Chapter 10-1

Garbage Collecting 과 패키지

Garbage Collecting 이란

Garbage Collecting

- 객체를 생성하는데 사용되는 키워드는 new이지만, 생성된 객체를 메모리에서 해제시키는 키워드는 존재하지 않습니다.
- 그 이유는 자바에서는 자바 가상머신(JVM)이 알아서 메모리를 관리해주며, 이를 가비지 컬렉팅이라 합니다.
- 객체가 더 이상 프로그램에서 사용되지 않으면 가비지 컬렉터가 스스로 판단해서 메모리를 해제하고 수거합니다.

```
public class MainClass {

     public static void main(String[] args) {

          Pen p;

          for(int i = 0; i <= 100000000; i++) {
               p = new Pen();
          }

     }
}
```

패키지

패키지(package)

- 클래스는 크게 2종류가 있습니다.
- 하나는 개발자가 직접 만들어 사용하는 사용자 정의 클래스이고, 다른 하나는 JDK(Java Development Kit)안에 포함되어 있거나, 다른 개발자들이 미리 만들어 놓은 클래스들이 있습니다.

- 그런데 이런 클래스들은 매우 종류가 많고 비슷한 기능들을 분류시켜야 할 필요성이 있기 때문에 서로 관계가 있는 클래스들을 패키지로 묶어서 관리합니다.

- 현재 클래스에서 다른 패키지의 외부 클래스를 사용하고 싶은 경우에 `import` 키워드로 해당 클래스의 전체 경로(패키지 명까지 포함)를 지정해줘야 합니다.

- 만약 해당 패키지 안에 들어있는 모든 클래스를 사용하고 싶다면 * 기호를 넣어주면 됩니다.

ex) `import` java.util.Scanner;
ex) `import` java.util.*;

소제목

package 선언 방법

1. 사용자가 임의로 패키지를 만드려면 클래스의 가장 윗부분에 패키지 구문을 포함시키면 됩니다.
   ex) package store;

2. 패키지 이름은 영문 소문자로 지정하는 것을 권장합니다.
3. 대분류와 소분류 패키지를 구분할 때는 .(dot)을 사용합니다.
4. 최상위 패키지의 이름은 java로 시작할 수 없습니다. java패키지는 표준 API들의 모음을 구성해놓은 패키지입니다.
5. 자바의 키워드들도 패키지이름으로 사용할 수 없습니다.

특정 패키지 안의 클래스를 사용하겠다.

```
import java.io.PrintWriter;
import java.util.*;
```

클래스 선언전 import선언 맨위는 package선언

```
package xxx;
import xxx.yyy.*;
class 선언
...
```

Chapter 10-2

기본타입 vs 참조타입

데이터 타입 (참조타입 vs 기본타입)

기본 타입(primitive type) vs 참조 타입(reference type)

- 기본 타입이란 정수, 실수, 문자, 논리 값을 저장하는 데이터 타입입니다.
- 참조 타입이란 객체의 주소를 참조하는 타입으로 배열, 클래스, 인터페이스 타입을 말합니다.

- 기본 타입으로 선언된 변수는 2실제 값(value)을 변수 안에 저장합니다.
- 참조 타입으로 선언된 변수는 메모리의 주소값을 변수 안에 저장합니다.

- 참조 타입으로 선언된 변수는 스택(stack)영역에 주소값을 저장하고 내부의 실제 값은 힙(heap) 영역에 저장합니다.

스택 영역

힙 영역

프로그램 실행에 따른 스택, 힙 영역 변화1

```
public class MainClass {

public static void main(String[] args) {

     int a = 10;
     int b = 20;
     int c = 30;

     Pen p1;
     p1 = new Pen();

     Pen p2 = new Pen();

     }
}
```

스택 영역

```
int c = 30;
int b = 20;
int a = 10;
```

힙 영역

프로그램 실행에 따른 스택, 힙 영역 변화2

```
public class MainClass {

public static void main(String[] args) {

     int a = 10;
     int b = 20;
     int c = 30;

     Pen p1;
     p1 = new Pen();

     Pen p2 = new Pen();

     }
}
```

스택 영역

```
Pen p1;
int c = 30;
int b = 20;
int a = 10;
```

힙 영역

프로그램 실행에 따른 스택, 힙 영역 변화3

```
public class MainClass {

public static void main(String[] args) {

     int a = 10;
     int b = 20;
     int c = 30;

     Pen p1;
     p1 = new Pen();

     Pen p2 = new Pen();

     }
}
```

Pen객체 생성
p1에 저장

스택 영역

```
Pen p1 = 100번지
int c = 30;
int b = 20;
int a = 10;
```

힙 영역

100번지 `new Pen();`

프로그램 실행에 따른 스택, 힙 영역 변화4

```
public class MainClass {

public static void main(String[] args) {

     int a = 10;
     int b = 20;
     int c = 30;

     Pen p1;
     p1 = new Pen();

     Pen p2 = new Pen();

     }
}
```

Pen객체 생성
p2에 저장

이 모든 걸 자동으로 해주니
이해 하고 넘어가도록 하자

스택 영역

```
Pen p2 = 200번지
Pen p1 = 100번지
int c = 30;
int b = 20;
int a = 10;
```

힙 영역

200번지 `new Pen();`

100번지 `new Pen();`

참조타입 String과 객체 동등비교 메서드 equals()

참조 타입 String과 객체 동등 비교 메서드 equals()

- 자바는 문자열이 동일하다면 String 객체를 공유하도록 되어있습니다.
- 그래서 단순히 문자열을 String 변수에 할당한다면 같은 주소값을 갖게 됩니다.

- 그러나 new키워드를 사용해서 String객체를 직접 heap영역에 생성한다면 문자열의 내용이 같더라도 다른 주소값을 가지게 되므로 동등, 비동등 연산자(==, !=)의 결과가 false로 나오게 됩니다.

- 그래서 동일 String객체이든 다른 String 객체이든 상관없이 문자열의 내용 값 그 자체를 비교할 때는 equals() 메서드를 사용해야 합니다.

```
ex) String str1 = "Hello";
    String str2 = "Hello";
  --> str1 == str2 -> true
```

```
ex) String str3 = new String("Hello");
    String str4 = new String("Hello");
  --> str3 == str4 -> false
```

스택 영역

힙 영역

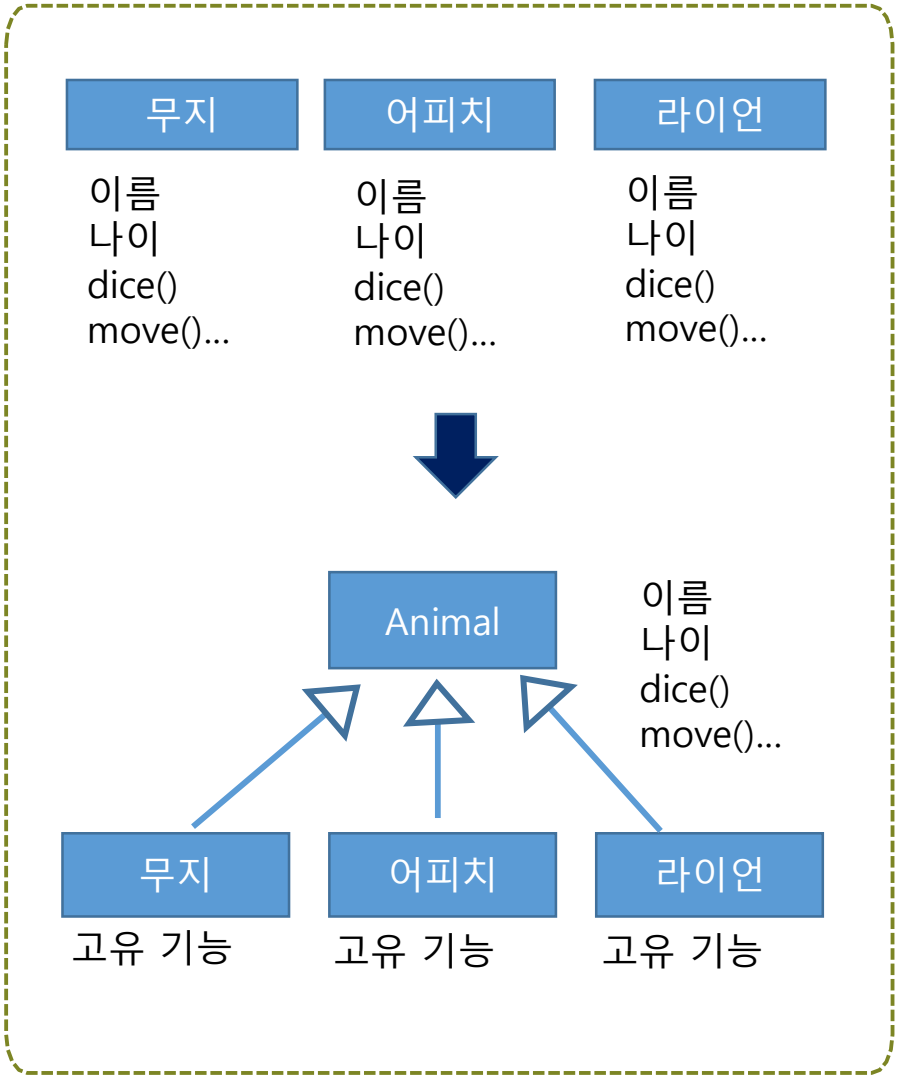
Chapter 10-3

객체지향 프로그래밍 기술 (상속, 은닉, 다형성)

상속

상속(Inheritance)

- OOP에서 상속은 기존의 클래스를 확장하여 새로운 클래스를 이끌어내는 것을 의미합니다.
- 상속은 기존의 **코드를 재사용**함으로써 불필요한 코드를 재작성하는 번거로움을 없앨 수 있고, 새로운 클래스를 만드는 시간과 노력을 줄일 수 있습니다.



상속

- 단일상속만을 지원합니다.
- 어떤 클래스가 다른 클래스로부터 상속을 받아 만들어지면 새롭게 만들어진 클래스를 자식(child or sub) 클래스라고 부르며, 멤버변수와 메서드를 물려준 클래스는 부모(parent or super)클래스라고 부릅니다.
- 상속을 하면 부모클래스의 멤버변수와 메서드가 자식클래스에 상속이 됩니다. 그러나 부모클래스의 생성자는 상속이 되지 않습니다.
- 상속을 사용하는 키워드는 extends 입니다.
- 자바의 모든 클래스는 Object 클래스를 상속받고 있습니다. Object클래스는 자바의 최상위 클래스입니다.

상속의 예제

Bad Case
개별 클래스들에 중복된 속성/기능을 포함

```
public class Employee {

String name;
int age;
String department;

String info() {
      return "이름: " + name + ", 나이: " + age;
}
}
```

```
public class Teacher {

String name;
int age;
String subject;

String info() {
      return "이름: " + name + ", 나이: " + age;
}
}
```

```
public class Student {

String name;
int age;
String studentId;

String info() {
      return "이름: " + name + ", 나이: " + age;
}
}
```

상속의 예제

Good Case
중복된 기능을 빼내 새로운 클래스로 작성한 후 구체화 시킴
일반화 된 클래스는 부모클래스
구체화 된 클래스는 자식클래스
extends 키워드를 사용

```
public class Person {

      String name;
      int age;

      String info() {
      return "이름: " + name + ", 나이: " + age;
      }
}
```

중복된 기능을 부모클래스로 묶는다

```
public class Employee extends Person {

      String department;
}
```

```
public class Student extends Person {

      String studentId;
}
```

```
public class Teacher extends Person {

      String subject;
}
```

Chapter 10

수고하셨습니다