**Paradygmaty programowania IT, inż. II rok**

|  |  |  |
| --- | --- | --- |
| Imię i nazwisko | Data | Godzina |
| Daniel Konsewicz | 03.11.2021 | 7:30 |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| nr listy:  3  zrobione | v | ocaml | v | v |  |  |  |  |  |  |
| nr listy:  4  zrobione | v | v |  |  |  |  |  |  |  |  |

Lista 3

Zad.1

1. let f1 x = x 2 2

x musi być funkcją przyjmującą dwa Int’y i zwracającą „a”, czyli x: int -> int -> 'a

**f1 : (int -> int -> 'a) -> 'a = <fun>**

1. let f2 x y z = x ( y^z)

y oraz z muszą być Stringami, ponieważ między nimi znajduje się znak scalania ^, natomiast x to funkcja przyjmująca stringa, czyli x: String -> ‘a

**f2 : x y z = x ( y ^ z );;**

Zad.4

1. W sytuacji, gdy pierwszy element listy jest największy, wtedy lista nie będzie się dzieliła i zapętli się w nieskończoność
2. Przez zły znak listy dzielą się na większe i mniejsze od x, przez co równe zostają odrzucone i lista pomijać będzie liczby występujące kilka razy i zapisze je tylko raz

Lista 4

Zad. 1

1. let f1 x y z = x y z;;

x musi być funkcją przyjmującą dwa argumenty dowolnego typu i zwracającą również nieznany nam typ, więc x: a’ -> b’ -> c’

**f1 : (a’ -> b’ -> c’) -> a’ -> b’ -> c’ = <fun>**

1. let f2 x y = function z -> x::y;;

funkcja przyjmuje dwa argumenty, gdzie x jest dowolnego typu, natomiast y to lista tego samego typu co x, równo się to funkcji zwracającej listę tego samego typu co przekazane argumenty

**f2 : 'a -> 'a list -> 'b -> 'a list = <fun>**