**Lab 3**

컴퓨터공학과

32191826 박주희(3분반)

1. **Element.java**

**public** **enum** Element : enum 선언. 원소의 번호

**public** **static** Element valueOf(**int** value) : integer형의 value를 받아 Element 리턴

**public** **static** Element nameOf(String name) : String형의 원소번호 또는 원소이름을 입력받으면 integer형의 원소번호를 return

**public** **static** String[] names() : Foreach문을 이용하여 Element.names()를 리턴

values() : enum의 요소들을 순서대로 enum타입의 배열로 리턴

ordinal() : 원소에 열거된 순서를 정수 값으로 리턴

1. **State.java**

**public** **enum** State : enum 선언. 원소의 상태

1. **PeriodicElement.java**

**public** PeriodicElement(**int** number, String symbol, String name, **double** weight): 생성자(Constructor)-클래스의 객체가 생성될 때 호출되는 메소드

**public** **int** getNumber() . . . : number, symbol, name, weight 각각의 getter,setter를 생성하는 메소드

**public** String toString():PeriodicElement [number, symbol, name, weight]형태로 리턴하는 메소드

**public** **static** State getState(PeriodicElement element): 입력받은 element의 state를 결정하는 메소드

switch문을 이용하여 입력받은 element의 원소번호로 state를 결정

**public** **static** PeriodicElement parsePeriodicElement(String line): fields 배열은 “ “단위로 split하여 fields에 number, symbol, name, weight를 차례대로 저장

PeriodicElement객체를 생성하여 element에 저장하여 element를 리턴

1. **UserInput.java**

**public** **static** String getString(): 한줄을 읽는 메소드

**public** **static** **int** getInteger(), **public** **static** **double** getDouble(): 한줄을 읽어서 각각 integer, double형으로 parse하는 메소드

**public** **static** **boolean** getExitKey(): 한줄을 읽고 읽은 값이 q이면 true 반환, 아닐 경우 false를 리턴

**public** **static** Element getElement(): “1” 또는 “Hydrogen”입력을 받으면 원소번호로 리턴(String 🡪 Element)하는 메소드

"1" or "Hydrogen"을 입력하면 HYDROGEN을 반환하는 Element의 nameOf메소드에 str을 입력하여 그값을 element에 저장하고 element을 리턴한다.

1. **Lab3.java**

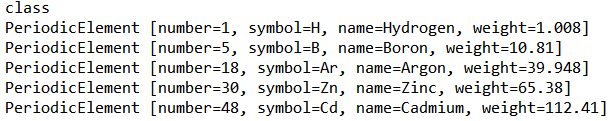
**private** **static** PeriodicElement[] *elements*: PeriodicElement[] elements에 50개(1 수소 ~ 50 주석) 화학원소데이터 저장

**private** **static** PeriodicElement getPeriodicElement(PeriodicElement[] array,Element e): element 배열데이터 안에서 Element에 따라 PeriodicElement를 반환하는 메소드

for문을 이용하여 입력받은 Element e의 원소번호값과 array[i]의 원소번호를 비교한다.

같다면 element에 array[i]를 대입한 후 element를 리턴한다.

PeriodicElement e1= **new** PeriodicElement(1,"H","Hydrogen",1.008);

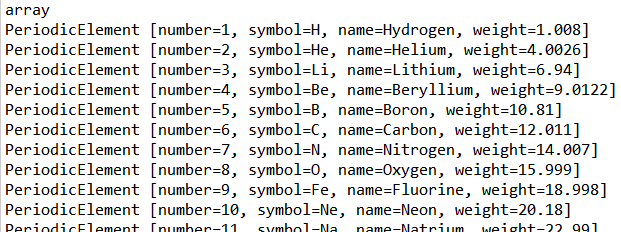
System.***out***.println(e1);

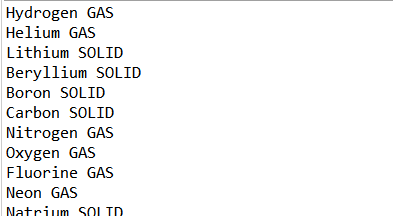
PeriodicElement 객체를 생성하고 이를 출력한다.

**for**(PeriodicElement i:*elements*)

System.***out***.println(i);

Foreach문을 이용하여 PeriodicElement[] elements를 모두 출력한다.





**for**(PeriodicElement i:*elements*)

{

System.***out***.print(i.getName()+" ");

System.***out***.println(PeriodicElement.*getState*(i));

}

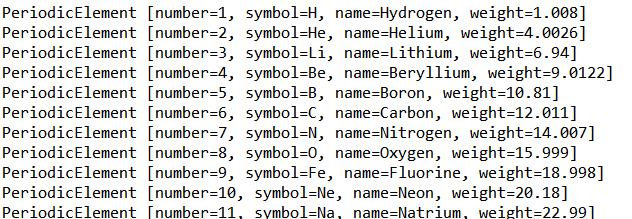
Foreach문과 PeridoicElement의 getState를 이용해서 Element의 이름과 State을 출력한다.

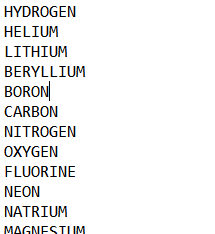
**for**(String s:lines) {

System.***out***.println(PeriodicElement.*parsePeriodicElement*(s));

}

Foreach문과 PeriodicElement의 parsePeriodicElement을 이용하여 lines를 출력한다.





**for**(String s: Element.*names*()) {

System.***out***.println(s);

}

Foreach문과 Element의 names()를 이용해서 원소이름 출력한다.

#206~#223

원소번호 또는 원소이름을 입력받아 그에 해당하는 원소 번호를 Element e에 저장한다. e가 원소번호인 PeriodicElement를 element에 저장하고 PeriodicElement의 getState를 이용하여 state를 찾는다. e.getValue(원소번호)로 금속, 준금속, 비금속을 구분한다.

