파이썬 프로그래밍

상속과 다형성

한국기술교육대학교
온라인평생교육원

▣ 상속 관계에 있는 클래스들의 정보 획득

1. 객체가 어떤 클래스에 속해 있는지 확인하기

- 객체의 자료형 비교 방법 I (전통적 방법)

```
import types

print type(123) == types.IntType
print type(123) == type(0)
```

```
True
True
```

- 1,2,3은 정수이므로 IntType을 가지고 있음
- type() → 인수의 타입을 알아보는 내장함수
- types라는 모듈을 import하여 알아보는 방법 → 많이 쓰이지 X

- 객체의 자료형 비교 방법 II (새로운 방법)
  - isinstance() 내장 함수와 기본 객체 클래스 사용

```
print isinstance(123, int)
print int
```

```
 True
<type 'int'>
```

- is로 시작하는 매소드 → 반드시 true, false로 반환
- isinstance(123, int) → 123이 int 타입의 인스턴스인지 물음
- isinstance(객체, 타입)
- 클래스 이름은 하나하나가 다 타입
- int도 하나의 클래스

▣ 상속 관계에 있는 클래스들의 정보 획득

1. 객체가 어떤 클래스에 속해 있는지 확인하기

- 서브 클래스의 인스턴스는 슈퍼 클래스의 인스턴스이기도 하다.

- obj가 클래스 A, B, C의 인스턴스인지 확인
- 클래스 C는 B의 상속을 받아 B, C 클래스의 인스턴스 모두 사용

▣ 상속 관계에 있는 클래스들의 정보 획득

1. 객체가 어떤 클래스에 속해 있는지 확인하기

```
class A:
    pass

class B:
    def f(self):
        pass

class C(B):
    pass

def check(obj):
    print obj, '=>',
    if isinstance(obj, A):
        print 'A',
    if isinstance(obj, B):
        print 'B',
    if isinstance(obj, C):
        print 'C',
    print

a = A()
b = B()
c = C()

check(a)
check(b)
check(c)
```

```
<__main__.A instance at 0x10de34e60> => A
<__main__.B instance at 0x10de34e18> => B
<__main__.C instance at 0x10de34cf8> => B C
```

▣ 상속 관계에 있는 클래스들의 정보 획득

2. 클래스 간의 상속 관계 알아내기

- issubclass() 내장 함수 활용

```
class A:
    pass

class B:
    def f(self):
        pass

class C(B):
    pass

def check(obj):
    print obj, '=>',
    if issubclass(obj, A):
        print 'A',
    if issubclass(obj, B):
        print 'B',
    if issubclass(obj, C):
        print 'C',
    print

check(A)
check(B)
check(C)
```

```
__main__.A => A
__main__.B => B
__main__.C => B C
```

▣ 상속 관계에 있는 클래스들의 정보 획득

2. 클래스 간의 상속 관계 알아내기

- issubclass(클래스, 클래스)
- C는 B를 상속 받아 B클래스의 인스턴스 모두 사용