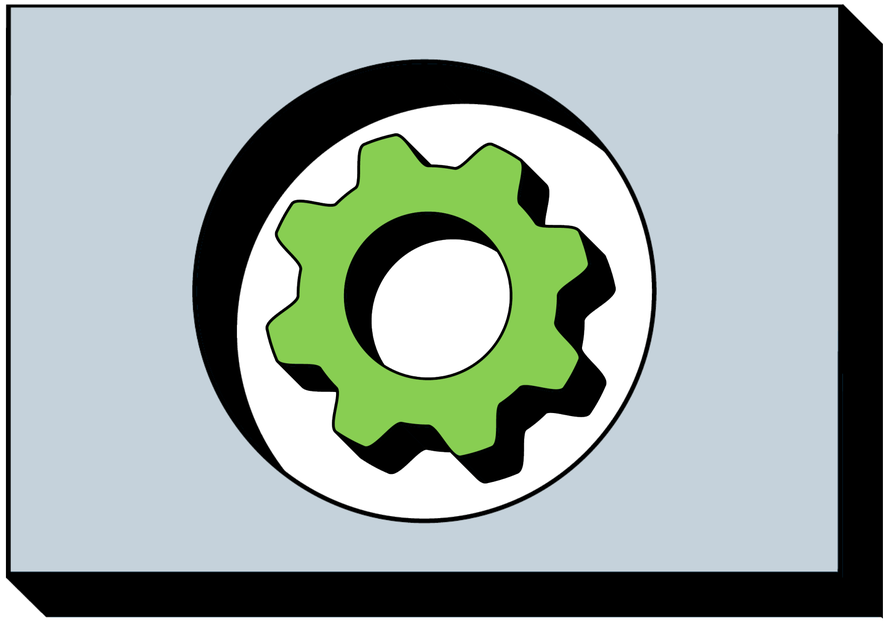
Essential Computing 1

Constructor & special methods



The constructor method creates the object.

```
class Monster
{
    int strength;
    int health;

    Monster( int strength )
    {
        this.strength = strength;
        health = 100;
    }

    {...}
}
```

The **class name** and the **constructor name** must match

```
class Monster
{
    int strength;
    int health;

    Monster( int strength )
    {
        this.strength = strength;
        health = 100;
    }

    {...}
}
```

Notice; no explicit return type (void, int, or such)

```
class Monster
{
    int strength;
    int health;

    Monster( int strength )
    {
        this.strength = strength;
        health = 100;
    }

    {...}
}
```

Constructor arguments that are received when invoked (when instantiating a new monster object)

```
class Monster
{
    int strength;
    int health;

    Monster( int strength )
    {
        this.strength = strength;
        health = 100;
    }

    {...}
}
```

Using **this** to refer to the object, to avoid "variable shadowing".

```
class Monster
{
    int strength;
    int health;

    Monster( int strength )
    {
        this.strength = strength;
        health = 100;
    }

    {...}
}
```

Setting a default health value for all monsters

```
class Monster
{
    int strength;
    int health;

    Monster( int strength )
    {
        this.strength = strength;
        health = 100;
    }

    {...}
}
```

Testing the class

```
public class Main {
    public static void main( String[] args )
    {
        Monster cyclops = new Monster( 10 );
        Monster dragon = new Monster( 30 );

        cyclops.hit( dragon );

        System.out.println( "dragon health: " + dragon.health );
    }
}
```

Overriding toString which exists in all java objects

```
class Monster
{
    {...}

    public String toString(){
        return "Monster. Health: " + health + ", Strength: " + strength;
    }
}
```

Testing toString() using System.out.println()

```
public class Main {
    public static void main( String[] args )
    {
        Monster cyclops = new Monster( 10 );

        System.out.println( cyclops );
    }
}
```

Testing toString() using System.out.println()

```
public class Main {
    public static void main( String[] args )
    {
        Monster cyclops = new Monster( 10 );

        System.out.println( cyclops );
    }
}
```

```
Monster. Health: 100, Strength: 10
```

Overriding equals which exists in all java objects

```
public class Vector2
{
    double x, y;

    public boolean equals( Object that )
    {
        // Self-check.
        if( this == that ) return true;
        // Null check.
        if( that == null ) return false;
        // Type check.
        if( this.getClass() != that.getClass() ) return false;

        Vector2 thatVec = (Vector2) that;
        return this.x == thatVec.x && this.y == thatVec.y;
    }
}
```