Philip Pesic

Week 18

May 21 2023

Week 18 Conceptual Hierarchy

**Alternative sequences**
Sequences are procedural sets of steps or actions. Alternative sequences are sequences which contain multiple branches, represented in code by if statements.
Ex:
If it is raining -> wear a jacket
if not -> don't
**Array**
An array is a collection of values with the same data type user a single alias and contiguous RAM addresses. Array are often used to store small chunks of data or values. Since arrays only have one alias, you can access different elements with the array index. The first element of ann array will have index 0, and you can manipulate the index you call to access different elements.
Ex:
int arr[] = {0, 1, 2, 3}; //arr[] with 4 values
System.out.println(arr[1]); //will print 1 because 1 is the second element
**Class**
Classes are encapsulations of code that allow a programmer to edit an object and its characteristics. To create a class, open a new file, enter a name, and add properties and methods.
Ex:
Public class myClass {
Private int x;
Public int mult(int y) {
Return x * y}}
**Composition**
Composition is a relationship between classes. With composition, a class has a HAS A relationship, or it is COMPOSED of other classes. A real-life example of composition is a school; a school HAS or IS COMPOSED OF students, teachers, staff, and more. To use composition, you need to declare an object of one class in another to compose that class of that object, and then call its methods.
Ex:
Private PartClass part1 = new PartClass(); //Instantiate partclass that composes whole class
Public WholeClass() {part1.setX(2);} //Call part method in constructor
//In Main class
WholeClass whole1 = new WholeClass(); //WholeClass instantiation - calls setX from constructor.
whole1.part1.getX(); //returns X from method in part class which is accessed through whole/composed class

Philip Pesic

Week 18

May 21 2023

Week 18 Conceptual Hierarchy

**CONCEPT**

A term that describes a group of things that share similar characteristics. Concepts are a more efficient way of storing information, and can help overcome your crow limit. First level concepts come from real things and direct observation.

If you were trying to remember the most important city in each US state, you could group each important city into a concept like a capital. Instead of trying to remember all 50 capitals, you could just remember that the most important city in a state is the capital.

**Functions**

Functions are a form of non-contiguous code reuse - they allow for a programmer to execute a block of code with a single command, without rewriting the entire block. Since functions can be called everywhere, they are considered non-contiguous. Function syntax:

Ex:

returnType name (args) {
Block
Return}

**Generic Method**

A generic method is a method whose parameters and contents use a “generic” datatype. That type is determined on function call by the function’s arguments. Generics can be used to reuse code by allowing the use of one function for many cases/datatypes.

Ex:

```
public static <T> void func(T[] x) {
   System.out.println(x[0]);
}
--------
func(charArray); //T replaced with array type
func(intArray);
```

**IDE**

An IDE is an integrated development environment. It is an app that contains all of the tools a programmer might need, such as a text editor, compiler, debugger, git integration, and more.

Ex: VS Code, Xcode, IntelliJ

**Inheritance**

Similar to composition, inheritance is another form of class relationship. Composition, however, is a HAS A relationship, where a class has or is composed of many elements, whereas inheritance is an IS A relationship, where a class IS another class. With inheritance, subclasses or specific classes inherit the properties of base/general/super classes.

Ex:

Philip Pesic

Week 18

May 21 2023

Week 18 Conceptual Hierarchy

A basketball IS A ball

**Instantiation**

To use a class and its methods in the main class, you need to instantiate an object or instance of that class. Then, you can call methods of that class through that object.

Ex:

```
Class mainClass {
Public static void main(String[] args) {
secondClass object = new secondClass(); //Object is an instance of secondClass
object.aFunction(); //uses object to call the aFunction method from secondClass
}}
```

**Method Overloading**

Method overloading is the act of writing several methods with the same name. By writing methods with the same name but different parameters, it is possible to call two separate methods with the same method name. The method with the corresponding number/datatype of arguments is called.

Ex:

```
void func(int x) { System.out.println("I'm a number"); }
void func(char x) { System.out.println("I'm a letter"); }
--------
func('c');
func(3);
```

**OOP**

Object oriented programming is a paradigm that focuses code around a thing or object. These objects have properties, such as variables and functions, that alter the object that they are part of.

Ex: Dog: Variables: Age, breed, length Functions: Sit, play, sleep

**Polymorphism**

Polymorphism is the act of creating many subclasses from a superclass and then overriding functions. This is, in a way, an extension of inheritance by extending the possibilities with inheriting methods.

Ex:

```
(Base class "animal") void sound() {System.out.println("Sound")};
(Sub class "dog") void sound() {System.out.println("bark")};
(Sub class "cat") void sound() {System.out.println("meow")};
animal.sound(); //Sound
dog.sound(); //bark
cat.sound(); //meow
```

Philip Pesic

Week 18

May 21 2023

Week 18 Conceptual Hierarchy

**Reading files**

To read from a text file, first create an object of the file. Then, create a loop that allows you to read line by line. Using obj.nextLine() allows the user to reference a line in the file and enter a new line for the next loop. This line can then be stored in a variable or array.

Ex:

```
//Read 10 lines and store in array
for(int i = 0; i < 10; i++) {
   fileText[i] = myFile.nextLine();
}
```

**Recursion**

Recursion is the act of a function calling itself. It is a form of code reuse, and works by reducing a problem to its smallest/simplest case. A recursive function must always have a default return value so it does not call itself infinitely.

Ex:

```
int recur(int x) {
   return recur(x--);
}
```

**Repeating sequences**

Repeating sequences are sequences or chunks of code that repeat themselves. In programming, they are represented by loops, which repeatedly execute their body while their condition is true.

Ex:

While hungry → eat

**Variables**

Variables are aliases for values and locations in memory where values are stored. Variables are first declared and initialized with a value, and then the OS memory manager will assign the value a location in RAM. Variables also have naming conventions, such as camel case, which dictate how variables are written. These are not required, but highly recommended as they make long variable names easier to read.

Ex:

Int myVariable = 4; //Alias: myVariable, Location: 0fa08c0a

Philip Pesic

Week 18

May 21 2023

Week 18 Conceptual Hierarchy