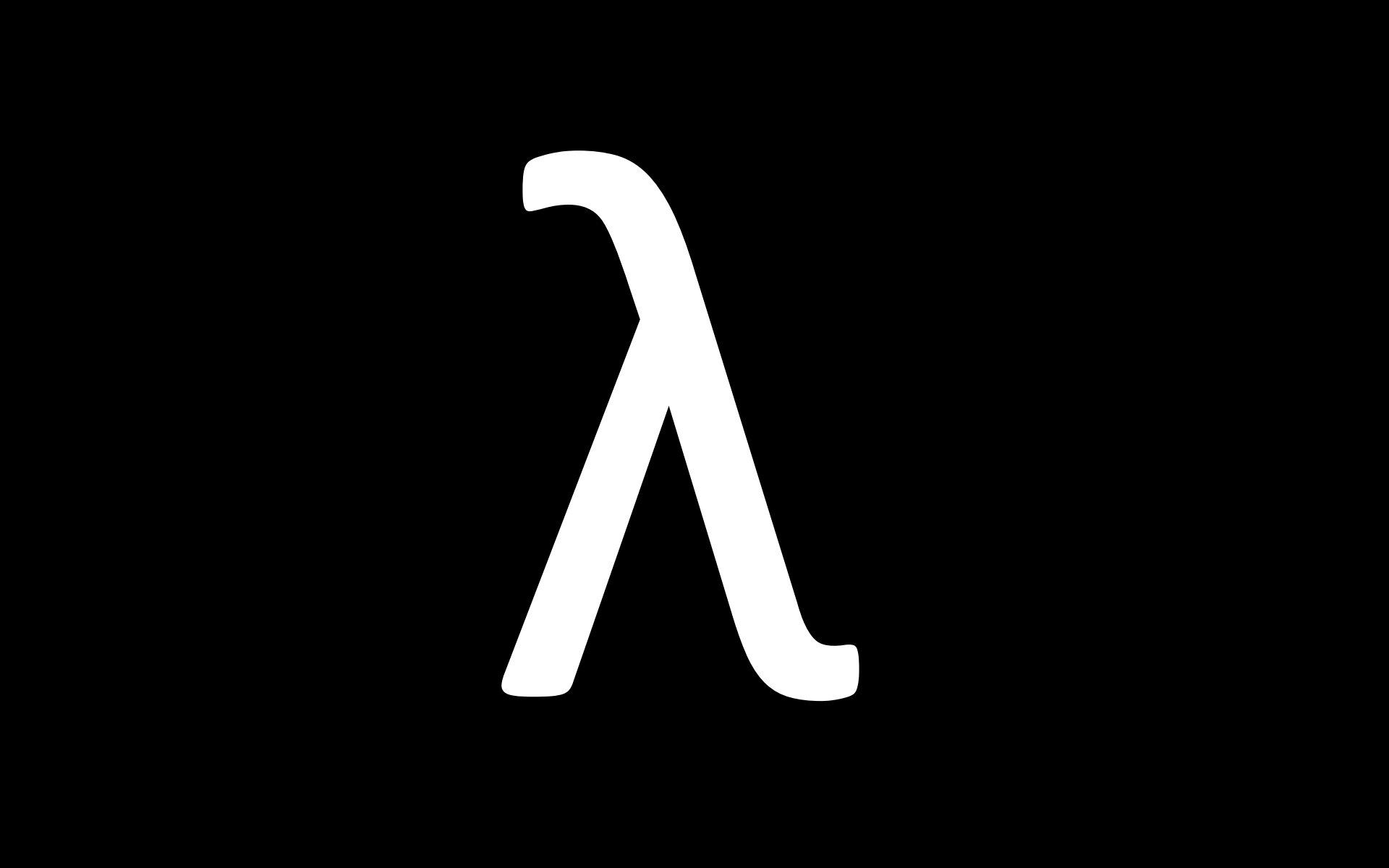


SAPIENZA
UNIVERSITÀ DI ROMA

Programmazione Funzionale e Parallela

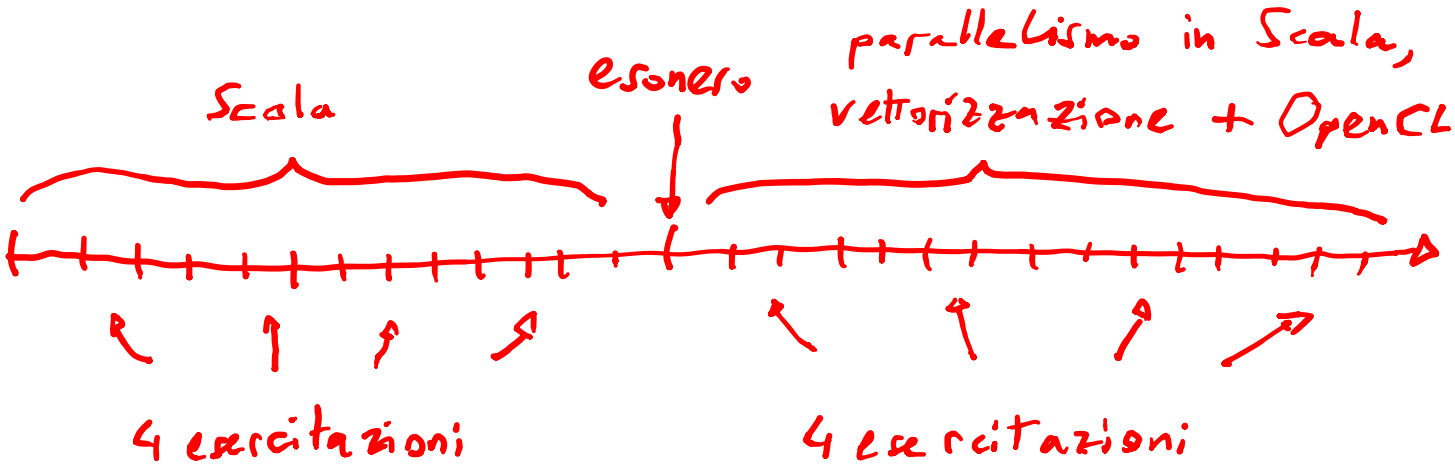
Corso di Laurea in Ingegneria Informatica e Automatica

Sito del corso

https://season-lab.github.io/PFP/

Modalità esame

Necessario registrarsi su form online per seguire le esercitazioni (sezione esami sito corso)



Prova di programmazione in Scala, C, OpenCL

Bonus esercitazioni

Esercitazioni \ Esame	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
18	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26	0,25	0,25	0,25	0,25	0,25	0,25	0,25
19	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26	0,25	0,25	0,25	0,25	0,25	0,25
20	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26	0,25	0,25	0,25	0,25	0,25
21	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26	0,25	0,25	0,25	0,25
22	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26	0,25	0,25	0,25
23	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26	0,25	0,25
24	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26	0,25
25	0,26	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28	0,26
26	0,25	0,26	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36	0,28
27	0,25	0,25	0,26	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57	0,36
28	0,25	0,25	0,25	0,26	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57
29	0,25	0,25	0,25	0,25	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57
30	0,25	0,25	0,25	0,25	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57
31	0,25	0,25	0,25	0,25	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57
32	0,25	0,25	0,25	0,25	0,28	0,36	0,57	1,01	1,63	2,24	2,50	2,24	1,63	1,01	0,57

Cos'è la programmazione funzionale PF?

Paradigma di programmazione incentrato sulle funzioni senza side-effect composte tra loro per ottenere un certo risultato.

Nel seguito vedremo le caratteristiche della programmazione funzionale. Introdurremo il linguaggio Scala che é multi-paradigma e consente oggetti, classi e side-effect oltre ad essere un linguaggio funzionale.

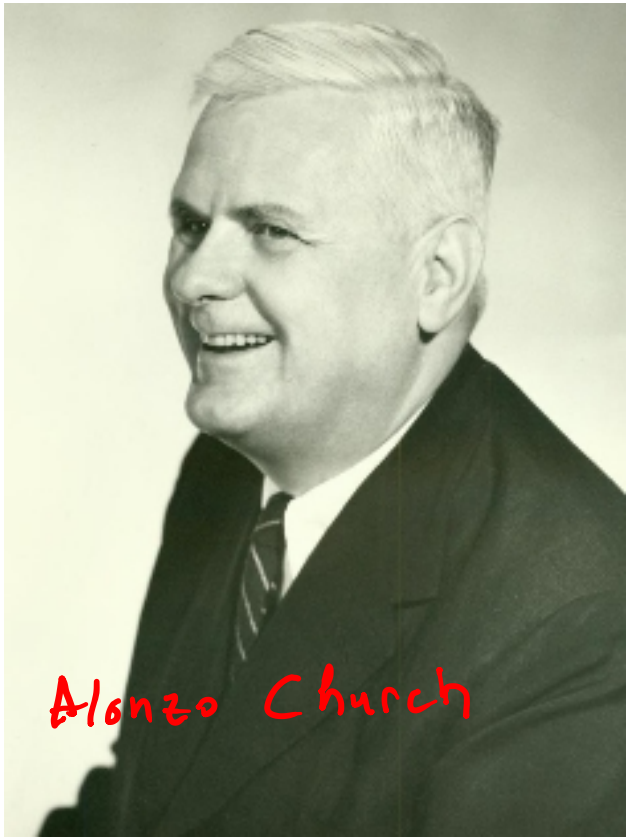
Perché il simbolo lambda è associato alla PF?

λ

Sistema formale introdotto negli anni '30 equivalente a una macchina di Turing.

Es. $\lambda x . x^2$ ⟷ $f(x) = x^2$

non ha un nome — ha un nome



SCOFF
LAMBDA
SCOFF - LAMBDA

Origini della programmazione funzionale

John McCarthy ha sviluppato il linguaggio LISP negli anni '50, introducendo vari elementi della programmazione funzionale ispirati dal λ-calcolo.

Funzioni pure

Funzione pura ← se il risultato dipende solo dall'input
se non ha side-effect al suo interno
se non modifica l'input

funzione pura

Es.
```
int f(int x)
    return 2*x;
}
```

a parità di valori di input restituisce gli stessi valori di output

side effect

```
int y;
int g(int x)
    y++;
    return x+y;
}
```

Trasparenza referenziale

Una funzione gode della proprietà della Trasparenza referenziale se rimpiazzando una invocazione alla funzione con il risultato della chiamata il risultato complessivo non cambia:

$f(2) + 1 \longrightarrow 4 + 1$

$f(2) = 4$

$g(2) + g(2) + 1 \longrightarrow 2 + 2 + 1$

$g(2) = 1 \qquad g(2) = 3$

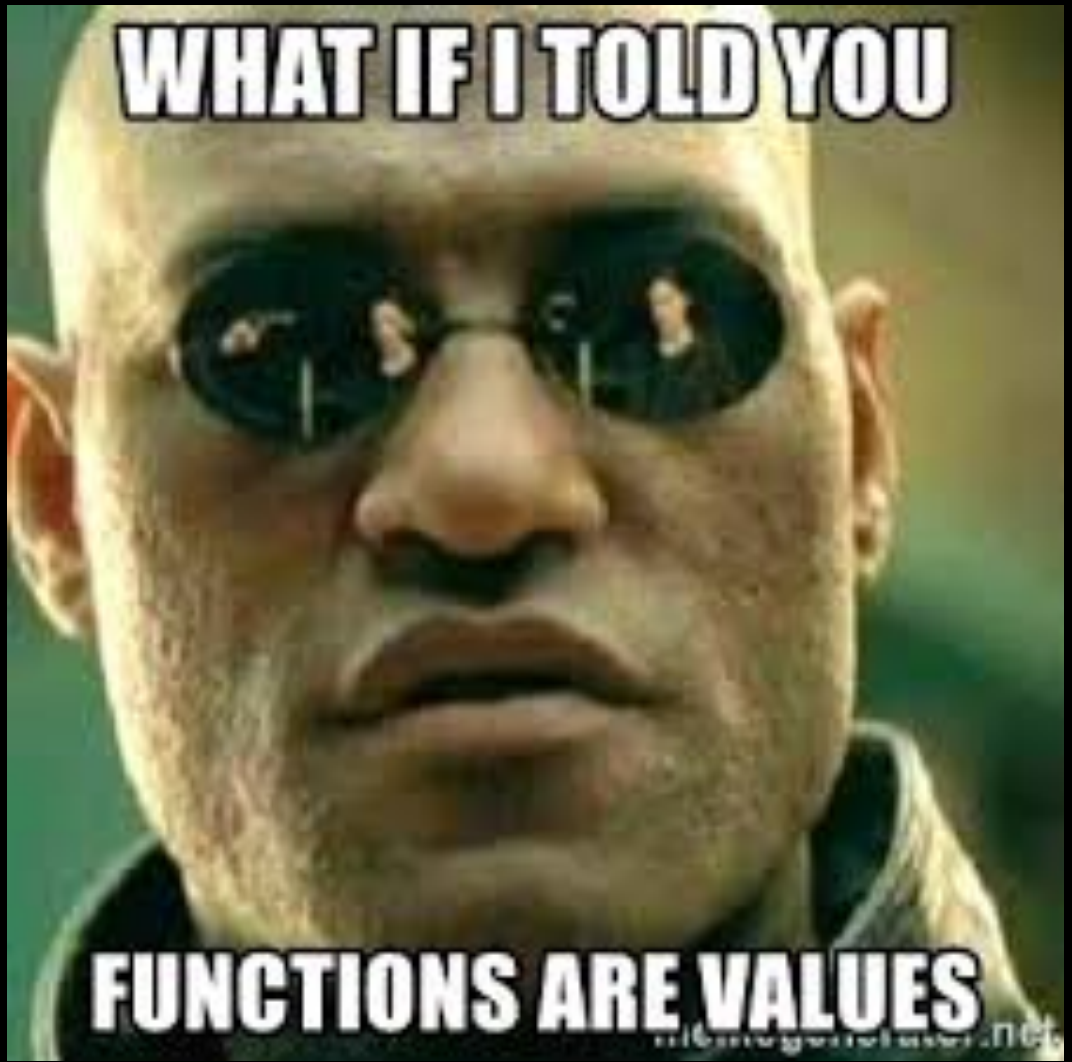
mancanza di trasparenza referenziale

Memoizzazione

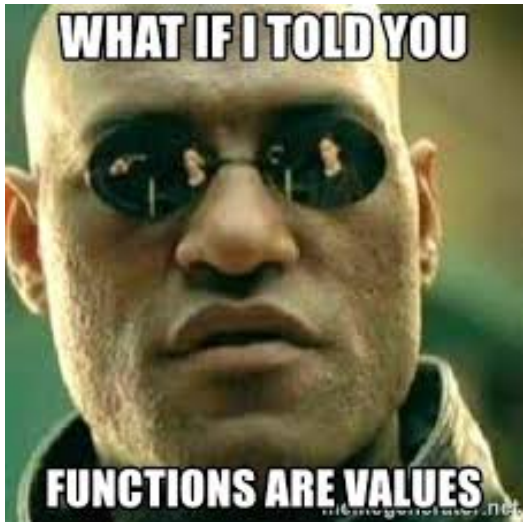
Poiché una funzione pura restituisce lo stesso valore a parità di valori di input, il risultato può essere memoizzato, cioé tenuto in un'apposita cache per evitare di ricalcolarla.

Funzioni di prima classe

Se una funzione può essere trattata come se fosse un valore (passata a una funzione, restituita, tenuta in una variabile ecc.) allora viene chiamata **funzione di prima classe.**


WHAT IF I TOLD YOU
FUNCTIONS ARE VALUES

Altre caratteristiche funzionali di Scala



Vedremo in seguito altre caratteristiche dei linguaggi funzionali supportati da Scala (es. funz. di ordine superiore, laziness, pattern matching, ecc.)

Vantaggi della programmazione funzionale

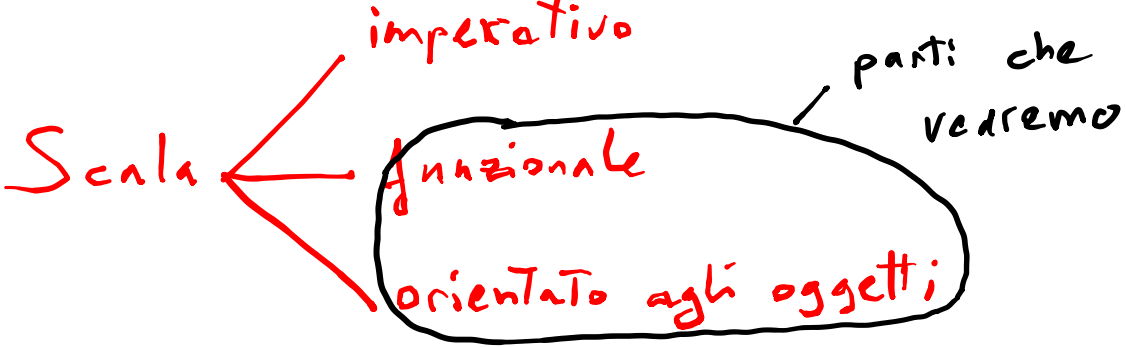
- Modularità: un programma può essere formato da funzioni pure che fungono da moduli
- Componibilità: le funzioni possono essere composte per creare moduli complessi
- Parallelizzabilità: l'assenza di side-effect evita le race condition
- Testabilità: l'assenza di side-effect rende più semplice testare una funzione (es. println("test") v.s. return "test")

Svantaggi della programmazione funzionale

- I/O: intrinsecamente basato su nozione di stato, è complesso da gestire in modo funzionale. Approccio: si isolano le parti del programma che usano l'I/O e si scrive il resto in stile funzionale
- Terminologia: funzioni pure, trasparenza referenziale, ecc.
- Astrazione molto diversa dall'hardware porta penalità prestazionali.

Scala come linguaggio ibrido multi-paradigma

Per trarre vantaggio al meglio dei diversi paradigmi, linguaggi come Scala ne supportano di multipli:



Vediamo ora le caratteristiche del linguaggio Scala.

Vantaggi: compattezza, costrutti espressivi

```
public class Hello {
   public static void main(String[] args){
       System.out.println("Hello");
   }
}
```

Java

```
object Hello extends App {
    println("Hello")
}
```

Scala

In Scala è possibile invocare qualsiasi metodo Java.

Scala gira sulla Java VM!

Svantaggi

Sintassi spesso ostica
riduce leggibilità e
aumenta curva di apprendimento.

A proposito di sintassi ostica…

"Hello world" scritto in brainfuck:

```
++++++++[>++++[>++>+++>+++>+<
<<<-]>+>+>->>+[<]<-]>>.>---
.+++++++..+++.>>.<-.<.+++.------.----
----.>>+.>++.
```