Static Polymorphism

Count of Parameters

```
void myFunc(int count)
{
    System.out.println("single param func");
}

void myFunc(int count, int bound)
{
    System.out.println("double param func");
}
```

Data Types of Parameters

```
void myFunc(int count)
{
    System.out.println("func with int param");
}
void myFunc(long count)
{
    System.out.println("func with long param");
}
```

Sequence of Data Type of Parameters

```
void myFunc(char c, int count)
{
    System.out.println("func with char first");
}
void myFunc(int count, char c)
{
    System.out.println("func with int first");
}
myFunc('a', 10);
myFunc(10, 'a');
myFunc('a', 'b');    //Compile Error - Ambiguity
```

Return Type

```
int square(int num)
{
    System.out.println("func with int output");
}
long square(int count)
{
    System.out.println("func with long output");
}
```

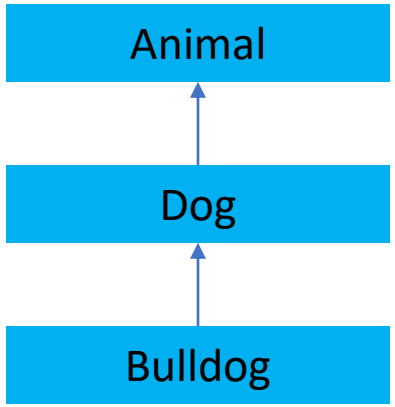
Compile Error

```
void myFunc(float num)
{
    System.out.println("func with float param");
}
void myFunc(long num)
{
    System.out.println("func with long param");
}

myFunc(10);
//func with long param
```

Type Promotion

```
byte → short → int → long → float → double
```

```java
private static void test(Animal animal)
{ System.out.println("func with animal param"); }

private static void test(Dog dog)
{ System.out.println("func with dog param"); }
```




```java
Bulldog bulldog = new Bulldog();
test(bulldog);
//func with dog param
```

```
void test(String... strings)
{
    System.out.println("test with strings");
}

void test(int... numbers)
{
    System.out.println("test with ints");
}

test();     //Compile Error - Ambiguity
```

```
void test(float... numbers)
{
    System.out.println("test with floats");
}

void test(int... numbers)
{
    System.out.println("test with ints");
}

test();      //test with ints
```