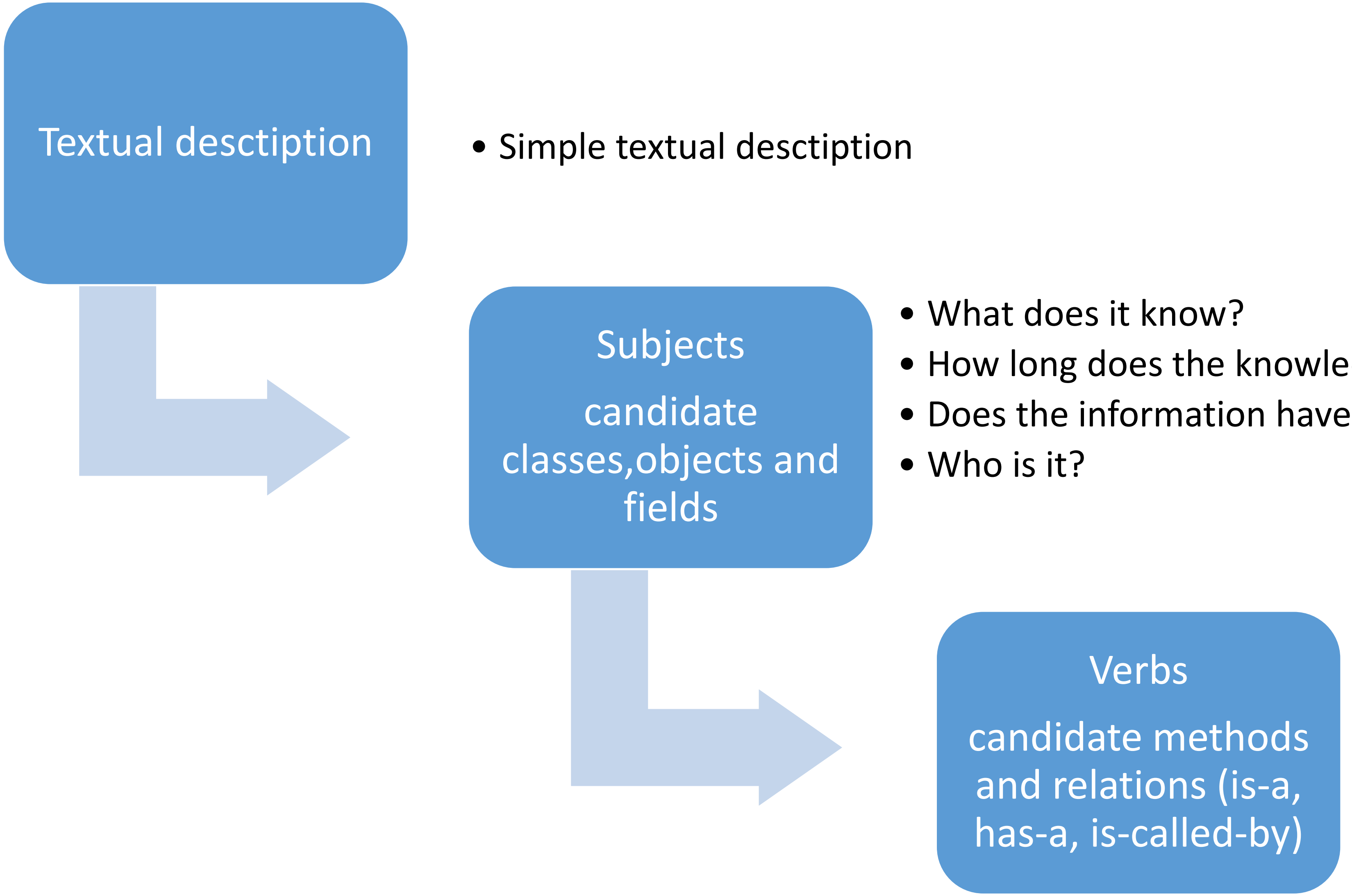
LUOKKA / CLASS

Luokkien löytäminen / Finding classes



Textual desctiption

- Simple textual desctiption

Subjects

candidate classes,objects and fields

- What does it know?
- How long does the knowledge last?
- Does the information have additional properties?
- Who is it?

Verbs

candidate methods and relations (is-a, has-a, is-called-by)

- What does it do?
- Who does it?
- Is there one or several possible actors involved?

Substantiivit / Subjects

- Mikä, kuka
- perusmuoto
- ei synonyymejä
- onko tieto jonkun toisen osa?
- sisältääkö tieto tietoa?
- muutetaanko tai käsitelläänkö tietoa?
- kuka tietoa käyttää?

- what, who
- no duplicates
- is the information part of another?
- does the information include other information?
- is the information handled or changed?
- who is using it?

Mikä se on? / What is it?

- Kokoelma tietoa, jota muut käyttävät ja jonka käsittelyyn tarvitaan käytetään juuri tähän kokoelmaan sopivia toimintoja -> luokka
- Kokoelma tietoa, jota muut käyttävät -> struct
- Tieto, joka on osa toista tietoa -> kenttä
- Lyhytaikanen tieto, elää jonkin toiminnon ajan -> paikallinen muuttuja

- Collection of information that others use through operations particularly connected with this data -> class
- Collection of information that others use –> struct
- Information that is part of other information –> field
- Temporary information, short lived, during one or few operations –> local variable

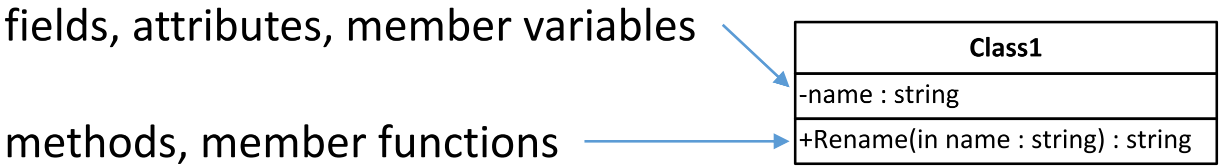
Verbit / Verbs

- Mitä tehdään tai ollaan?
- Mitä tietoja tarvitaan toiminnon suorittamiseen? -> parametrit
- Mitkä tiedot muuttuvat toiminnon aikana tai seurauksena? -> paikalliset ja luokan omat muuttujat
- Mikä tieto on toiminnon tuloksena? -> palautusarvot
- Onko jokin jotain? -> hierarkia
- Sisältääkö joku jotain? ->kooste
- Pyytääkö joku toiminnon suorittamista? -> viestiyhteys

- What is done or what one is?
- What information is needed for the operation to be performed? -> parameters
- Which information is affected or changed during or because of the operation? -> local variables and fields
- What is the result of the operation? -> return value
- Is something something? -> hierarchy
- Does something contain something? -> aggregation
- Does somebody ask for an operation to be performed? -> message passing

Luokka / Class

- compartment model
- used when showing details



fields, attributes, member variables

methods, member functions

Class1

-name : string

+Rename(in name : string) : string

```
class Class1
{
            private string name;

            public string Rename ( string name)
            {
            }
}
```

Class2

- compact form
- used when showing the big picture

```
class Class2
{
}
```

Olion luominen / Creating an object

- Olio luo luodaan vain kerran
- Olio tuhotaan vain kerran
- Yhdestä luokasta voi luoda monta oliota
- Jokaisella oliolla on samat kentät, mutta jokaisen olion kentän arvot omat sen omat (olio omistaa omat kenttänsä)
- Jokainen olio osaa samat asiat
- Kentät ja metodit, joiden edessä on static, omistaa luokka

- Object is created only once
- Object is deleted only once
- There can be several objects created from one class
- Each object has its own values to fields (object owns them)
- Each object can do same things
- Fields and methods marked static are owned by class

Creating and object - constructor

- Same name as class
- No return type nor value
- Called only once during objects lifetime
- Default constructor does not need to be written unless it is overriden
- Parameterized, if there is a parameterized constructor then there should be a particularly written default constructor (override one - override all)

```
class Class1
{
        private string name;

        public string Rename ( string name)
        {
        }

        //default constructor
        public Class1 (){}

        //parameterized constructor
        public Class1 (string name)
        {
                this.name = name.ToUpper();
        }
}
```

Deleting an object

- Destructor is called by garbage collector
- No parameters, only one/class
- Same name as class preceded by ~
- Garbage collector deleted objects that don’t have any references
- The destructor implicitly calls Finalize on the base class of the object

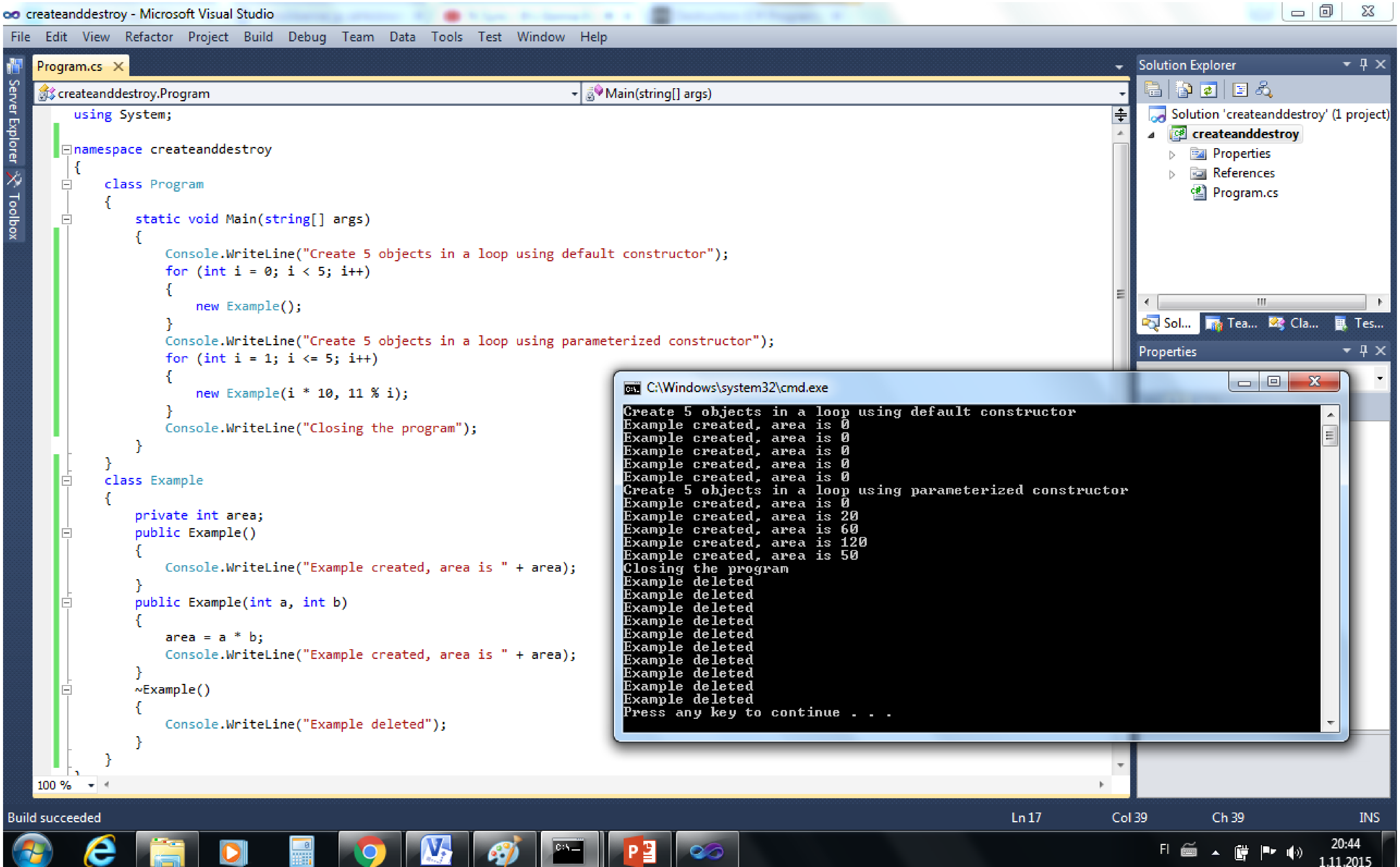
```
class Class1
{
        private string name;

        public string Rename ( string name)
        {
        }

        //default constructor
        public Class1 (){}

        //parameterized constructor
        public Class1 (string name)
        {
                this.name = name.ToUpper();
        }

        //overriden destructor, not called
        ~Class1()
        {
         //cleanup code
        }
}
```



```
using System;

namespace createanddestroy
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Create 5 objects in a loop using default constructor");
            for (int i = 0; i < 5; i++)
            {
                new Example();
            }
            Console.WriteLine("Create 5 objects in a loop using parameterized constructor");
            for (int i = 1; i <= 5; i++)
            {
                new Example(i * 10, 11 % i);
            }
            Console.WriteLine("Closing the program");
        }
    }
    class Example
    {
        private int area;
        public Example()
        {
            Console.WriteLine("Example created, area is " + area);
        }
        public Example(int a, int b)
        {
            area = a * b;
            Console.WriteLine("Example created, area is " + area);
        }
        ~Example()
        {
            Console.WriteLine("Example deleted");
        }
    }
```

```
Create 5 objects in a loop using default constructor
Example created, area is 0
Example created, area is 0
Example created, area is 0
Example created, area is 0
Example created, area is 0
Create 5 objects in a loop using parameterized constructor
Example created, area is 0
Example created, area is 20
Example created, area is 60
Example created, area is 120
Example created, area is 50
Closing the program
Example deleted
Example deleted
Example deleted
Example deleted
Example deleted
Example deleted
Example deleted
Example deleted
Example deleted
Example deleted
Press any key to continue . . .
```