[연습문제]

[9-1] 다음과 같은 실행결과를 얻도록 SutdaCard클래스의 equals()를 멤버변수인 num, isKwang의 값을 비교하도록 오버라이딩하고 테스트 하시오.

[연습문제]/ch9/Exercise9_1.java

```
class Exercise9_1 {
   public static void main(String[] args) {
        SutdaCard c1 = new SutdaCard(3,true);
        SutdaCard c2 = new SutdaCard(3,true);

        System.out.println("c1="+c1);
        System.out.println("c2="+c2);
        System.out.println("c1.equals(c2):"+c1.equals(c2));
   }
}

class SutdaCard {
   int num;
   boolean isKwang;

   SutdaCard() {
        this(1, true);
   }

   SutdaCard(int num, boolean isKwang) {
        this.num = num;
        this.isKwang = isKwang;
   }

   public boolean equals(Object obj) {
        /*
           (1) 매개변수로 넘겨진 객체의 num, isKwang과
                멤버변수 num, isKwang을 비교하도록 오버라이딩 하시오.
        */
   }

   public String toString() {
        return num + ( isKwang ? "K":"");
   }
}
```

[실행결과]

```
c1=3K
c2=3K
c1.equals(c2):true
```

[9-2] 다음과 같은 실행결과를 얻도록 Point3D클래스의 equals()를 멤버변수인 x, y, z의 값을 비교하도록 오버라이딩하고, toString()은 실행결과를 참고해서 적절히 오버라이딩하시오.

[연습문제]/ch9/Exercise9_2.java

```
class Exercise9_2 {
   public static void main(String[] args) {
        Point3D p1 = new Point3D(1,2,3);
        Point3D p2 = new Point3D(1,2,3);

        System.out.println(p1);
        System.out.println(p2);
        System.out.println("p1==p2?"+(p1==p2));
        System.out.println("p1.equals(p2)?"+(p1.equals(p2)));
   }
}

class Point3D {
   int x,y,z;

   Point3D(int x, int y, int z) {
        this.x=x;
        this.y=y;
        this.z=z;
   }

   Point3D() {
        this(0,0,0);
   }

   public boolean equals(Object obj) {
        /*
             (1) 인스턴스변수 x, y, z를 비교하도록 오버라이딩하시오.
        */
   }

   public String toString() {
        /*
             (2) 인스턴스변수 x, y, z의 내용을 출력하도록 오버라이딩하시오.
        */
   }
}
```

[실행결과]

```
[1,2,3]
[1,2,3]
p1==p2?false
p1.equals(p2)?true
```

[9-3] 다음과 같은 실행결과가 나오도록 코드를 완성하시오.

[연습문제]/ch9/Exercise9_3.java

```
class Exercise9_3 {
   public static void main(String[] args) {
        String fullPath = "c:\\jdk1.5\\work\\PathSeparateTest.java";
        String path = "";
        String fileName = "";

        /*
            (1) 알맞은 코드를 넣어 완성하시오.
        */

        System.out.println("fullPath:"+fullPath);
        System.out.println("path:"+path);
        System.out.println("fileName:"+fileName);
   }
}
```

[실행결과]

```
fullPath:c:\jdk1.5\work\PathSeparateTest.java
path:c:\jdk1.5\work
fileName:PathSeparateTest.java
```

[9-4] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

메서드명 : printGraph
기　　능 : 주어진 배열에 담긴 값만큼 주어진 문자를 가로로 출력한 후, 값을 출력한다.
반환타입 : 없음
매개변수 : int[] dataArr - 출력할 그래프의 데이터
　　char ch - 그래프로 출력할 문자.

[연습문제]/ch9/Exercise9_4.java

```
class Exercise9_4 {
   static void printGraph(int[] dataArr, char ch) {
        /*
            (1) printGraph메서드를 작성하시오.
        */
   }

   public static void main(String[] args) {
        printGraph(new int[]{3,7,1,4},'*');
   }
}
```

[실행결과]

```
***3
*******7
*1
****4
```

[9-5] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

메서드명 : count
기　　능 : 주어진 문자열(src)에 찾으려는 문자열(target)이 몇 번 나오는지 세어서 반환한다.
반환타입 : int
매개변수 : String src
　　String target

[Hint] String클래스의 indexOf(String str, int fromIndex)를 사용할 것

[연습문제]/ch9/Exercise9_5.java

```
class Exercise9_5 {
  public static int count(String src, String target) {
      int count = 0; // 찾은 횟수
      int pos = 0;   // 찾기 시작할 위치

      /*
          (1) 반복문을 사용해서 아래의 과정을 반복한다.
          1. src에서 target을 pos의 위치부터 찾는다.
          2. 찾으면 count의 값을 1 증가 시키고,
             pos의 값을 target.length만큼 증가시킨다.
          3. indexOf의 결과가 -1이면 반복문을 빠져나가서 count를 반환한다.
      */
  }

  public static void main(String[] args) {
      System.out.println(count("12345AB12AB345AB","AB"));
      System.out.println(count("12345","AB"));
  }
}
```

[실행결과]

```
3
0
```

[9-6] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

메서드명 : fillZero
기　　능 : 주어진 문자열(숫자)로 주어진 길이의 문자열로 만들고, 왼쪽 빈 공간은 '0'으로 채운다.
　　만일 주어진 문자열이 null이거나 문자열의 길이가 length의 값과 같으면 그대로 반환한다.
　　만일 주어진 length의 값이 0보다 같거나 작은 값이면, 빈 문자열("")을 반환한다.
반환타입 : String
매개변수 : String src - 변환할 문자열
　　int length - 변환한 문자열의 길이

[연습문제]/ch9/Exercise9_6.java

```
class Exercise9_6 {
   public static String fillZero(String src, int length) {
       /*  (1) fillZero메서드를 작성하시오.
           1. src가 널이거나 src.length()가 length와 같으면 src를 그대로 반환한다.
           2. length의 값이 0보다 같거나 작으면 빈 문자열("")을 반환한다.
           3. src의 길이가 length의 값보다 크면 src를 length만큼 잘라서 반환한다.
           4. 길이가 legnth인 char배열을 생성한다.
           5. 4에서 생성한 char배열을 '0'으로 채운다.
           6. src에서 문자배열을 뽑아내서 4에서 생성한 배열에 복사한다.
           7. 4에서 생성한 배열로 String을 생성해서 반환한다.
       */
   }

   public static void main(String[] args) {
       String src = "12345";
       System.out.println(fillZero(src,10));
       System.out.println(fillZero(src,-1));
       System.out.println(fillZero(src,3));
   }
}
```

[실행결과]

```
0000012345

123
```

[9-7] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

메서드명 : contains
기　　능 : 첫 번째 문자열(src)에 두 번째 문자열(target)이 포함되어 있는지 확인한다.
　　포함되어 있으면 true, 그렇지 않으면 false를 반환한다.
반환타입 : boolean
매개변수 : String src
　　String target

[Hint] String클래스의 indexOf()를 사용할 것

[연습문제]/ch9/Exercise9_7.java

```
class Exercise9_7 {
   /*
      (1) contains메서드를 작성하시오.
   */

   public static void main(String[] args) {
       System.out.println(contains("12345","23"));
       System.out.println(contains("12345","67"));
   }
}
```

[실행결과]

```
true
false
```

[9-8] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

메서드명 : round
기　　능 : 주어진 값을 반올림하여, 소수점 이하 n자리의 값을 반환한다.
　　예를 들어 n의 값이 3이면, 소수점 4째 자리에서 반올림하여 소수점 이하 3자리의 수를 반환한다.
반환타입 : double
매개변수 : double d - 변환할 값
　　int n - 반올림한 결과의 소수점 자리

[Hint] Math.round()와 Math.pow()를 이용하라.

[연습문제]/ch9/Exercise9_8.java

```
class Exercise9_8 {
   /*
      (1) round메서드를 작성하시오.
   */

   public static void main(String[] args) {
       System.out.println(round(3.1415,1));
       System.out.println(round(3.1415,2));
       System.out.println(round(3.1415,3));
       System.out.println(round(3.1415,4));
       System.out.println(round(3.1415,5));
   }
}
```

[실행결과]

```
3.1
3.14
3.142
3.1415
3.1415
```

[9-9] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

메서드명 : delChar
기　　능 : 주어진 문자열에서 금지된 문자들을 제거하여 반환한다.
반환타입 : String
매개변수 : String src - 변환할 문자열
　　String delCh - 제거할 문자들로 구성된 문자열

[힌트] StringBuffer와 String클래스의 charAt(int i)과 indexOf(int ch)를 사용하라.

[연습문제]/ch9/Exercise9_9.java

```
class Exercise9_9 {
   /*
      (1) delChar메서드를 작성하시오.
   */

   public static void main(String[] args)     {
       System.out.println("(1!2@3^4~5)"+" -> "
                                    + delChar("(1!2@3^4~5)","~!@#$%^&*()"));
       System.out.println("(1 2    3   4\t5)"+" -> "
                                    + delChar("(1 2    3   4\t5)"," \t"));
   }
}
```

[실행결과]

```
(1!2@3^4~5) -> 12345
(1 2    3   4   5) -> (12345)
```

[9-10] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

메서드명 : format
기　　능 : 주어진 문자열을 지정된 크기의 문자열로 변환한다. 나머지 공간은 공백으로 채운다.
반환타입 : String
매개변수 : String str - 변환할 문자열
int length - 변환된 문자열의 길이
int alignment - 변환된 문자열의 정렬조건
(0:왼쪽 정렬, 1: 가운데 정렬, 2:오른쪽 정렬)

[연습문제]/ch9/Exercise9_10.java

```
class Exercise9_10
{
   /*
        (1) format메서드를 작성하시오.
        1. length의 값이 str의 길이보다 작으면 length만큼만 잘라서 반환한다.
        2. 1의 경우가 아니면, length크기의 char배열을 생성하고 공백으로 채운다.
        3. 정렬조건(alignment)의 값에 따라 문자열(str)을 복사할 위치를 결정한다.
           (System.arraycopy()사용)
        4. 2에서 생성한 char배열을 문자열로 만들어서 반환한다.
   */

   public static void main(String[] args) {
        String str = "가나다";

        System.out.println(format(str,7,0)); // 왼쪽 정렬
        System.out.println(format(str,7,1)); // 가운데 정렬
        System.out.println(format(str,7,2)); // 오른쪽 정렬
   }
}
```

[실행결과]

```
가나다
  가나다
    가나다
```

[9-11] 커맨드라인으로 2~9사이의 두 개의 숫자를 받아서 두 숫자사이의 구구단을 출력하는 프로그램을 작성하시오.

예를 들어 3과 5를 입력하면 3단부터 5단까지 출력한다.

[실행결과]

```
C:\jdk1.5\work>java Exercise9_11 2
시작 단과 끝 단, 두 개의 정수를 입력해주세요.
USAGE : GugudanTest 3 5

C:\jdk1.5\work>java Exercise9_11 1 5
단의 범위는 2와 9사이의 값이어야 합니다.
USAGE : GugudanTest 3 5

C:\jdk1.5\work\ch9>java Exercise9_11 3 5
3*1=3
3*2=6
3*3=9
3*4=12
3*5=15
3*6=18
3*7=21
3*8=24
3*9=27

4*1=4
4*2=8
4*3=12
4*4=16
4*5=20
4*6=24
4*7=28
4*8=32
4*9=36

5*1=5
5*2=10
5*3=15
5*4=20
5*5=25
5*6=30
5*7=35
5*8=40
5*9=45
```

[9-12] 다음과 같이 정의된 메서드를 작성하고 테스트하시오.

[주의] Math.random()을 사용하는 경우 실행결과와 다를 수 있음.

메서드명 : getRand
기　　능 : 주어진 범위(from~to)에 속한 임의의 정수값을 반환한다.
(양쪽 경계값 모두 범위에 포함)
from의 값이 to의 값보다 클 경우도 처리되어야 한다.
반환타입 : int
매개변수 : int from - 범위의 시작값
int to - 범위의 끝값

[Hint] Math.random()과 절대값을 반환하는 Math.abs(int a), 그리고 둘 중에 작은 값을 반환하는 Math.min(int a, int b)를 사용하라.

[연습문제]/ch9/Exercise9_12.java

```
class Exercise9_12
{
   /*
      (1) getRand메서드를 작성하시오.
   */

   public static void main(String[] args)
   {
       for(int i=0; i< 20; i++)
           System.out.print(getRand(1,-3)+",");
   }
}
```

[실행결과]

```
0,-1,1,0,-2,-2,1,1,-3,0,-1,1,1,1,0,-1,1,0,-1,-3,
```