COMPSCI 201 Lab 2

Object Oriented Programming in Java

Class Animal

Class Animal

class Animal{

}

Class Animal

```
class Animal{
private String name;
private String habitat;
private int Age;
public void Eat() {}
}
```

Class Animal

```
abstract class Animal{
private String name;
private String habitat;
private int Age;
public abstract void Eat();
}
```

Class Cow

```
class Cow extends Animal{
}
```

Class Animal

```
abstract class Animal{
private String name;
private String habitat;
private int Age;
public void Eat() {}
}
```

Class Animal

```
abstract class Animal{
protected String name;
protected String habitat;
protected int Age;
public void Eat() {}
}
```

Constructors Default constructor

```
class Cow extends Animal{
Cow(){
name="cow";
habitat="grassland";
Age=0;
}
}
```

Parametrized constructor

```
Cow(String name, String habitat ,int Age){
this.name=name;
this.habitat=habitat;
this.Age=Age;
}
```

Copy constructor

```
Cow(Cow otherCow)
{
name=otherCow.name;
habitat=otherCow.habitat;
Age=otherCow.Age;
}
```

Getters

```
public String getName() {return name;}
public String getHabitat() {return habitat;}
public int getAge() {return Age;}
```

Setters

```
public void setName(String name) {this.name=name;}
public void setHabitat(String habitat) {this.habitat=habitat;}
public void setAge(int Age) {this.Age=Age;}
```

Overriding

```
public void Eat() {System.out.println("Eating grass ^^^^^");}
```

Overloading

```
public void Eat() {System.out.println("Eating grass ^^^^^");}
public void Eat(String food) {System.out.println("Eating "+food);}
```

Creating an object

```
Cow basicCow= new Cow();
Cow underwaterCow= new Cow("Daisy","ocean",5);
Cow copyCow=new Cow(basicCow);
```

With and without private/protected

System.out.println(underwaterCow.name);
=> System.out.println(underwater.getName());

Using Setters

```
underwaterCow.setName("Coral");
```

Using methods

BasicCow.Eat();

Output:

Eating grass ^^^^^

Using methods

```
UnderwaterCow. Eat("Fish");
```

Output:

"Eating Fish"

Classes in java

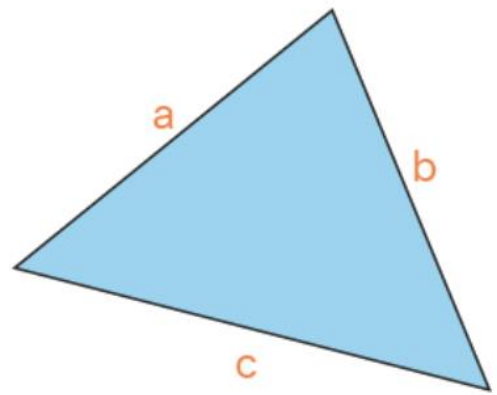
https://docs.oracle.com/javase%2F7%2Fdocs%2Fapi%2F%2F/allclasses-noframe.html

Generic types

```
ArrayList<String> = new ArrayList<String> ();
ArrayList<ArrayList<Integer>> = new ArrayList<ArrayList<Integer>> ();
```

Exercise 1 Class Triangle

- Create a class triangle which has three sides as fields
- Create constructors, getters, and setters
- Create a method for calculating perimeter (s1+s2+s3)
- Create a method to calculate an area of a triangle using Heron's formula:



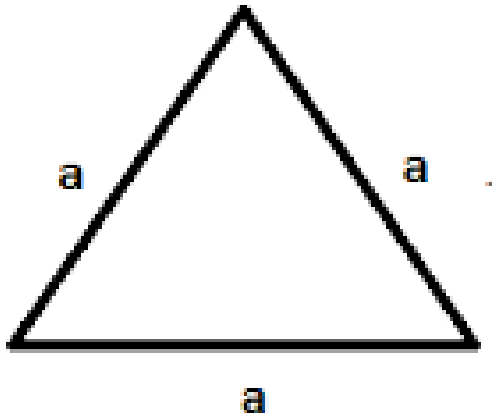
$$s = \frac{a + b + c}{2}$$

s → Semi Perimeter

$$\text{Area of Triangle} = \sqrt{s(s - a)(s - b)(s - c)}$$

Exercise 2 Equilateral triangle

- Create a class equilateral triangle which inherits the class triangle
- Override the method Area form the class triangle to calculate the area using a formula:



$$A = \frac{\sqrt{3}}{4} a^2$$

Class rectangle

- Create a class rectangle