|  |  |  |
| --- | --- | --- |
| **2019\_2\_java2\_12** | **학번 : 20165164** | **이름 : 지현한** |

* **내용 점검**

1. 객체와 클래스에 대한 설명으로 맞는 것은 O, 틀린것은 X표 하시오
2. 클래스는 객체를 생성하기 위한 설계도와 같은 것이다( O )
3. New 연산자로 클래스의 생성자를 호출함으로써 객체가 생성된다( O )
4. 하나의 클래스로 하나의 객체만 생성할 수 있다( X )
5. 객체는 클래스의 인스턴스이다( O )
6. 필드에 대한 설명으로 맞는 것에 O, 틀린것에 X표 하세요
7. 필드는 메소드에서 사용할 수 있다( O )
8. 필드는 클래스 블록 어디서든 선언할 수 있다(생성자, 메소드 내부 제외)( O )
9. 필드는 객체 외부에서 접근할 수 없다( X )
10. 필드는 초기값을 주지 않더라도 기본값으로 자동 초기화된다( X )
11. 생성자에 대한 설명으로 맞는 것은 O, 틀린것은 X표 하시오
12. 객체를 생성하려면 생성자 호출이 반드시 필요한 것은 아니다( O )
13. 생성자는 다른 생성자를 호출하기 위해 this()를 사용할 수 있다( O )
14. 생성자가 선언되지 않으면 컴파일러가 기본 생성자를 추가한다( O )
15. 매개변수의 수, 타입, 순서가 다른 생성자를 여러 개 선언할 수 있다( O )
16. 다음은 클래스의 구성 멤버에 대한 설명입니다. 필드, 생성자, 메소드 중 어느 것에 대한 설명인지 밑줄 친 부분에 채워 넣으세요.
17. 객체의 데이터(상태)를 저장한다 : 필드
18. 객체의 초기화를 담당한다 : 생성자
19. 객체의 동작부분으로 실행코드를 가지고 있는 블록이다 : 메소드
20. 다음 클래스에서 해당 멤버가 필드, 생성자, 메소드 중 어떤 것인지 제시하시오

public class Student{

String name; (1) 필드

Student(String name){ ……… } (2) 생성자

void setName(String name) { ……… } (3) 메소드

}

1. 현실세계의 회원을 Member 클래스로 모델링하려고 합니다 회원의 데이터로는 이름, 아이디, 패스워드, 나이가 있습니다. 이들 데이터를 필드로 갖는 Member 클래스를 작성해 보세요. 접근지정자는 동일한 패키지내에서 접근할 수 있도록 합니다

|  |  |  |
| --- | --- | --- |
| 데이터 이름 | 필드이름 | 자료형 |
| 이름 | name | 문자열 |
| 아이디 | id | 문자열 |
| 패스워드 | password | 문자열 |
| 나이 | age | 정수 |

**public** **class** Member {

**public** String name;

**public** String id;

**public** String password;

**public** **int** age;

}

1. 위에서 작성한 Member 클래스의 필드를 main() 메소드에서 사용하려고 합니다. (1)~(3)을 채워 넣으세요

public static void main(String[] args){

Member member = (1) new Member();

(2) member.name = “최아름”; //name 필드 값

(3) member.password = “password”; //password 필드 값

1. 다음과 같이 Member객체를 생성하고 싶습니다. 생성자에서 name 필드와 id필드를 매개변수로 받은 값으로 초기화 할 수 있도록 Member 클래스를 작성하세요

Member user1=new Member(“자바”, “java”);

**public** **class** Member {

**public** String name;

**public** String id;

**public** String password;

**public** **int** age;

**public** Member(String name, String id, String password, **int** age) {

**this**.name = name;

**this**.id = id;

**this**.password = password;

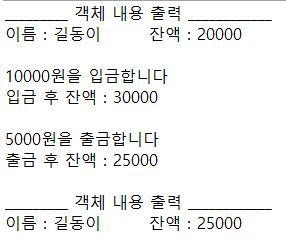
**this**.age = age;

}

}

* 실습 과제

1. 은행 계좌를 처리하기 위한 클래스를 작성하고 테스트하는 프로그램이다. 밑줄 친 부분을 채워 넣은 후 실행 결과를 제시하시오



//은행 계좌 처리를 위한 클래스

public class Account {

//이름을 저장하기 위한 필드 name 선언, private

//잔액을 저장하기 위한 필드 balance 선언(4byte 정수형), private

//생성자 – 매개변수로 받은 값으로 이름과 잔액을 초기화- 본인작성

//형식 매개변수로 받은 값만큼 잔액을 증가하는 메소드, 반환값 없음 - 본인작성

void dePosit(int money){

}

//형식 매개변수로 받은 값만큼 잔액을 감소하는 메소드, 반환값 없음 – 본인작성

void withDraw(int money){

}

//필드 balance 값을 반환하는 getBalance() 메소드, 매개변수 없음

public String toString(){ //객체 내용(이름과 잔액)을 문자열로 반환

}

}

//Account 클래스에 대한 객체를 생성하고 테스트 하는 클래스

public class AccountTest {

public static void main(String[] args) {

//Account 클래스 객체 obj 선언 & 생성 – 본인작성

//dePosit()메소드 호출, 10000을 매개변수로 전달- 본인작성

System.*out*.println("입금 후 잔액 : " + obj.getBalance());

//withDraw() 메소드 호출, 5000을 매개변수로 전달- 본인작성

System.*out*.println("출금 후 잔액 : " + obj.getBalance());

System.*out*.println(obj); //객체 내용 출력

}

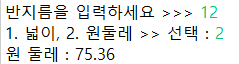
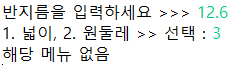
|  |
| --- |
| **[프로그램 소스]**  **public** **class** Account {  **private** String name;  **private** **int** balance;    **public** Account(String name, **int** balance) {  **this**.name = name;  **this**.balance = balance;  }    **void** dePosit(**int** money) {  **this**.balance += money;  }  **void** withDraw(**int** money) {  **this**.balance -= money;  }    **public** **int** getBalance() {  **return** **this**.balance;  }  **public** String toString() {  **return** "이름 : " + name + "\n잔액 : " + balance;  }      }  **public** **class** AccountTest {  **public** **static** **void** main(String[] args) {  Account obj = **new** Account("지현한",0);  obj.dePosit(10000);    System.***out***.println("입금 후 잔액 : " +obj.getBalance());    obj.withDraw(5000);    System.***out***.println("출금 후 잔액: "+obj.getBalance());  System.***out***.println(obj);  }  } |
| **[실행 결과]** |

1. 원 둘레와 원 넓이를 계산하는 Circle 클래스를 작성하시오.

* 필드 : double형의 반지름
* 생성자 : 매개변수로 받은 값을 필드로 초기화
* getArea() 메소드- 원 넓이를 계산하여 출력, 반환값 없음
* getCircum() 메소드 - 원 둘레를 계산하여 출력, 반환값 없음

Circle 객체를 생성하고 테스트하는 CircleTest 클래스를 작성 하시오.

* main() 메소드 – Circle 객체 생성, 입력 받은 값으로 반지름초기화, 객체 내용 출력

|  |
| --- |
| **[프로그램 소스]**  **public** **class** Circle {  **double** radius;    **public** Circle(**double** radius) {  **this**.radius=radius;  }    **public** **void** getArea() {  System.***out***.println("원 넓이: "+radius\*radius \*3.14);  }  **public** **void** getCircum() {  System.***out***.println("원 둘레: "+2\*radius\*3.14);  }  }  **import** java.util.\*;  **public** **class** CircleTest {  **public** **static** **void** main(String[] args) {  Scanner scan = **new** Scanner(System.***in***);    System.***out***.print("반지름을 입력하세요 >>> ");  Circle circle=**new** Circle(scan.nextDouble());    System.***out***.print("1. 넓이, 2.원둘레 >> 선택: ");  **int** a= scan.nextInt();    **if**(a==1) {  circle.getArea();  }**else** **if**(a==2) {  circle.getCircum();  }**else** {  System.***out***.println("해당 메뉴 없음");  }  }  } |
| **[실행결과]** |

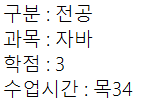
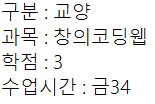
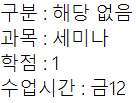
1. 다음과 같은 필드와 메소드로 구성되는 Sugang 클래스를 작성하고 테스트해 보세요

* 필드 : 구분(1~2 사이의 한자리 정수), 과목명(문자열), 학점(정수), 수업시간(문자열)
* 메소드

-write() – 반환값과 매개변수는 없으며, 필드값을 출력

-생성자 – 매개변수로 받은 값을 필드로 초기화

-gubun() – 필드 ‘구분’에 저장된 값이 1이면 “전공”, 2이면 “교양”, 그 외는 “해당없음” 문자열을 반환

**public** **class** SugangTest {

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

Sugang obj=**new** Sugang(1, 3, "자바", "목34");

obj.write();

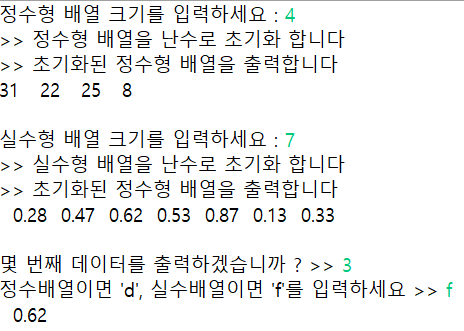
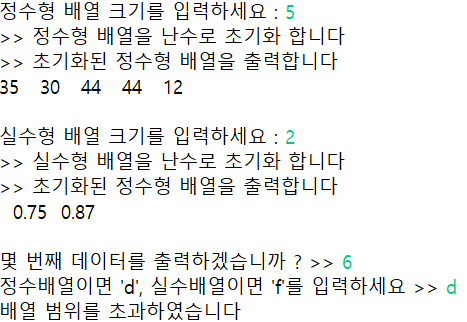
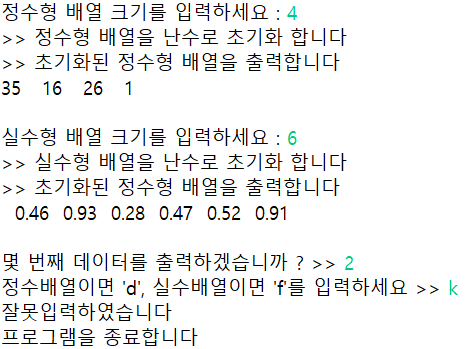
}

}

|  |
| --- |
| **[프로그램 소스]**  **public** **class** Sugang {  **int** num;  String subject;  **int** grade;  String time;    **void** write() {  System.***out***.printf("구분: %s\n과목: %s\n학점: %d\n수업시간: %s",**this**.gubun(),subject,grade,time);  }  **public** Sugang(**int** num, **int** grade, String subject, String time) {  **this**.num = num;  **this**.subject = subject;  **this**.grade = grade;  **this**.time = time;  }    String gubun() {  **if**(**this**.num == 1) {  **return** "전공";  }**else** **if**(**this**.num == 2){  **return** "교양";  }**else** {  **return** "해당없음";  }  }      } |
| **[실행결과]** |

1. 입력 받은 값만큼 배열을 생성하고 난수로 초기화 한 후 다음과 같은 내용을 출력하는 프로그램을 작성하세요

* 배열에 저장된 모든 원소 출력
* 입력 받은 위치에 해당하는 원소만 출력
* 범위를 초과하면 “배열 범위를 초과하였습니다” 출력

**public** **class** Dim {

//필드 선언 : 정수형과 실수형 배열 선언, private

**void** init(**int** size, **char** type) {

//매개변수로 받은 정수값만큼 배열을 생성한다.

//매개변수로 받은 문자가 ‘I’이면 정수형 배열, ‘d’이면 실수형 배열

}

**int** isearch(**int** digit) {

//매개변수로 받은 값에 해당하는 정수형 배열 원소를 반환, 범위를 벗어나면 -1반환

}

**double** dsearch(**int** digit) {

//매개변수로 받은 값에 해당하는 실수형 배열 원소를 반환, 범위를 벗어나면 -1.0반환

}

}

**import** java.util.\*;

**public** **class** DimTest {

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

}

}

|  |
| --- |
| **[프로그램 소스]**  **public** **class** Dim {  **private** **int** in[];  **private** **double** dou[];    **void** init(**int** size,**char** type) {  **if**(type == 'i') {  in = **new** **int**[size];  System.***out***.println(">> 정수형 배열을 난수로 초기화 합니다");  **for**(**int** i=0;i<in.length;i++) {  in[i]=(**int**)(Math.*random*()\*49)+1;  }  }    **else** **if**(type == 'd') {  dou = **new** **double**[size];  System.***out***.println(">> 실수형 배열을 난수로 초기화 합니다");  **for**(**int** i=0;i<dou.length;i++) {  dou[i]=Math.*random*();  }  }  }    **int** isearch(**int** digit) {  **if**(digit<0 && digit>=in.length) {  **return** -1;  }  **return** in[digit];  }    **double** dsearch(**int** digit) {  **if**(digit<0 && digit>=dou.length) {  **return** -1.0;  }  **return** dou[digit];  }    **void** aprint(**char** type) { //배열 출력 메소드  **if**(type == 'i') {  **for**(**int** i=0;i<in.length;i++) {  System.***out***.printf("%d ",in[i]);  }  System.***out***.println();  }**else** **if**(type == 'd') {  **for**(**int** i=0;i<dou.length;i++) {  System.***out***.printf("%.2f ",dou[i]);  }  System.***out***.println();  }    }    }  **import** java.util.\*;  **public** **class** DimTest {  **public** **static** **void** main(String[] args) {  Scanner scan=**new** Scanner(System.***in***);  Dim dim = **new** Dim();    System.***out***.print("정수형 배열 크기를 입력하세요: ");  dim.init(scan.nextInt(), 'i');  System.***out***.println(">> 초기화된 정수형 배열을 출력 합니다");  dim.aprint('i');  System.***out***.println();      System.***out***.print("실수형 배열 크기를 입력하세요: ");  dim.init(scan.nextInt(),'d');  System.***out***.println(">> 초기화된 실수형 배열을 출력 합니다");  dim.aprint('d');  System.***out***.println();      System.***out***.print("몇 번째 데이터를 출력하시겠습니까? >> ");  **int** num=scan.nextInt();  System.***out***.print("정수배열이면 'd', 실수배열이면 'f'를 입력 하세요 >> ");  **char** type = scan.next().charAt(0);    **if**(type == 'd') {  System.***out***.println(dim.isearch(num-1));  }**else** **if**(type == 'f'){  System.***out***.println(dim.dsearch(num-1));  }  }  } |
| **[실행결과]** |