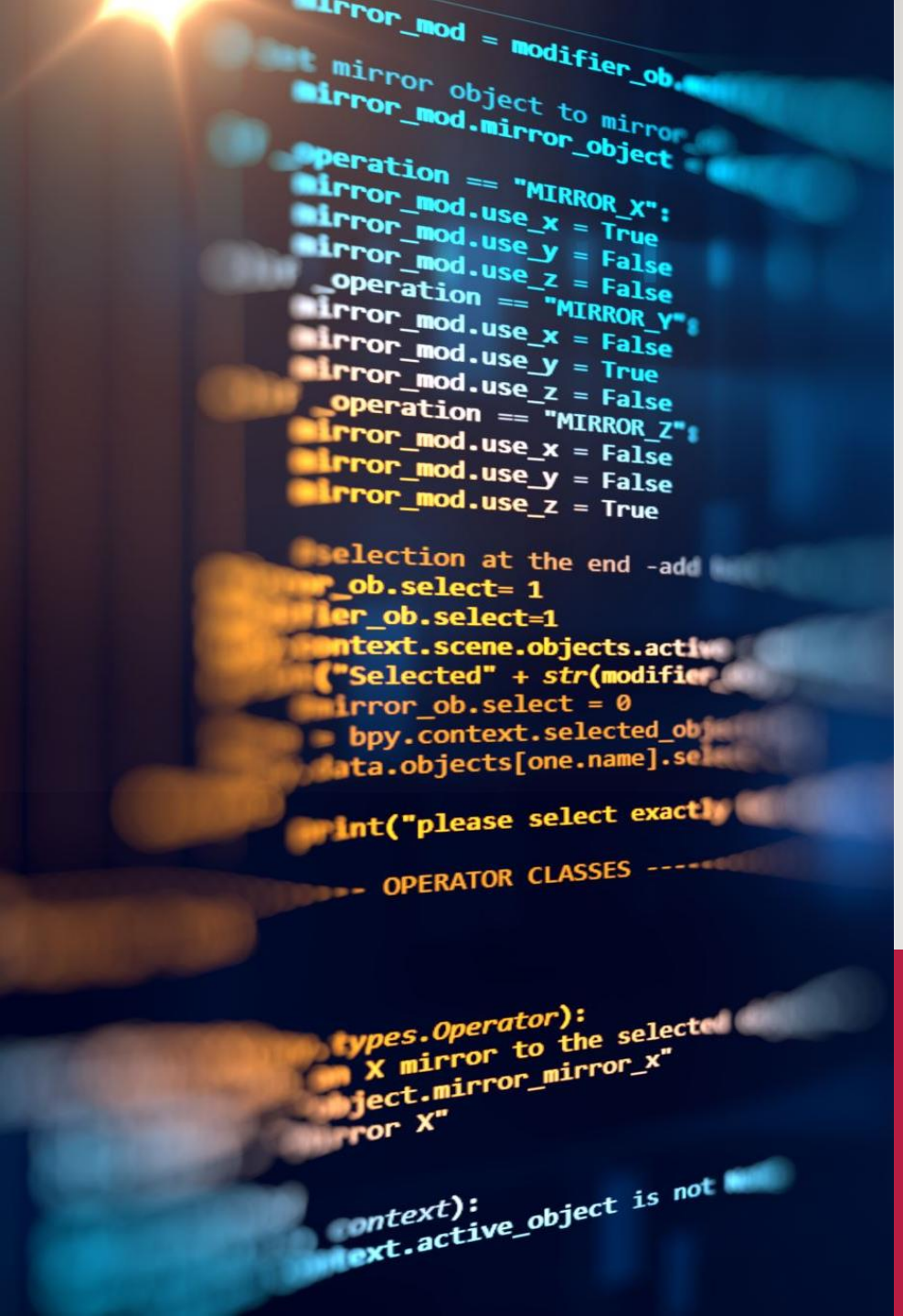

COMP 348 PRINCIPLES OF PROGRAMMING LANGUAGES

LECTURE 1-B – OBJECT-ORIENTED PROGRAMMING WITH JAVA (VARIOUS QUESTIONS)

Dept of Computer Science and Software Engineering, Concordia University

Fall 2020

Object-Oriented Programming with Java

Various Questions

Acknowledgement and Copyright Notice

- The following materials are the original lecture note from the Course Pack “COMP 348 Principles of Programming Languages” written and developed by:

Dr. Constantinos Constantinides, P.Eng.
Department of Computer Science and Software Engineering
Concordia University, Montreal, Quebec, Canada

cc@cse.concordia.ca

https://users.encs.concordia.ca/~cc/

added

Various Questions

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();
```

Q. What is the static type of lassie?

A. The set {Dog, Animal, Object}

Q. What is the difference between the declared type and the static type of an object?

A. The declared type is an element of the set of static type.

Q. What is the dynamic type of the object?

A. Collie.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();
```

Q. Why will the assignment statement compile?

A. Because static type checking is successful.

Q. What is static type checking?

A. The validation of an assignment statement whereby the type of the expression on the RHS must be a subtype of the type of the variable on the LHS.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();
```

- Now, let us have the following:

```
lassie.whatIdo();
```

Q. What is this?

A. It is a message passing statement. No, it is not a method call.

Q. Under what conditions, if any, will this compile?

A. It will compile provided that there exists a matching method in the static type of the object.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();
```

- Now, let us have the following:

```
lassie.whatIdo();
```

Q. Can a compiler throw an exception?

A. No.

Q. What is the responsibility of the run-time system in this case?

A. To select and invoke a method.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();
```

- Now, let us have the following:

```
lassie.whatIdo();
```

> "responsibility of the run-time system"

Q. What is the protocol for the above?

A. The tun-time system performs a search for a matching method starting from the class that corresponds to the dynamic type of the object. If a mathing method is found, then it invokes that method. if not, the run-time system continues the search up the inheritance chain.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();
```

- Now, let us have the following:

```
lassie.whatIdo();
```

Q. What if it does not find the method?

A. There is a guarantee that a method will be found.

Q. Why?

A. Because the compiler has ensure this.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();

lassie.whatIam();
```

Assume whatIam() is a static method in all classes.

Q. Which whatIam() is called?

A. The one decided upon by the COMPILER. This is called static binding.

Q. Why?

A. Because it's static.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();

lassie.whatIam();
```

Assume whatIam() is a static method in all classes.

In Java, it is confusing and misleading that we are allowed to use an object reference to call a static method. That is indeed a method call!

Smalltalk does not allow that. So in Smalltalk we would only be able to say Dog.whatIam(), or Animal.whatIam(), etc.

added

Various Questions /cont.

- Consider the following example

```
class Animal {...}
class Dog extends Animal {...}
class Collie extends Dog {...}

Dog lassie = new Collie();

lassie.whatIam();
```

Q. What is the motivation for explicit casting?

A. To “temporarily” extend the static type of an object.

and many more…

added

Further Reading

- Part IV of the Book "COMP 348 Principles of Programming Languages"
  - **Ch. 21:** Object Oriented programming with message passing
  - **Ch. 22:** Inheritance
  - **Ch. 23:** Additional Examples