

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

and the following expression:

```
mystery(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 and 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.

The groupBy higher-order function groups a list of elements based on a specific condition. For example, if we have a list of integers, we can group them based on whether they are even or odd. The result of the groupBy function is a Map where the keys are the conditions and the values are the corresponding lists of elements. For example, if we have a list of integers [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] and we use the groupBy function with a condition that checks if a number is even, the result will be a Map where the key is true (for even numbers) and the value is a list of even numbers [2, 4, 6, 8, 10], and the key is false (for odd numbers) and the value is a list of odd numbers [1, 3, 5, 7, 9].

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.

~~The declaration of x1 is not correct because it is not possible to create a List of Nothing. The declaration of x2 is not correct because it is not possible to create an Array of Nothing.~~

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.

~~It prints 5 because the class B extends class A and the get method of class B is not defined, so it inherits the get method of class A.~~

Domanda **5**

Completo

Punteggio max.:
1,00

What does the following function compute?

```
def mystery2(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 LAAl - 11.6.2020](#)