Lifetime

The **lifetime** of a variable is the **time interval** during which the **variable has been allocated a block of memory**

- **Static allocation:**
  - All variables have a lifetime that are the complete execution time (Fortran, Cobol)
  - Dynamic memory management has to be handled by programmers

- **Dynamic allocation:**
  - **Lifetime** is **linked with the scope** of variables
  - Methods to specify lifetime are different in different languages
Example: static in C

#include <stdio.h>

void foo () {

    int i = 10;
    static int si = 10;

    i += 10;
    si += 10;

    printf("i = %3d, si = %3d \n", i, si);
}

int main () {
    int i = 0;

    for (i = 0; i < 10; i++)
        foo();
}
Example: `static` in C

```c
#include <stdio.h>

void foo () {
    int i = 10;
    static int si = 10;

    i += 10;
    si += 10;

    printf("i = %3d, si = %3d \n", i, si);
}

int main () {
    int i = 0;

    for (i = 0; i < 10; i++)
        foo();
}
```

Give a local variable in a function a global lifetime
Example: static in Java

```java
public class ExampleForStaticVariable {

    private static int counter;

    public void increaseCounter () {
        counter++;
    }

    public int getCounter () {
        return counter;
    }

    public static void main (String[] args) {
        ExampleForStaticVariable example1 = new ExampleForStaticVariable();
        ExampleForStaticVariable example2 = new ExampleForStaticVariable();

        example1.increaseCounter();
        example2.increaseCounter();

        System.out.println("example1's counter: "+example1.getCounter());
        System.out.println("example2's counter: "+example2.getCounter());
    }
}
```
Example: static in Java

```java
public class ExampleForStaticVariable {

    private static int counter;

    public void increaseCounter () {
        counter++;
    }

    public int getCounter () {
        return counter;
    }

    public static void main (String[] args) {
        ExampleForStaticVariable example1 = new ExampleForStaticVariable();
        ExampleForStaticVariable example2 = new ExampleForStaticVariable();

        example1.increaseCounter();
        example2.increaseCounter();

        System.out.println("example1's counter: " + example1.getCounter());
        System.out.println("example2's counter: " + example2.getCounter());
    }
}
```

A class field has a global lifetime, or in other words all objects of the class refer to the same binding of a static variable.