| | |
|---|---|
| **Iniziato** | venerdì, 3 luglio 2020, 10:09 |
| **Stato** | Completato |
| **Terminato** | venerdì, 3 luglio 2020, 10:59 |
| **Tempo impiegato** | 49 min. 59 secondi |

Domanda **1**

Completo

Punteggio max.: 1,00

Consider the following function:

```
def mistery(l:List[Int],f:Int=>Boolean)=
  ((l filter f) foldLeft 0) (_ + _)
```

and the following expression:

```
mistery(List(5,2,7,9,21,15), ( (x) => (5<x) && (x<16) ))
```

What is the result of the evaluation of the above expression?
Justify your answer.

The result of the evaluation is: 31.

What the function does for first, is applying to the input list l the filter f. The filter f filters the elements that respect that condition, that basically means "filter the element from the list which are greater than 5 an less than 16 strictly. The elements that respect the condition thus are: 7, 9, 15. After that the foldLeft is applied. The computation of the foldLeft proceeds as follow:

from the list (7, 9, 15) the first computation executed is

- (0 + 7) = 7 (0 is the starting element provided and 7 is the left most element from the list.

- (7 + 9) = 16

- (16 + 15) = 31.

Domanda **2**

Completo

Punteggio max.: 1,00

Describe by words and by means of an example the groupBy higher-order function.

[redacted]

Domanda **3**

Completo

Punteggio max.: 1,00

Are the following declarations correct or not?

val x1: List[String] = List[Nothing]()
val x2: Array[String] = Array[Nothing]()

Justify your answer.

[redacted]

Domanda **4**

Completo

Punteggio max.: 1,00

Consider the following excerpt of code:

```
class A (val x:Int) {
  def get = x+1
}

class B (k:Int) extends A(k) {
  override def get = x-1
}

val z:A = new B(5)

println (z.get)
```

What is printed? Justify your answer.

[redacted]

Domanda **5**

Completo

Punteggio max.: 1,00

What does the following function compute?

```
def mistery2(l:List[Int]) = {
  def f(l:List[Int]):List[Int]=
    if (l.isEmpty) List(0)
    else (l takeWhile (_ == l.head)).length ::
      f(l dropWhile (_ == l.head))

  f(l).max
}
```

Justify your answer.

This function computes what is the maximum length of the longest sequence of consecutive equals numbers within the input List. Namely, if the input List is List(6, 3, 3, 3, 3, 2, 2) the function returns 4 which is the length of the longest consecutive sequence (which is (3, 3, 3, 3)).

What basically this function does is taking the an element from the list and "consider them" (with takeWhile) until these elements are equal to the head and then take the length of the elements that respect this condition. After that the lenght is stored into a list by concatenation and all these elements are dropped whith dropWhile. Then f is called recursively until the list is empty. So we will have in our case a list like this List(1, 4, 2, 0) from which we take the max, which is 4.

◄ Exam - module 2 LAAI - 11.6.2020

Vai a...