**Отчет по домашнему заданию по F#**

3

(количество листов)

Студент группы ИУ5-32:

Платонов Артемий

Подпись Дата 06.12.2017

Руководитель: Гапанюк Ю.Е

Подпись

Дата

Москва, 2017

***Задание:***

На основе рассмотренного примера составить программу на функциональном языке программирования для решения биквадратного уравнения с использованием алгоритма рассмотренного в разделе «Биквадратное уравнение» статьи https://ru.wikipedia.org/wiki/Уравнение\_четвёртой\_степени. Программа должна использовать алгебраические типы и механизм сопоставления с образцом. В случае комплексных корней их вычисление не обязательно, можно выводить информацию о том, что корни комплексные.

***Текст программы:***

open System

type SquareRootResult =

| NoRoots

| OneRoot of double

| TwoRoots of double\*double

| ThreeRoots of double\*double\*double

| FourRoots of double \* double\*double \* double

| NotBE

///Функция вычисления корней уравнения

let CalculateRoots(a:double, b:double, c:double):SquareRootResult =

if a = 0.0 then NotBE

else

let D = b\*b - 4.0\*a\*c;

if D < 0.0 then NoRoots

else if D = 0.0 then

let trt = -b / (2.0 \* a)

if trt<0.0 then NoRoots

else if trt=0.0 then OneRoot 0.0

else

let rt11=Math.Sqrt(trt)

let rt12= -rt11

TwoRoots(rt11, rt12)

else

let sqrtD = Math.Sqrt(D)

let trt1 = (-b - sqrtD) / (2.0 \* a);

let trt2 = (-b + sqrtD) / (2.0 \* a);

if trt2<0.0 && trt1<0.0 then NoRoots

else if trt2=0.0 && trt1<0.0 || trt1=0.0 && trt2<0.0 then OneRoot 0.0

else if trt1=0.0 && trt2>0.0 then

let rt201 = Math.Sqrt(trt2)

let rt202 = -rt201

ThreeRoots(0.0, rt201, rt202)

else if trt2=0.0 && trt1>0.0 then

let rt201 = Math.Sqrt(trt1)

let rt202 = -rt201

ThreeRoots(0.0, rt201, rt202)

else if trt1>0.0 && trt2>0.0 then

let rt211 = Math.Sqrt(trt1)

let rt212 = -rt211

let rt221 = Math.Sqrt(trt2)

let rt222 = -rt221

FourRoots(rt211, rt212, rt221, rt222)

else if trt2>0.0 then

let rt1 = Math.Sqrt(trt2)

let rt2 = -rt1

TwoRoots (rt1,rt2)

else

let rt1 = Math.Sqrt(trt1)

let rt2 = -rt1

TwoRoots (rt1,rt2)

///Вывод корней (тип unit - аналог void)

let PrintRoots(a:double, b:double, c:double):unit =

printf "Коэффициенты: a=%A, b=%A, c=%A. " a b c

let root = CalculateRoots(a,b,c)

//Оператор сопоставления с образцом

let textResult =

match root with

| NoRoots -> "Действительных корней нет"

| OneRoot(rt) -> "Один корень " + rt.ToString()

| TwoRoots(rt1,rt2) -> "Два корня " + rt1.ToString() + " и " + rt2.ToString()

| ThreeRoots(rt1,rt2,rt3) -> "Три корня " + rt1.ToString() + ", " + rt2.ToString() + " и " + rt3.ToString()

| FourRoots(rt1,rt2,rt3,rt4) -> "Четыре корня " + rt1.ToString() + ", " + rt2.ToString() + ", " + rt3.ToString() + " и " + rt4.ToString()

| NotBE-> "Уравнение не является биквадратным"

printfn "%s" textResult

[<EntryPoint>]

let main argv =

//4 корная

let a1 = 4.0;

let b1 = -5.0;

let c1 = 1.0;

//3 корня

let a2 = -1.0;

let b2 = 4.0;

let c2 = 0.0;

//2 корня

let a3 = 1.0;

let b3 = 2.0;

let c3 = -8.0;

//1 корень

let a4 = 1.0;

let b4 = 0.0;

let c4 = 0.0;

//нет корней

let a5 = 1.0;

let b5 = 0.0;

let c5 = 4.0;

//не является биквадратным

let a6 = 0.0;

let b6 = 1.0;

let c6 = -1.0;

PrintRoots(a1,b1,c1)

PrintRoots(a2,b2,c2)

PrintRoots(a3,b3,c3)

PrintRoots(a4,b4,c4)

PrintRoots(a5,b5,c5)

PrintRoots(a6,b6,c6)

//|> ignore - перенаправление потока с игнорирование результата вычисления

Console.ReadLine() |> ignore

0 // возвращение целочисленного кода выхода

