Защищено:
Гапанюк Ю.Е.

"__"_____2020 г.

Демонстрация:
Гапанюк Ю.Е.

"__"_____2020 г.

Отчет по лабораторной работе № 5
по курсу
Базовые компоненты интернет-технологий

ИСПОЛНИТЕЛЬ:

студент группы ИУ5Ц-52Б _____
(подпись)

Яровенко М. В.

"__"_____2020 г.

Москва, МГТУ - 2020

СОДЕРЖАНИЕ ОТЧЕТА

1. Задание

1. Создайте два варианта функции, которая возвращает кортеж значений. Первый вариант принимает на вход параметры в виде кортежа, второй вариант параметры в каррированном виде.

2. Выберите простой алгоритм, который может быть реализован в виде рекурсивной функции и реализуйте его в F#. Пример – вычисление суммы целых чисел в заданном диапазоне.

3. Преобразуйте разработанную рекурсивную функцию в форму хвостовой рекурсии.

4. Разработайте конечный автомат из трех состояний и реализуйте его в виде взаимно-рекурсивных функций.

5. Разработайте функцию, которая принимает 3 целых числа и лямбда-выражение для их суммирования в виде кортежа и в каррированном виде.

2. Листинг программы

```
// Яровенко Максим, ИУ5Ц-52Б
open System


let FuncKort (a:int, b:int, c:int) =
  let s = a+b+c
  let p = a*b*c
  (s, p)

let FuncCar (a:int)(b:int)(c:int) =
  let s = a+b+c
  let p = a*b*c
  (s, p)

let rec Factorial(n:int):int =
  if n<=1 then 1
  else n*Factorial(n-1)

let rec FactorialXV(n:int, acc:int):int =
  if n=1 then acc
```

```
    else FactorialXV(n-1, n*acc)

let rec FactorialX n = FactorialXV(n,1)

let rec State1(x:int) =
    printfn "%i - (+1) %i" x (x+1)
    let x_next = x+1
    if x_next>5 then State2(x_next)
    else State1(x_next)

and State2(x:int) =
    printfn "%i - (+10) %i" x (x+10)
    let x_next = x+1
    if x_next>10 then State3(x_next)
    else State2(x_next)

and State3(x:int) =
    printfn "%i - (+100) %i" x (x+100)
    let x_next = x+1
    if x_next<=15 then State3(x_next)

let sum (a:int, b:int, c:int, func1: int*int*int->int) = func1 (a, b, c)

let sumK (a:int, b:int, c:int, func1: int->int->int->int) = func1 a b c




[<EntryPoint>]
let main argv =
    let resultKor = FuncKort(2, 4, 3)
    let resultCar = FuncCar(2)(4)(3)
    printfn "%s" ("Результаты функции, реализованной через кортеж и в
каррированном виде, соответсвенно: " + resultKor.ToString() + " и " +
resultCar.ToString())

    let resRec = Factorial(7)
    printfn "%s" ("\nРезультат рекурсивной функции - вычисление
факториала (7): " + resRec.ToString())

    let resRecX = FactorialX(5)
    printfn "%s" ("\nРезультат хвостовой рекурсивной функции -
вычисление факториала (5): " + resRecX.ToString())
```

```
printfn "%s" ("\nПример автомата из трех состояний:")
State1(1)


let primer1 = sum(5, 7, 2, fun(a, b, c)->a+b+c)
let primer2 = sumK(5, 7, 2, fun a b c -> a*b*c)
printfn "%s" ("\nПример результата работы функции, которая принимает 3 целых числа и лямбда-выражение для их суммирования в виде кортежа и в каррированном виде соответсвенно: " + primer1.ToString() + " и " + primer2.ToString())

0
```

3. Результаты работы программы