**Generating objects: Factory Functions, Constructor Functions**

1. **Factory Functions**

We use Factory Functions when we want to create multiple object instances quickly. A factory function is a function that returns an object and can be reused to make multiple object instances.

Factory functions can also have parameters allowing us to customize the object that gets returned.

**const <functionName> = (para1, para2,…) => {**

**return {**

**para1\_key : para1;**

**<methodName>(){**

//statement with para2;

**}**

**}**

**};**

Then we can call this function to create instances.

const <instance> = <functionName>(arg1, arg2);

1. **Constructor Function with ‘this’ keyword & ‘new’ operator**

This is another way to create object instances, in a more OOP way. We first create a constructor function:

**const <ObjectType> = function (para1, para2,...){**

**this.<property> = para1;**

**this.<method> = function() {**

**//statement with para2, this.<property>,….;**

**}**

**}**

To create instances, we use the **‘new’** keyword along with this constructor function. The **new** keyword will bind the new object instance (newInstance) with the constructor function’s prototype

🡪 References to **this** in constructor function now refers to newInstance

**const <instanceName> = new <ObjectType> (para1, para2, …)**

// var personA = new Person(50,’teacher’);