**Московский государственный технический**

**университет им. Н.Э. Баумана.**

Факультет «Информатика и управление»

Кафедра «Системы обработки информации и управления»

Курс «Основы информатики»

Отчет по лабораторной работе №3

«трек курса «функциональное программирование»»

|  |  |  |
| --- | --- | --- |
| Выполнил: |  | Проверил: |
| студент группы ИУ5-34 |  | преподаватель каф. ИУ5 |
| Кравцов А.Н. |  | Ю.Е. Гапанюк. |
| Подпись и дата: |  | Подпись и дата: |

Москва, 2020 г.

1. Разработайте функцию, которая принимает три параметра обобщенных типов и возвращает их в виде кортежа. Модифицируйте функцию: не указывая явно типы параметров, задавая выражения в теле функции, сделайте так, чтобы параметры были типов int, float, string.

2. С использованием двухэтапного создания обобщенных функций реализуйте функции, которые осуществляют сложение:

* трех аргументов типа int;
* трех аргументов типа float;
* трех аргументов типа string.

3. С использованием list comprehension для четных элементов списка [1..10] верните список кортежей. Каждый кортеж содержит элемент списка, его квадрат и куб.

4. Напишите два варианта функции, которая принимает на вход список и возвращает квадраты его значений. Необходимо использовать свойства списка Head и Tail. Первый вариант функции использует оператор if, второй вариант использует сопоставление с образцом на уровне функции.

5. Последовательно примените к списку функции map, sort, filter, fold, zip, функции агрегирования. Функции применяются в любом порядке и произвольно используются в трех комбинациях.

* Первая комбинация заканчивается функцией агрегирования (например, сумма элементов списка). Cписок предварительно может быть отсортирован, отфильтрован и т.д.
* Вторая комбинация заканчивается функцией fold, которая осуществляет свертку списка. Вторая комбинация выполняет те же действия, что и первая комбинация и должна возвращать такой же результат.
* Третья комбинация заканчивается функцией zip, которая соединяет два списка.

6. Реализуйте предыдущий пункт с использованием оператора потока « |> ».

7. Реализуйте предыдущий пункт с использованием оператора композиции функций « >> ».

open System

//1

let Fn1()=

let Func (a, b, c) = (a,b,c)

let Func\_int (a, b, c) = (a+1,b+1,c+1)

let Func\_float (a, b, c) = (a+1.0,b+1.0,c+1.0)

let Func\_string (a, b, c) = (a.ToString(),b.ToString(),c.ToString())

let q1 = Func(1,2,3)

let q2 = Func(1,2,3)

let q3 = Func(1,2,3)

let q4 = Func(1,2,3)

printfn "%A" q1

printfn "%A" q2

printfn "%A" q3

printfn "%A" q4

//2

let Fn2()=

let Func\_gen(a,b,c, func)= func(a,b,c)

let sum\_int(a,b,c)=Func\_gen(a,b,c,fun(a,b,c)->a+b+c)

let sum\_float(a,b,c)=Func\_gen(a,b,c,fun(a,b,c)->a+b+c+0.0)

let sum\_string(a,b,c)=Func\_gen(a,b,c,fun(a,b,c)->a+b+c+"")

let q1 = sum\_int(1,2,3)

let q2 = sum\_float(1.0,2.0,3.0)

let q3 = sum\_string("a","b","c")

printfn "%A" q1

printfn "%A" q2

printfn "%A" q3

//3 list comprehension

let Fn3()=

let list = [ for a in 1 .. 10 do if a%2=0 then yield (a) ,(a \* a),(a \* a \* a) ]

printfn "The list: %A" list

//4\_1

let Fn4()=

let rec List\_fn1(lst:int list):int list=

if lst.IsEmpty then []

else (lst.Head\*lst.Head)::List\_fn1(lst.Tail)

let list2 = List\_fn1([1..3])

printfn "The list: %A" list2

//4\_2

let rec List\_fn2 = function

|[]->[]

|x::xs -> x\*x::List\_fn2(xs)

let list3 = List\_fn1([1..3])

printfn "The list: %A" list3

//5 Последовательно примените к списку функции map, sort, filter, fold, zip,

//функции агрегирования.

//5\_1

let Fn5()=

let list4\_1 = (List.sum(List.sort((List.map (fun x->x+2+1)(List.filter(fun x->x%2=0) ([2;3;5;6;7;2;3;5;7;8;10;11;13]))))))

printfn "The list: %A" list4\_1

//5\_2

let list4\_2 = (List.fold(fun acc x->acc+x+1) 0 (List.sort((List.map (fun x->x+2)(List.filter(fun x->x%2=0) ([2;3;5;6;7;2;3;5;7;8;10;11;13]))))))

printfn "The list: %A" list4\_2

//5\_3

let list4\_3 = List.zip ([2;3;5;6;7;2;3;5;7]) (List.sort((List.map (fun x->x+2)([2;3;5;6;7;2;3;5;7]))))

printfn "The list: %A" list4\_3

//6

let Fn6()=

//6\_1

let list5\_1 = [2;3;5;6;7;2;3;5;7;8;10;11;13] |> List.filter(fun x->x%2=0) |> List.map (fun x->x+2+1) |> List.sort |> List.sum

printfn "The list: %A" list5\_1

//6\_2

let list5\_2 = [2;3;5;6;7;2;3;5;7;8;10;11;13] |> List.filter(fun x->x%2=0) |> List.map (fun x->x+2) |> List.sort |> List.fold(fun acc x->acc+x+1) 0

printfn "The list: %A" list5\_2

//6\_3

let list5\_3 = [2;3;5;6;7;2;3;5;7] |> List.map (fun x->x+2) |> List.sort |> List.zip [2;3;5;6;7;2;3;5;7]

printfn "The list: %A" list5\_3

//7

//7\_1

let Fn7()=

let Func\_7\_1 = List.filter(fun x->x%2=0) >> List.map (fun x->x+2+1) >> List.sort >> List.sum

let list6\_1 = Func\_7\_1 [2;3;5;6;7;2;3;5;7;8;10;11;13]

printfn "The list: %A" list6\_1

//7\_2

let Func\_7\_2 = List.filter(fun x->x%2=0) >> List.map (fun x->x+2) >> List.sort >> List.fold(fun acc x->acc+x+1) 0

let list6\_2 = Func\_7\_1 [2;3;5;6;7;2;3;5;7;8;10;11;13]

printfn "The list: %A" list6\_2

//7\_3

let Func\_7\_3 = List.map (fun x->x+2) >> List.sort >> List.zip [2;3;5;6;7;2;3;5;7]

let list6\_3 = Func\_7\_3 [2;3;5;6;7;2;3;5;7]

printfn "The list: %A" list6\_3

let menu()=

let mutable x = 0

let mutable work = true

while work do

printfn "==================== MENU ====================

1 - Задание 1

2 - Задание 2

3 - С использованием list comprehension для четных элементов вернуть элемент, квадрат и куб в виде списка

4 - Задание 4

5 - 3 комбинации map, sort, filter, fold, zip, функции агрегирования

6 - Реализуйте предыдущий пункт с использованием оператора потока « |> »

7 - Реализуйте предыдущий пункт с использованием оператора композиции функций « >> »

=============================================="

x <- int32(System.Console.ReadLine())

match x with

| 1 -> Fn1()

| 2 -> Fn2()

| 3 -> Fn3()

| 4 -> Fn4()

| 5 -> Fn5()

| 6 -> Fn6()

| 7 -> Fn7()

| 8 -> work<-false

menu()

