 개별제

- : C109S를 실제 사용 가능하게 만드는 것.
 - : 참조할 수 있다.

3. 자료형

① 원시타입 : 실제값 저장

ex) int

② 참조파입 : 객체의 주소값을 heap에 저장

ex) `int [] a = new int [];`

A 배열은 참조타입, 값에는 int 원시타입 저장.

③ 예시

int c = 30;

word {

String Kor

String Eng

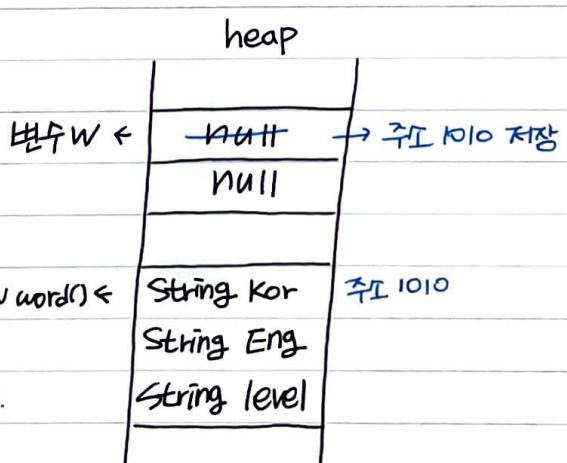
5

Week 1 - New word [?] :

→ 100% 클러스터를 저작한 배열을 만들어라.

new = new word();

생성자



① Code | ② Stack | ③ heap

개체의 생식과 소멸의 사이클을 관리한다.

[기출정의]

1. 참조형

: 참조한다 = 주소를 찾아간다.

① 예시

W 배열이 있다면 가정하면

heap

null
null
null

sysout (w[0]);

→ null

sysout (w[0].kor);

→ NullPointerException (오류)

heap

null
null
null
null

→ 1010 sysout (w[0]@a)

→ null 출력

주소: 1010

변수명: a

2. 참조하는 방법

: 참조하려면 생성한 객체의 변수를 만든다.

new word () → 객체생성

word t = new word () → 생성한 객체의 주소를 변수 t에 저장.

① 예시

School s = new School (); → 변수명 s의 자료형은 School 객체이다.

new 사용으로 School을 객체로 사용 가능하다.

heap에는 School 객체의 주소값이 저장되었다.

School [] ss = new School [4]; → 변수명 ss은 배열이다.

new 사용으로 School의 배열을 사용 가능하다.

heap에는 4개의 배열이 저장. 초기값은 null이다.

* NullPointerException 오류코드

: 참조한 객체가 존재하지 않는다

1. CLASS 설계

① 단일 책임의 원칙

: 하나의 class는 하나의 기능 → 프로그램 설계 전제를 보아한다.

② 필드 변수

: 저장 → 클래스를 불러 하려면 필드 변수, 메서드 확인.

메서드

: 기능

③ 독립성

: class는 독립적이다.

: 업그레이드 사항이 있어도 class 수정을 최소화해야 한다.

: class 내부는 독립적이지만 의존관계에서는 요청할 수 있다.

④ Method 정의

접근제어자 리턴타입 메서드명 (매개변수 저장) 을

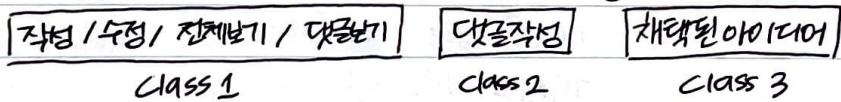
return

3

* 메서드의 return 값 반환 ① 데미드 종료 시

② return 값이 있을 시. 뒤에 코드가 더 있어도 메서드 종료.

⑤ 기능 분류하기 → class 정의하기 → 하나의 기능



⑥ 클래스의 생성자

: 클래스 시작할 때 처음 실행되는 메서드

: 재호출이 불가능하다

: 리턴타입이 없다.

: 생략이 가능하며 알아서 만들어준다.

⑦ 변수

지역변수 : 블록 안에서 생성과 소멸을 한다.

멤버변수 : 클래스의 전부. (잘 사용하지 않는 멤버변수는 메모리 낭비이다.)

↳ garbage collector가 더 이상 사용하지 않은 객체를 제거함.

1. 객체지향 언어

- : 요구사항을 추상화하여 Class (필드와 메서드)를 정의하고
객체로 만들어 실제 사용하는 것.

2. 캡슐화

- : 내용물이 빠져 않도록 Class를 캡슐화한다.

3. 접근제어자.

- : 맴버변수와 메서드의 접근권한을 정의하는 것.

① Public

- : 어디에서나 가능

② Private

- : 같은 클래스 내부에서만 가능, 외부에서 빠져나갈 때 사용.

③ Protected

④ default

- : 같은 패키지 내부에서 가능. 아무것도 안 지정하면 default이다.

예시

같은 Package 다른 Class

```
Class Member {  
    String id;  
    String name;  
    Public void prt()  
    Private void aaa()  
}
```

```
Class MemberAdm {  
    Member m = new Member();  
    m.id → 가능 (default)  
    m.name → 가능 (default)  
    m.prt() → 가능 (Public)  
    m.aaa() → 불가능 (Private)  
}
```

1. Collection Framework.

: 자료 구조 (자료를 저장하는 형태)

2. 종류

- ① Set : 순서 없다
- ② List : 순서가 있다
- ③ Map : 순서가 없다

3. ArrayList

: 가변길이 배열

: 자료가 추가되면 배열의 길이 늘어나고 자료가 삭제되면 배열의 길이가 줄어든다.

: 순서가 있다 = index번호가 있다. 중간에 삭제되면 빈칸을 입어기 위해 앞으로 당겨진다.

: Class로 정의되어 있기 때문에 메서드 사용 가능 ex) add(): 추가, size(): 배열의 길이

→ 번호는 은닉화되어 있음.

① ArrayList 선언

```
ArrayList <저장할 자료형 type> 변수명 = new ArrayList <>();
```

② ArrayList 출력

```
for (int i=0; i< mmList.size(); i++) {  
    mmList.get(i).print()  
}
```

→ 가변길이 배열이어서 if(a[i] != null) 이 없어도 된다.

③ ArrayList 삭제

```
mmList.remove (삭제할 Index 번호)
```

→ 값을 삭제하면 Index 번호가 앞으로 밀려서 나갈 수도 있다.

1. ArrayList

- : Util이라는 패키지에서 Java가 기본적으로 제공하는 Class
- : 암포트할 때 Java.util.ArrayList
- : 맵변수와 매번드가 있다

2. ArrayList의 메서드

- ① add(): 리스트의 마지막 index에 추가
 - ② remove(): 지정 index의 값 제거 → 뒤의 값을 앞으로 당겨진다.
 - ③ get(): 지정 index의 값 반환
 - ④ size(): 배열의 길이 반환
 - ⑤ clear(): 모든 요소 제거
- * API 찾으면 메서드 가능 확인 가능.
* 새로운 Class를 만나면 이론(변수 / 메서드)을 적용할 수 있다.

1. Java에서 오라클 연동

① DAO

- : Data Access Object
- : 데이터베이스에 접근

② DTO

- : Data Transfer Object
- : 객체가 데이터를 전송

③ JDBC (드라이버)를 build

- : 연동해려면 DB에서 제공하는 DB 드라이버를 연동한다.
- : 한번만 하면 된다.

④ Access

- : DB의 저장방식은 투플, Java는 class

⑤ DAO - DTO

- : DB의 투플 → ResultSet 객체에 저장 → List에 저장 → Java에 리턴.

2. DAO 코드 작업 절차

- ① 드라이버 빌드 (JDBC) - 프로젝트 당 1번만
- ② 클래스로 드라이버 로드 - class당 1번만
- ③ 데이터베이스와 연결 (Connection) - SQL 쿼리를 실행마다 매번
- ④ 쿼리작성
- ⑤ 쿼리 전송 (executeUpdate - insert, update, delete → 행의 수 반환
executeQuery - select → 데이터반환 → ResultSet에 저장)
- ⑥ 4,5번에서 사용한 자원 반납

Public Dto Selecte_id (Dto dto) { // 현재 사용법이 참조 가능한 Select-ID 메서드는 호출시 매개변수를
접근해야 하며 Dto를 통해 전달받은 값 저장한 뒤 dto에 // 전달하고 dto에 저장한다. 이런 값이 있는데 dto은 Dto

Dto newd = null;

자료형 변수명 초기값 // Dto 클래스를 값으로 가질 수 있는 newd 선언하고 참조 가능한 자료형이고 초기값은 null이다.

If (getConnection () != null) { // getConnection()의 리턴값이 null이 아니면 → 연결성공이라면
함수호출

try { // 시도한다

PreparedStatement pstmt = null; // 자료형이 커리큘럼과 맞지 않아 pstmt 변수를 선언하고 초기값은 null이다.
자료형 변수명 초기값

ResultSet rs = null; // 커리큘럼과 맞지 않아 rs를 선언하고 초기값은 null이다.
자료형 변수명 초기값

String sql = "select id, pw from userinfo where userid=? and password=?";

자료형 변수명 초기값 (userinfo 테이블에서 id, pw 컬럼의 값을 선택. ?이 어디에 ?비밀번호인 것들 둘째.)

pstmt = conn.prepareStatement (sql);

변수명 대입 connection의 메서드호출 // SQL 쿼리를 아래처럼 넣어 실행할 때 pstmt에 저장
→ 연결해서 SQL 쿼리를 DB에 대입한 주소?를 pstmt에 대입한다.

pstmt.setString (1, dto.getTextId()); // SQL 쿼리를 pstmt에 실행할 때 ?에 dto에서 가져온다.

// SQL 쿼리를 실행할 때 ?에 넣을 문자열을 dto에서 가져온다.

pstmt.setString (2, dto.getTextPw()); // SQL 쿼리를 pstmt가 실행할 때 ?에 dto에서 가져온 pw를 넣어준다.

rs = pstmt.executeQuery (); // 쿼리를 대입하고 실행한 결과값을 rs에 대입한다.

대입 커리큘럼 창업별로 PIKE

If (rs.next ()) { // 만약 실행결과값에 다음행이 있다면.

String pw-chk = rs.getString ("password");

자료형 변수명 초기값

// password 컬럼에서 문자열을 반환한 것을 pw-chk에 넣는다.

결과값에 저장된

String id-chk = rs.getString ("userid");

SQL결과값에 저장된 userid 컬럼의 문자열을 반환한 것을 id-chk 변수에 넣는다

newd = new Dto();

변수명 대입 Dto를 객체로 생성

newd.setTextId (id-chk); // 쿼리와 저장한 Id를 newd의 Id에 저장한다.

dto의 메서드 대체

newd.setTextPw (pw-chk); // 쿼리와 저장한 pw를 newd의 pw에 저장한다.

Catch (Exception e)

예외처리로 프로그램을 종료시키지 않고 실행.

return newd;

리턴값 Dto를 저장되어 있는 변수명 // 반환한 것으로 초기화를 수행한다.

1. 마이스트 소개.

; Selecte -Id ()는 리턴값이 있고 매개변수도 네온다.

리턴값은 셸쿼리를 실행 결과값인 List와 PW를 주제로 DTO가져 news를 반환한다.

연결이 성공해나갈 때 동작하는데 실패해도 예외처리된다.

DB에서 사용자 입력한 아이디와 비밀번호를 조회한다.

의 주소값을 저장한

Dto가져 news를 반환한다.

2. 마이스트 기능

DB에 연결 성공하면 JTextField에 입력한 문자열인 아이디와 비밀번호를

DB에 저장한 table에서 조회한다.

입력한 문자열들은 마이스트 흘러서 DB매핑으로 전달받는다.

1. 컨테이너

: 레이아웃이 있다.

① Border Layout

: 화면을 동·서·남·북·중앙으로 구분한다.

: JFrame

② Flow Layout

: 하나의 구역에 차례대로 오른쪽에 추가한다.

: JPanel

③ Grid Layout

: 그리드를 만든다.

2. 컴포넌트

: 컨테이너에 배치하는 객체

3. 상속 (extends)

: 부모클래스가 가지고 있는 멤버변수와 메서드를 자식에게 상속.

: 자식은 부모로부터 상속받고 나아가 확장 (extends) 할 수 있다.

: JFrame을 상속받아 GUI를 구현할 수 있다.

① this

: 내 자신의 주소

: 내 자신의 주소를 참조할 수 있다

: 내 자신의 주소를 매개변수로 넘길 수 있다.

② super

: 부모의 주소

: 부모의 주소를 참조할 수 있다.

4. 인터페이스 (implements)

: 어떠한 기능을 정의한다.

: 개발자는 인터페이스의 기능을 재정의 할 수 있다.

① 예시

마우스 인터페이스 (드래그, 휠 등등)