Introduktion til Programmering og Problemløsning (PoP)

Jon Sporring
Department of Computer Science
2021/09/30

UNIVERSITY OF COPENHAGEN

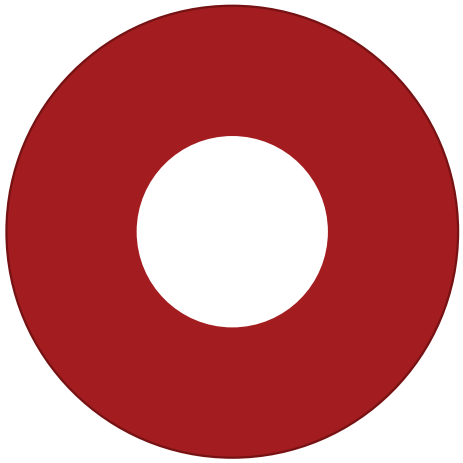
Hvor langt er I kommet med materialet?

https://tinyurl.com/2dt64n6w

Hvad er typen

https://tinyurl.com/h8j9ary8

Kodegenbrug



Hvad er arealet af en annulusen med ydre og indre diameter på 7 cm og 3 cm?

Opgaveneddeling:
1. Areal af en cirkel
2. Areal af en annulus

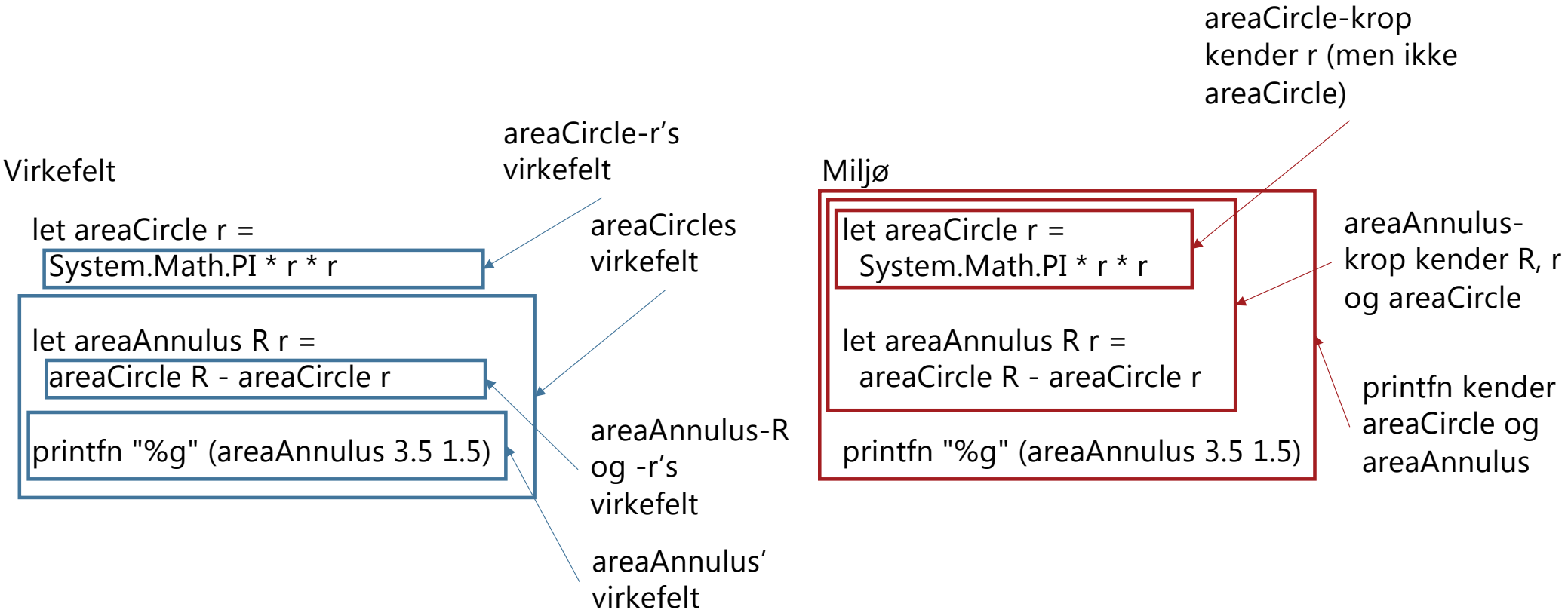
~~let areaCircle r =~~
~~printfn "%g" (System.Math.PI * r * r)~~

```
let areaAnnulus R r =
  printfn "%g" (areaCircle R - areaCircle r)

printfn "%g" (areaAnnulus 3.5 1.5)
```

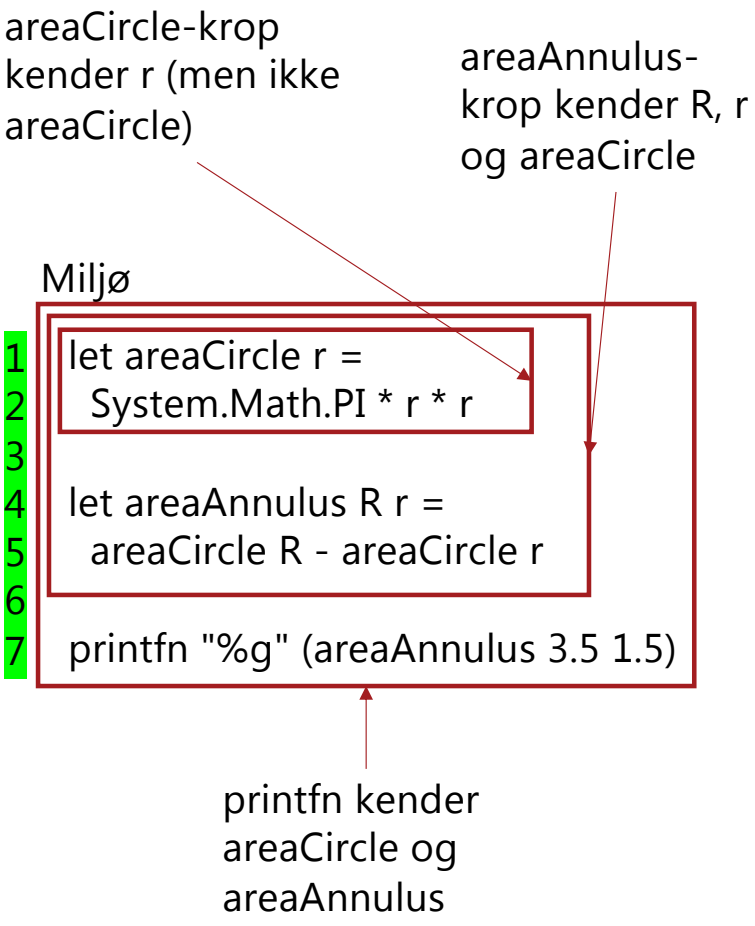
https://tinyurl.com/2wt4dy7p

```
let areaCircle r =
  System.Math.PI * r * r

let areaAnnulus R r =
  areaCircle R - areaCircle r

printfn "%g" (areaAnnulus 3.5 1.5)
```

Virkefelter vs. Miljø (scope vs. environment)



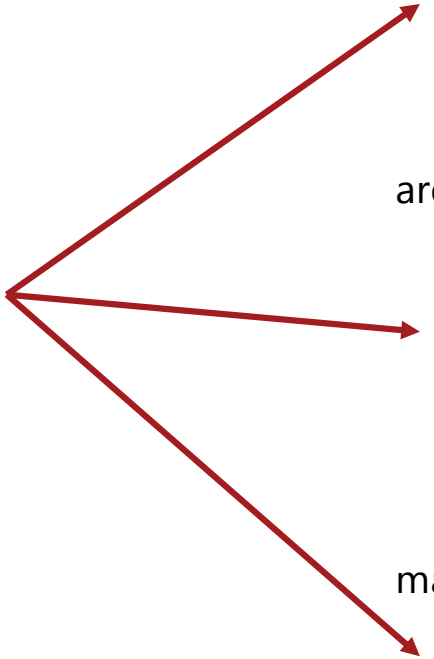
Håndkøring



Step	Linje	Miljø	Binding
1	1	E_0	areaCircle = ((r), System.Math.PI * r * r, ())
2	4	E_0	areaAnnulus = ((R,r), areaCircle R - areaCircle r, (areaCircle))
3	7	E_0	printfn "%g" (areaAnnulus 3.5 1.5) = ?
4	7	E_0	areaAnnulus 3.5 1.5 = ?
5	4	E_1	((R=3.5,r=1.5), areaCircle R - areaCircle r, (areaCircle))
6	5	E_1	areaCircle 3.5 - areaCircle 1.5 = ?
7	5	E_1	areaCircle 3.5 = ?
8	1	E_2	((r=3.5), System.Math.PI * r * r, ())
9	2	E_2	return = 38.4 ⋯
10	5	E_1	areaCircle 3.5 = 38.4⋯
11	5	E_1	areaCircle 1.5 = ?
12	1	E_3	((r=1.5),krop,())
13	2	E_3	return = 7.0 ⋯
14	5	E_1	areaCircle 1.5 = 7.0⋯
15	5	E_1	areaCircle 3.5 - areaCircle 1.5 = 31.4
16	5	E_1	return = 31.4⋯
17	7	E_0	areaAnnulus 3.5 1.5 = 31.4⋯
18	7	E_0	output = 31.4⋯
19	7	E_0	printfn "%g" (areaAnnulus 3.5 1.5) = ()
20	7	E_0	return = ()

Interface, implementation og applikation

```
let areaCircle r =
  System.Math.PI * r * r

let areaAnnulus R r =
  areaCircle R - areaCircle r

printfn "%g" (areaAnnulus 3.5 1.5)
```



area.fsi

```
module area

val areaCircle : float -> float
val areaAnnulus : float -> float -> float
```

area.fs

```
module area

let areaCircle r = System.Math.PI * r * r
let areaAnnulus R r = areaCircle R - areaCircle r
```

main.fsx

```
printfn "%g" (area.areaAnnulus 3.5 1.5)
```

Dokumentationsstandarden

area.fsi

```
module area

/// <summary>Calculate the area of a circle.</summary>
/// <remarks>Radius is assumed to be non-negative.</remarks>
/// <example>
/// The following code:
/// <code>
///   let r = 1.5
///   let a = areaCircle r
///   printfn "areaCircle %.1f = %.1f" r a
/// </code>
/// prints <c>areaCircle 1.5 = 7.1</c>.
/// </example>
/// <param name="r">Radius of the circle.</param>
/// <returns>The area of the circle.</returns>
val areaCircle : r:float -> float

/// Calculate the area of an annulus with outer and inner
/// radius R and r.
val areaAnnulus : R:float -> r:float -> float
```

area.fs

```
module area

let areaCircle r = System.Math.PI * r * r

// We assume that R > r.
// Note to self: add error handling in the future
let areaAnnulus R r = areaCircle R - areaCircle r
```

Diskuter med din nabo:

1. Hvilke værdier for areaCircle og areaAnnulus vil være nyttige at teste i en blackbox afprøvning?
2. Hvilke 'units' vil I vælge til en whitebox afprøvning?

Advarsel: biblioteks- og applikationsfil skal have forskellige navne!

area.fs

```
module area

let areaCircle r = System.Math.PI * r * r
let areaAnnulus R r = areaCircle R - areaCircle r
```

~~area.fsx~~

```
printfn "%g" (area.areaAnnulus 3.5 1.5)
```

areaTest.fsx

```
printfn "%g" (area.areaAnnulus 3.5 1.5)
```

Spørgsmål