2024/05/29

package 이항연산자;

import java.util.Scanner;

public class Ex02산술연산자 {

public static void main(String[] args) {

// 1) 입력하는 도구 가져오기

Scanner sc = new Scanner(System.in);

// ctrl + shift + o : import 자동 진행

// 2) 사용자로 부터 숫자 한개를 입력받아(세자리 숫자)

// 3) num에 저장

System.out.print("숫자를 입력하세요 >> ");

int num = sc.nextInt();

// 4) 백의자리 숫자 구하기

System.out.println("백의 자리 숫자 : "+ num/100);

// 5) 십의자리 숫자 구하기 556 ---> 56 ---> 5

// 789 ---> 89 ---> 8

System.out.println("십의 자리 숫자 : "+ num%100/10);

// 6) 일의 자리 숫자 구하기

System.out.println("일의 자리 숫자 : "+ num % 10);

// 786 % 10 ---> 6

// 456 % 10 ---> 6

}

package 단항연산자

public class Ex01증감연산자 {

public static void main(String[] args) {

// ++, -- (앞/뒤)

// num ++ ---> num = num + 1 (num += 1)

int num = 3;

System.out.println(num++); // 3

// 증감연산자는 뒤에 붙어 있으면

// 먼저 실행코드(출력문)가 동작하고 나서 연산을 진행한다

// 콘솔에서는 볼 수 없지만 num = 4 인상태이다.

System.out.println(--num); // 3

// 증감 연산자는 앞에 붙어 있으면

// 먼저 연산이 진행되고나서 실행코드가 동작한다

}

}

package 이항연산자;

public class Ex03비교연산자 {

public static void main(String[] args) {

// < <= > >= ==(같다) !=(다르다)

// 비교연산자들은 결과값으로 boolean 자료형을 돌려준다

// - true/false

int num1 = 3;

int num2 = 7;

System.out.println(num1 == num2);

System.out.println(num1 != num2);

}

}

package 삼항연산자;

import java.util.Scanner;

public class Ex02삼항연산자 {

public static void main(String[] args) {

// 1) 입력하는 도구가 필요하다

Scanner sc = new Scanner(System.in);

// 2) 출력문

System.out.print("첫 번째 정수 입력 : ");

// 3) 입력문 - num1에다가 저장

int num1 = sc.nextInt();

// 4) 출력문

System.out.print("두 번째 정수 입력 : ");

// 5) 입력문 - num2에다가 저장

int num2 = sc.nextInt();

// 6) 두 수의 차가 결과값으로 나와야한다

// 큰숫자 - 작은숫자

// 어떤 수가 더 큰 숫자인지 비교!

// num1 - num2 num1이 더크면

// num2 - num1 num2가 더크면

int result = num1>num2?num1-num2:num2-num1;

System.out.println("두 수의 차 : "+result);

}

}

프로그래밍 언어

: 컴퓨터와 소통하기 위한 언어

컴파일러(번역기)

: 프로그래밍 언어를 컴퓨터가 알아먹을 수있는 기계어로 번역해주는 기능을 하고 있다~

변수

: 데이터를 담을 수 있는 공간(변할 수 있는 수)

자료형 변수명 = 데이터;

int num = 5; --> 선언 및 초기화 구문

선언 : int num;

데이터가 들어갈 수 있는 공간만 만들어 놓는 것

초기화 : num = 5;

변수에 처음으로 값을 할당한다.

재할당 : num = 6;

변수에 값을 다시 대입해주는 것

상수 : 불변하는, 변하지 않는

프로그래밍 언어에서는 데이터를 담을 수 있는 공간

단, 초기화 후 재할당은 불가능하다

final 자료형 변수명 = 데이터값;

기본데이터 타입 8가지

1. 논리형 1) boolean : true/false (1byte)

2. 문자형 1) char : (2byte) 문자 한 개, 유니코드

저장 할 때 ''(작은따옴표)를 사용해서 저장

3. 정수형

1) byte (1byte) -128 ~ 127

2) short (2byte)

3) int (4byte)

4) long (8byte)

4. 실수형

1) float (4byte)

2) double (8byte)

float pi = 3.141592 ---> double

\*\* 형변환 (Casting/Cast)

1) 강제형변환(명시적형변환) 자료형을 작은 자료형으로 변환 할 때 사용한다

2) 자동형변환(묵시적형변환) 은 자료형을 큰 자료형으로 변환 할 때 사용한다

연산자

피연산자(항)

식 : 피연산자와 연산자가 합쳐져 있는 것 (조건식)

기준 : 항의 개수를 기준으로

- 단항 연산자

1) 증감연산자 (++ , --)

앞/뒤 - num++, ++num

앞 : 연산이 먼저 이루어 진다음 코드가 실행된다

뒤 : 실행코드가 동작한 다음 연산이 이루어진다

- 이항 연산자

1) 산술 연산자 : + - \* / %

+ : 문자열 연산도 가능하다!

7 + "7" = "77"

앞에있는 숫자가 자동으로 문자열 형변환이 된다

2) 비교 연산자 : > >= < <= == !=

결과값으로 boolean결과값을 돌려준다

3) 논리 연산자 : !(NOT), &&(AND), ||(OR)

피연산자(항)이 boolean type만 올 수 있다

: true/false, 비교연산식

- 부정연산자(NOT)

!true -----> false

- AND

양쪽다 true를 가지고 있느면 결과값으로 true를 돌려준다 (하나라도 false면 결과값이 false이다.)

- OR

하나라도 true를 가지고 있으면 결과값으로 true

를 돌려준다!(양쪽다 false면 결과값이 false이다.)

- 삼항 연산자 (조건 연산자)

? :

(조건식) ? 실행문1 : 실행문2

실행문1은 조건식이 true일 경우 실행되는 코드

실행문2는 조건식이 false일 경우 실행되는 코드

조건식 : 결과값으로 t/f를 돌려주는 연산자와 피연산자가

합쳐진 구조

변수

연산자

제어문

1) 조건문 - if, ifelse, elseif, switch - case

2) 반복문 - for, while, do- while

이중 제어문

배열(자료구조) - 게임

이차원배열

메서드

OOP

상속추상인터페이스

JDBC

2024/05/30

1. 단항연산자(++,--)

앞: 연산 후 실행/ 뒤: 실행 후 연산

1. 이항연산자(산술/비교/논리/비트/시프트)

+:문자열 연산 가능

비교: == !=

논리: ! && ||

1. 삼항연산자 ㅁ?ㅁ :ㅁ

제어문

1. 조건문: if/ ifelse/ elseif/ switch-case
2. 반복문: for/ while/ do-while

main 밖

public : 접근제한자

static : 저장소

데이터타입, void : 리턴타입

--> 메서드를 통해 얻는 결과값(리턴값)의 데이터 타입

메서드명(호출명)

(매개변수)--> 모든 메서드들은 소괄호를 가지고 있다

매개변수 : 메서드 안에서 사용되는 변수로써 선언단만 작성

{} : 메서드 로직

로직 안쪽으로 return

1) 값을 호출한 곳으로 반환

2) 메서드 종료

리턴타입에 데이터 타입이 명시되어있으면 return 필수

void return 생략가능

사용(실행)

호출명(전달인자)

전달인자 == 매개변수

개수, 데이터타입, 순서

- 오버로딩(중복정의)

: 같은 클래스 내에서 같은 이름의 메서드를 만들 수 있다

단, 매개변수의 형태가 달라야한다

(데이터 타입, 매개변수의 개수, 순서)

패러다임: 개발자가 가지는 관점

객체지향 특징: 코드 재사용, 신뢰성, 유지보수, 직관적, 생산성

추상화(abstract):공통된 속성과 행위 추출,선언부분만 설계, 상세 정보 무시 상속(inheritance):메서드, 특징 들을 물려주는것

, 다형성(polymorphism): 다양한 형태로 나타남, 같은 기능에도 객체에 따라 다르게 동작 ,캡슐화(정보은닉 포함, encapsulation):외부 조작 제한, 인터페이스 통해 접근

Class:설계도

구조: filed(=데이터, 변수,속성,정보)클래스 내 정의된 변수-> 특성 나타냄

/ method: 객체의 행위 표현, 객체의 기능, 공통적인 기능

Object:실제로 존재하는것, 설계도를 기반으로 만듬

클래스를 오브젝트로 만드는 것은: 인스턴스

package 저금통;

public class Main {

public static void main(String[] args) {

// 1) PiggyBank 설계도를 사용해서 객체생성

// pig

PiggyBank pig = new PiggyBank();

// 2) pig에 money를 10000원으로 초기화 시키자

// pig.money = 100000000;

// 3) pig에 deposit이라는 기능을 사용해서

// 5000원을 입금

// 매개변수로 정수형 데이터 1개를 사용

// 전달인자도 1개가 있다...

pig.deposit(5000);

// 4) 잔액을 확인했을 때

pig.showMoney();// 15000원

// 5) 돈을 인출 - 1200원

pig.withdraw(1200);

// 6) 잔액 확인

pig.showMoney(); // 13800원

// 7) getter를 사용해서 잔액을 확인해보자!

System.out.println(pig.getMoney());

// 8) setter를 사용해서 잔액을 18000원으로 바꿔보자!

pig.setMoney(18000);

pig.showMoney();

}

}

---판매왕 풀이---

Scanner sc = new Scanner(System.in);

System.out.print("5월 판매 보험개수 : ");

int sale = sc.nextInt(); // 판매 보험개수 저장변수

int money = 0; // 성과금

if(sale > 20){

// 20개보다 상품을 더 팔았을때

int diff = sale - 20; // 더 판 상품개수 구하기

money = diff \* 50000 + 500000; // 더 판 상품개수 \* 5만원 + 기본성과금 50만원

}else if(sale == 20){

money = 500000; // 기본 20개 상품만 팔았을때 기본 성과금

}else{

// 20개보다 상품을 더 못 팔았을때

int diff = 20 - sale; // 더 못 판 상품개수 구하기

money = 500000 - diff \* 50000; // 기본 성과금 - 더 못판 상품개수 \* 5만원 차감

}

money = money < 0 ? 0 : money; // 금액이 마이너스인 경우 0으로 초기화

System.out.println(돈);

---주식 풀이---

Scanner sc = new Scanner(System.in);

System.out.print("[1]IT [2]예술 [3]금융 >> ");

int event = sc.nextInt();

System.out.print("금액 >> ");

int inputMoney = sc.nextInt();

System.out.print("개수 >> ");

int count = sc.nextInt();

int money = 0;

if(event == 1){

int investment = inputMoney \* count; // 투자금

int proceeds = (int) (investment \* 0.35); // 수익금

money = investment + proceeds; // 총금액

}else if(event == 2){

int investment = inputMoney \* count; // 투자금

int proceeds = (int) (investment \* 0.40); // 수익금

money = investment + proceeds; // 총금액

}else if(event == 3){

int investment = inputMoney \* count; // 투자금

int proceeds = (int) (investment \* 0.15); // 수익금

money = investment - proceeds; // 총금액

}

System.out.println(돈);