|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **2019\_1\_java1\_13** | **학번 :** | **20135151** | **이름 :** | **이갑성** |

* **내용 점검**

|  |
| --- |
| 1. 정적 메소드와 정적 필드  * 정적 메소드는 클래스 이름으로 호출 * 메소드나 필드 선언 시 static 키워드 사용 * 정적 메소드에서는 정적 필드와 메소드만 사용하거나 호출할 수 있다  1. 메소드 오버로딩  * 매개변수 개수와 타입, 메소드 반환 타입을 다르게 정의 * 단, 메소드 이름은 동일 |
| **class** Compute{  //메소드 오버로딩 : 매개변수 개수와 타입, 메소드 반환 타입을 다르게 정의  //단, 메소드 이름은 동일하다  **static** **int** add(**int** a, **int** b) {  **return** a+b;  }  **static** **double** add(**double** a, **double** b) {  **return** a+b;  }    **static** **int** add(**int**... a) { //가변 길이 매개 변수  **int** sum=0;  **for**(**int** t : a)  sum+=t;  **return** sum;  }  **static** **double** add(**double**[] dim) {  **double** sum=0;  **for**(**double** t : dim)  sum+=t;  **return** sum;  }  }  **public** **class** MethodOverloading {  **public** **static** **void** main(String[] args) {  **double** sum = Compute.*add*(34.2, 45.32);  System.***out***.println("sum(double) = " + sum);  System.***out***.println("sum(int) = " + Compute.*add*(34, 50));  System.***out***.println("sum(int...) = " + Compute.*add*(2, 4, 7, 12, 4, 6));  //무명배열을 매개변수로 전달  System.***out***.println("sum(double[]) = " + Compute.*add*(**new** **double**[] {3.4, 6.5, 2.4, 7.4, 12.8, 9.12}));  }  } |
| **[실행 결과]** |

|  |
| --- |
| 1. 객체 배열 |
| **import** java.util.\*;  **class** Member{  **private** String id; //필드 선언- 정보은닉  **private** **int** pass;    **public** Member() { //디폴트 생성자  **this**(" ", 0); //오버로딩 된 생성자 호출  }    **public** Member(String id, **int** pass) { //생성자 오버로딩  **this**.id=id;  **this**.pass = pass;  }    **public** String toString() { //객체 내용 출력  **return** "id = " + id +"\tpassword = " + pass;  }  //getter 메소드 - 필드값 반환  **public** String getId() {  **return** id;  }  **public** **int** getPass() {  **return** pass;  }    //setter 메소드 - 필드값 변경  **public** **void** setId(String id) {  **this**.id = id;  }  **public** **void** setPass(**int** pass) {  **this**.pass = pass;  }  }  **public** **class** MemberTest {  **public** **static** **void** main(String[] args) {  Scanner key=**new** Scanner(System.***in***);  **int** i;  Member[] obj = **new** Member[5]; //Member 객체 배열 선언 & 생성    **for**(i=0; i<obj.length;i++) {  System.***out***.print("id를 입력하세요 : ");  //생성된 객체를 배열에 저장  obj[i] = **new** Member(key.next(), (**int**)(Math.*random*()\*9000)+1000); }    //객체 배열 원소 출력  **for**(Member m : obj)  System.***out***.println(m.toString());    System.***out***.print(("변경하기 전 id를 입력하세요 >>>>> "));  String id=key.next();  **for**( i=0; i<obj.length;i++) {  **if**(id.equals(obj[i].getId())) { //접근자 메소드를 이용하여 동일한 id를 검색  System.***out***.print("입력한 id로 변경합니다. id를 입력하세요 : ");  //입력한 id를 설정자 메소드로 전달하여 id 필드값 변경  obj[i].setId(key.next()); System.***out***.println("변경된 결과를 출력합니다");  System.***out***.println(obj[i].toString());  **break**;  }  }  **if**(i >= obj.length)  System.***out***.println("해당 id가 존재하지 않습니다");  }  } |
| **[실행 결과]** |

* **프로그램 과제**

1. 제시된 프로그램은 매개변수로 전달된 값에서 큰 값을 반환하는 프로그램이다. 완성하고 결과를 제시하시오

**class** MaxOver{

//int 데이터 2개를 매개변수로 받아 큰 수를 반환하는 정적 max() 메소드 - 본인 작성

//double 데이터 2개를 매개변수로 받아 큰 수를 반환하는 정적 max() 메소드 - 본인 작성

}

**public** **class** Answer1 {

**public** **static** **void** main(String[] args) {

//max() 메소드 호출, 매개변수로 int 데이터 전달, 반환 결과 출력 - 본인작성

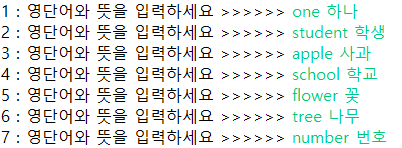
//max() 메소드 호출, 매개변수로 double 데이터 전달, 반환 결과 출력 - 본인작성

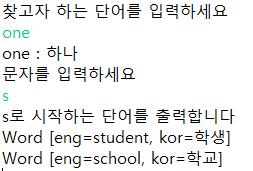
}

}

|  |
| --- |
| **[프로그램 소스]**    **class** MaxOver {  // int 데이터 2개를 매개변수로 받아 큰 수를 반환하는 정적 max() 메소드 - 본인 작성  **public** **static** **int** max(**int** a,**int** b)  {  **int** result = a > b ? a : b;    **return** result;    }  // double 데이터 2개를 매개변수로 받아 큰 수를 반환하는 정적 max() 메소드 - 본인 작성  **public** **static** **double** max(**double** a, **double** b)  {  **double** result = a > b ? a : b;    **return** result;  }  }  **public** **class** Answer1 {  **public** **static** **void** main(String[] args) {  // max() 메소드 호출, 매개변수로 int 데이터 전달, 반환 결과 출력 - 본인작성  System.***out***.println("정수형 MaxOver 결과: " + MaxOver.*max*(10, 5));  // max() 메소드 호출, 매개변수로 double 데이터 전달, 반환 결과 출력 - 본인작성  System.***out***.println("실수형 MaxOver 결과: " + MaxOver.*max*(10.5, 12.8));  }  } |
| **[실행결과]** |

1. 키보드로 영 단어를 입력하면 단어 뜻을 한글로 출력하는 프로그램을 작성하시오. 영 단어와 한글은 객체 배열 생성시 초기화 하도록 한다.





class Word{

//영어 단어 저장을 위한 필드, private

//영 단어에 대한 뜻을 저장하는 필드, private

//생성자 – 매개변수로 받은 값을 필드로 초기화

//모든 필드에 대한 getter 메소드

//객체 내용을 문자열로 반환하는 toString() 메소드

}

class WordTest{

public static void main(String[] args){

//객체 배열을 선언 & 생성, 배열 크기는 7- 본인작성

//입력한 값으로 배열 초기화 - 본인 작성

//접근자 메소드를 사용하여 객체 배열에 저장된 영 단어와 입력된 영 단어를 equals() 메소드를 사용하여 검색, 일치하는 영단어를 찾으면 그에 해당하는 뜻을 출력 - 본인작성

//입력된 문자로 시작하는 영 단어와 뜻을 한글로 출력, charAt() 사용- 본인작성

**}**

**}**

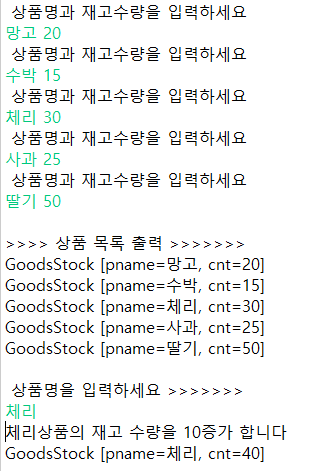
|  |
| --- |
| **[프로그램 소스]**    **class** Word {  // 영어 단어 저장을 위한 필드, private  **private** String eng;  // 영 단어에 대한 뜻을 저장하는 필드, private  **private** String kor;  // 생성자 – 매개변수로 받은 값을 필드로 초기화  **public** Word(String eng, String kor)  {  **this**.eng = eng;  **this**.kor = kor;  }  // 모든 필드에 대한 getter 메소드  **public** String getEng()  {  **return** eng;  }  **public** String getKor()  {  **return** kor;  }    // 객체 내용을 문자열로 반환하는 toString() 메소드  **public** String toString()  {  **return** "Word[" + eng + " / " + kor + "]";  }  }  **import** java.util.\*;  **class** WordTest {  **public** **static** **void** main(String[] args) {  Scanner sc = **new** Scanner(System.***in***);    **int** i;  String eng, kor;  **char** ch;  // 객체 배열을 선언 & 생성, 배열 크기는 7- 본인작성  Word[] w = **new** Word[7];  // 입력한 값으로 배열 초기화 - 본인 작성  **for**(i = 0; i < w.length; i++)  {  System.***out***.print((i + 1) + " : 영어단어와 뜻을 입력하세요 >>>>>>>> ");  w[i] = **new** Word(sc.next(), sc.next());  }    //접근자 메소드를 사용하여 객체 배열에 저장된 영 단어와 입력된 영 단어를 equals() 메소드를 사용하여 검색, 일치하는 영단어를 찾으면 그에 해당하는 뜻을 출력 - 본인작성  System.***out***.print("찾고자 하는 단어를 입력하세요 >> ");  eng = sc.next();  **for**(i = 0; i < w.length; i++)  {  **if**(w[i].getEng().equals(eng))  {  System.***out***.println(w[i].getEng() + " : " + w[i].getKor());  **break**;  }  }  **if**(i >= w.length)  {  System.***out***.println("해당하는 단어가 단어장에 없습니다.");  }  // 입력된 문자로 시작하는 영 단어와 뜻을 한글로 출력, charAt() 사용- 본인작성  System.***out***.print("문자를 입력하세요: ");  ch = sc.next().charAt(0);  System.***out***.println(ch + " 로 시작하는 단어를 출력합니다.");  **for**(i = 0; i < w.length; i++)  {  **if**(w[i].getEng().charAt(0) == ch)  {  System.***out***.println(w[i]);  }  }  }  } |
| **[실행 결과]** |

1. 상품재고를 관리하는 GoodsStock클래스를 작성하시오.

* 필드구성: 상품명(String, private), 재고 수량(int, private)
* 메소드:
* 매개변수값 만큼 재고를 더하는 메소드, 반환값 없음
* 매개변수값 만큼 재고를 빼는 메소드(재고가 빼는 수량보다 작을 경우 0을 반환하고 그렇지 않은 경우 뺀 수를 반환한다),
* 객체 정보(상품명, 재고수량)를 출력하는 toString() 메소드
* 재고수량과 상품명을 매개변수값으로 초기화 하는 생성자.
* 재고 수량과 상품명을 반환하는 접근자

1. GoodsStock 클래스 객체를 생성하고 테스트 하는 클래스를 작성하시오

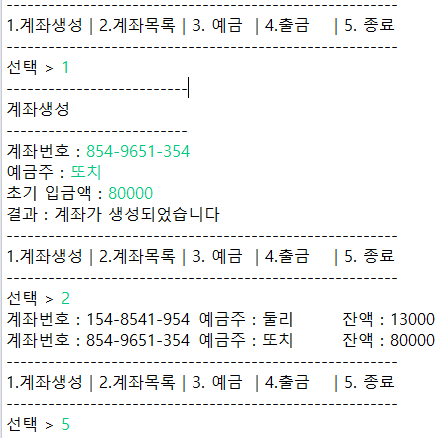
* 크기가 5인 객체 배열을 생성하여 처리한다.
* 상품명과 재고수량을 입력 받아 객체 생성시 생성자 매개변수로 전달
* 키보드로 입력 받은 상품명과 일치하는 객체의 재고수량을 더하거나 뺀 후 출력



|  |
| --- |
| **[3번~4번 프로그램 소스]**  **public** **class** GoodsStock {  **private** String name; //상품명  **private** **int** restCnt; //재고수량    //재고수량과 상품명을 매개변수값으로 초기화 하는 생성자.  **public** GoodsStock(String n, **int** r)  {  name = n;  restCnt = r;  }    //재고 수량과 상품명을 반환하는 접근자  **public** String getName() {  **return** name;  }    **public** **int** getRestCnt() {  **return** restCnt;  }    //매개변수값 만큼 재고를 더하는 메소드, 반환값 없음  **public** **void** addStock(**int** n)  {  restCnt += n;  }    //매개변수값 만큼 재고를 빼는 메소드(재고가 빼는 수량보다 작을 경우 0을 반환하고 그렇지 않은 경우 뺀 수를 반환한다)  **public** **int** subStock(**int** n)  {  **int** temp = restCnt;    restCnt -= n;  **if**(restCnt < 0)  {  restCnt = temp;  **return** 0;  }    **return** restCnt;  }    //객체 정보(상품명, 재고수량)를 출력하는 toString() 메소드  **public** String toString()  {  **return** "[GoodsStock [ pname = " + name + ", cnt = " + restCnt + " ]";    }    }  **import** java.util.\*;  **public** **class** GoodsStockTest {  **public** **static** **void** main(String[] args) {  Scanner sc = **new** Scanner(System.***in***);    GoodsStock[] gs = **new** GoodsStock[5];  **int** i, addStockCnt;  String name;    **for**(i = 0; i < gs.length; i++)  {  System.***out***.println("상품명과 재고수량을 입력하세요 >> ");  gs[i] = **new** GoodsStock(sc.next(), sc.nextInt());  }    System.***out***.println("\*\*\*\*\* 상품 목록 출력 \*\*\*\*\*");  **for**(i = 0; i < gs.length; i++)  {  System.***out***.println(gs[i]);  }    System.***out***.print("상품명을 입력하세요 >> ");  name = sc.next();  **for**(i = 0; i < gs.length; i++)  {  **if**(gs[i].getName().equals(name)) {  System.***out***.println(gs[i].getName() + "상품의 재고 수량이 10 증가합니다.");  gs[i].addStock(10);  System.***out***.println(gs[i]);  **break**;  }    }  **if**(i >= gs.length)  {  System.***out***.println(name + "을 상품 목록에서 찾을 수 없습니다.");  }  }  } |
| **[실행 결과]** |

1. 키보드로 계좌정보를 입력 받아 계좌를 관리하는 프로그램을 작성하시오. – 교재6장 확인 학습 20번





|  |
| --- |
| **[프로그램 소스]**  **public** **class** Bank {  **private** String account; //계좌번호  **private** String name; //예금주  **private** **int** totalMoney; //총액    **public** Bank(String a, String n, **int** t)  {  account = a;  name = n;  totalMoney = t;    }    **public** String getAccount()  {  **return** account;  }    **public** **void** deposit(**int** d)  {  totalMoney += d;  }  **public** **void** withdraw(**int** w)  {  totalMoney -= w;  }    **public** String toString()  {  **return** "계좌번호 : " + account + " 예금주 : " + name + " 잔액 : " + totalMoney;  }  }  **import** java.util.\*;  **public** **class** BankTest {  **public** **static** **void** main(String[] args) {  Scanner sc = **new** Scanner(System.***in***);    String account; //계좌번호  String name; //예금주  **int** deposit; //입금 금액  **int** withdraw;  **int** menu, i;  **boolean** flag = **true**;  Bank[] b = **new** Bank[100];  **int** index = 0;    **while**(flag)  {  System.***out***.println("----------------------------------------------");  System.***out***.println("1.계좌생성 |2.계좌목록|3.예금|4.출금|5.종료");  System.***out***.println("----------------------------------------------");  System.***out***.print("선택 >> ");  menu = sc.nextInt();  **switch**(menu)  {  **case** 1:  System.***out***.println("---------------------------");  System.***out***.println("계좌생성");  System.***out***.println("---------------------------");  System.***out***.print("계좌번호: ");  account = sc.next();  System.***out***.print("예금주: ");  name = sc.next();  System.***out***.print("초기 입금액: ");  deposit = sc.nextInt();  b[index] = **new** Bank(account, name, deposit);  index++;  System.***out***.println("결과: 계좌가 생성되었습니다.");  **break**;  **case** 2:  **for**(i = 0; i < index; i++)  {  System.***out***.println(b[i]);  }    **break**;  **case** 3:  System.***out***.println("---------------------------");  System.***out***.println("예금");  System.***out***.println("---------------------------");  System.***out***.print("계좌번호: ");  account = sc.next();  System.***out***.print("입금액: ");  deposit = sc.nextInt();  **for**(i = 0; i < index; i++)  {  **if**(b[i].getAccount().equals(account)) {  b[i].deposit(deposit);  System.***out***.println("결과: 입금이 성공되었습니다.");  **break**;  }  }  **if**(i >= index)  {  System.***out***.println("해당하는 계좌번호가 존재하지 않습니다. ");  }  **break**;  **case** 4:  System.***out***.println("---------------------------");  System.***out***.println("출금");  System.***out***.println("---------------------------");  System.***out***.print("계좌번호: ");  account = sc.next();  System.***out***.print("출금액: ");  withdraw = sc.nextInt();  **for**(i = 0; i < index; i++)  {  **if**(b[i].getAccount().equals(account)) {  b[i].withdraw(withdraw);  System.***out***.println("결과: 출금이 성공되었습니다.");  **break**;  }  }  **if**(i >= index)  {  System.***out***.println("해당하는 계좌번호가 존재하지 않습니다. ");  }  **break**;  **case** 5:  System.***out***.println("프로그램을 종료합니다.");  flag = **false**;  **break**;    }  }  }  } |
| **[실행 결과]** |