

Intro to Java

Tutorial 7

Owen Huyn

February 27, 2017

Development environments

You will need Java SE Development Kit:

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

An IDE (many options):

IntelliJ

<https://www.jetbrains.com/idea/>

Eclipse:

<https://eclipse.org/>

Netbeans:

<https://netbeans.org/features/java/>

Agenda

1. Review of some OOP concepts in Java
2. MIS to Code in Java
3. Exceptions

OOP in Java

As most of you have done OOP in Java, we will go through basic examples to refresh your memory.

1. Inheritance
2. Polymorphism

Inheritance

- To have a class inherit from another do the following:

```
class Super {  
    . . . . .  
    . . . . .  
}  
class Sub extends Super {  
    . . . . .  
    . . . . .  
}
```

Overriding

```
class Animal{  
    public void eat () {}  
}  
class Dog extends Animal {  
    public void eat () {} ← dog will use this instead  
}
```

Overloading

```
class Duck extends Animal {  
    public void walk (int steps) {}  
    public void walk (float distance) {}  
}
```

Previous assignment - 2009

We will be going through an assignment back in 2009 that will be used as an example for your upcoming Java assignment.

This assignment consists of nodes, points, edges all on a map-like structure

PointADT Module

Please check NodeT.java in the tutorial source folder.

Point ADT Module

Template Module

PointT

Uses

N/A

Syntax

Exported Constants

TOLERANCE = 1×10^{-4}

Exported Types

PointT = ?

Exported Access Programs

Routine name	In	Out	Exceptions
new PointT	real, real	PointT	
xcoord		real	
ycoord		real	
dist	PointT	real	
equal	PointT	boolean	

Semantics

State Variables

xc: real

yc: real

State Invariant

None

Assumptions

None

Access Routine Semantics

new PointT (*x*, *y*):

- transition: $xc, yc := x, y$
- output: $out := self$
- exception: none

xcoord:

- output: $out := xc$
- exception: none

ycoord:

- output: $out := yc$
- exception: none

dist(*p*):

- output: $out := \sqrt{(xc - p.xcoord)^2 + (yc - p.ycoord)^2}$
- exception: none

equal(*p*):

- output: $out := self.dist(p) \leq TOLERANCE$
- exception: none

Node Module

Please check NodeT.java in the tutorial source folder.

Node Module

Template Module

NodeT inherits PointT

Uses

PointT

Syntax

Exported Types

NodeT = ?

nodeTypeT = { JUNCTION, EGG, BLOCKAGE, OPENING }

Exported Access Programs

Routine name	In	Out	Exceptions
new NodeT	real, real, nodeTypeT	NodeT	
ntype		nodeTypeT	

Semantics

State Variables

nt: nodeTypeT

State Invariant

None

Assumptions

None

Access Routine Semantics

new NodeT (*x, y, n*):

- transition: *xc, yc, nt* := *x, y, n*
- output: *out* := *self*
- exception: none

ntype():

- output: *out* := *nt*
- exception: none

Edge Module

Please check EdgeT.java in the tutorial source folder.

Edge Module

Template Module

EdgeT

Uses

PointT, NodeT

Syntax

Exported Types

EdgeT = ?

Exported Constants

TOLERANCE = 1×10^{-5}

Exported Access Programs

Routine name	In	Out	Exceptions
new EdgeT	NodeT, NodeT	EdgeT	
node1		NodeT	
node2		NodeT	
length		real	
is_horizontal		boolean	
equal	EdgeT	boolean	

Semantics

State Variables

$n1$: NodeT

$n2$: NodeT

State Invariant

None

Assumptions

None

Access Routine Semantics

new EdgeT ($nod1, nod2$):

- transition: $n1, n2 := nod1, nod2$
- output: $out := self$
- exception: none

node1():

- output: $out := n1$
- exception: none

node2():

- output: $out := n2$
- exception: none

length():

- output: $out := n1.dist(n2)$
- exception: none

is_horizontal():

- output: $out := |n1.ycoord() - n2.ycoord()| \leq TOLERANCE$
- exception: none

equal(e):

- output:
 $out := (n1.equal(e.node1) \wedge n2.equal(e.node2)) \vee (n1.equal(e.node2) \wedge n2.equal(e.node1))$
- exception: none

Map Module

Please check MapT.java in the tutorial source folder.

Map Module

Module

Map

Uses

EdgeT

Syntax

Exported Access Programs

Routine name	In	Out	Exceptions
init			
add	EdgeT		ALREADY_IN_MAP
del	EdgeT		NOT_IN_MAP
contains	EdgeT	boolean	

Semantics

State Variables

s: set of EdgeT

State Invariant

None

Assumptions

init() is called before any other access program.

Access Routine Semantics

init():

- transition: $s := \{\}$
- exception: none

add(e):

- transition: $s := s \cup e$
- exception: $exc := \exists(f : \text{EdgeT} | f \in s \wedge f.\text{equal}(e)) \Rightarrow \text{ALREADY_IN_MAP}$

del(e):

- transition: $s := s - e$
- exception: $exc := (\neg \exists(f : \text{EdgeT} | f \in s \wedge f.\text{equal}(e))) \Rightarrow \text{NOT_IN_MAP}$

contains(e):

- output: $out := \exists(f : \text{EdgeT} | f \in s \wedge f.\text{equal}(e))$
- exception: none

Map wrapper for ArrayList

- In the assignment, we cannot use ArrayList directly, so we need to wrap it with the specifications of our Map module
- The functions in the Map module will modify the ArrayList which is hidden

Exceptions in Java

- To invoke an exception, you must 'throw' it
- Here is an example from the assignment:

```
public class Map
{
    private static ArrayList<EdgeT> s;

    public static void init()
    {
        s = new ArrayList<EdgeT>();
    }

    public static void add(EdgeT e)
    {
        if (contains(e))
        {
            throw new ALREADY_IN_MAP("The element being added is already in the map");
        }

        s.add(e);
    }
}
```

“Throws” in Java

```
class Demo
{
    static void throwMethod() throws NullPointerException
    {
        System.out.println ("Inside throwMethod");
        throw new NullPointerException ("Demo");
    }
    public static void main(String args[])
    {
        try
        {
            throwMethod();
        }
        catch (NullPointerException exp)
        {
            System.out.println ("The exception get caught" +exp);
        }
    }
}
```

More Java tutorials

<https://www.tutorialspoint.com/java/index.htm>

<https://docs.oracle.com/javase/7/docs/api/>

Common Java exceptions:

https://www.tutorialspoint.com/java/java_builtin_exceptions.htm