

1. Program jest zgodny z podaną specyfikacją.
2. Program się kompiluje bez błędów i ostrzeżeń.
3. Został przetestowany.
4. Student go rozumie i **potrafi wyjaśnić przy „tablicy”**.

## Paradygmaty programowania - ćwiczenia

### Lista 1

Wszystkie funkcje należy napisać w obu językach: OCaml i Scala. W specyfikacjach będzie na ogół wykorzystywana notacja OCaml, ponieważ jest prostsza.

1. Zdefiniuj funkcję `flatten1` : 'a list list -> 'a list, która dla argumentu będącego listą list tworzy listę, złożoną z elementów wszystkich podlist z zachowaniem ich kolejności, np. `flatten1 [[5;6];[1;2;3]]` zwraca `[5; 6; 1; 2; 3]`, czyli spłaszcza listę o jeden poziom.  
Scala `flatten1[A] (xss: List[List[A]]): List[A]`
2. Zdefiniuj funkcję `count` : 'a \* 'a list -> int obliczającą ile razy dany element występuje w danej liście, np. `count ('a', ['a'; 'l'; 'a'])` zwraca 2.  
Scala `count[A] (x: A, xs: List[A]): Int`
3. Zdefiniuj funkcję `replicate`: 'a \* int -> 'a list powtarzającą dany obiekt określoną liczbę razy i zwracającą wynik w postaci listy, np. `replicate ("la",3)` zwraca `["la"; "la"; "la"]`.  
Scala `replicate[A] (x: A, n: Int): List[A]`
4. Zdefiniuj funkcję `sqrList` : int list -> int list podnoszącą do kwadratu wszystkie elementy danej listy liczb, np. `sqrList [1;2;3;-4]` zwraca `[1; 4; 9; 16]`.  
Scala `sqrList(xs: List[Int]): List[Int]` (metoda)  
i `sqrList: List[Int] => List[Int]` (funkcja)
5. Zdefiniuj funkcję `palindrome` : 'a list -> bool sprawdzającą, czy dana lista jest palindromem, tj. równa się sobie samej przy odwróconej kolejności elementów, np. `palindrome ['a'; 'l'; 'a']` zwraca true.  
Scala `palindrome[A] (xs: List[A]): Boolean`
6. Zdefiniuj swoją funkcję `listLength` : 'a list -> int, obliczającą długość dowolnej listy (oczywiście bez użycia standardowej funkcji `List.length`).  
Scala `listLength[A](xs: List[A]): Int`
7. Rozwiąż układ równań rekurencyjnych (zakładając, że  $N$  jest potęgą dwójki):  
$$T(1) = 1$$
$$T(N) = c(\lg N) + T(N/2) \quad \text{dla } N \geq 2$$

Wykorzystaj technikę zilustrowaną na wykładzie 1 (Dodatek: Złożoność obliczeniowa. Podstawowe pojęcia).