

Учреждение образования

«Белорусский государственный университете  
информатики и радиоэлектроники»

Кафедра программного обеспечения информационных технологий

Отчёт  
по лабораторной работе №2  
«Основы разработки на языке функционального программирования»  
Вариант №9

Выполнил: Левко Сергей Владимирович  
магистрант кафедры программного  
обеспечения информационных  
технологий группа №757041

Проверил: Парамонов Антон Иванович  
кандидат технических наук, доцент

Минск 2019

## СОДЕРЖАНИЕ

1	Задание.....	3
2	Решение задачи .....	4
3	Вывод.....	7

## 1 Задание

Задание данной лабораторной работы разделено на 3 части.

В первой части необходимо реализовать программное средство на функциональном языке программирования LISP, Haskell или F#, позволяющее вычислить математическое выражение, представленное в формуле (1).

$$\left| \ln(\sqrt{2 + 0.2n + 0.4m}) + \sin(\sqrt{n^2 + m^2}) - \cos(e^{\cos(n+\pi)}) \right| \quad (1)$$

Во второй части необходимо реализовать программное средство на функциональном языке программирования LISP, Haskell или F#, позволяющее вычислить математическое выражение, представленное в формуле (2).

$$Y(x) = \begin{cases} a^x, & x < 0, \\ tg(x), & 1 \leq x \leq 3 \end{cases} \quad (2)$$

В третьей части необходимо реализовать программное средство на функциональном языке программирования LISP, Haskell или F#, которое меняет местами первый и последний элементы исходного списка.

## 2 Решение задачи

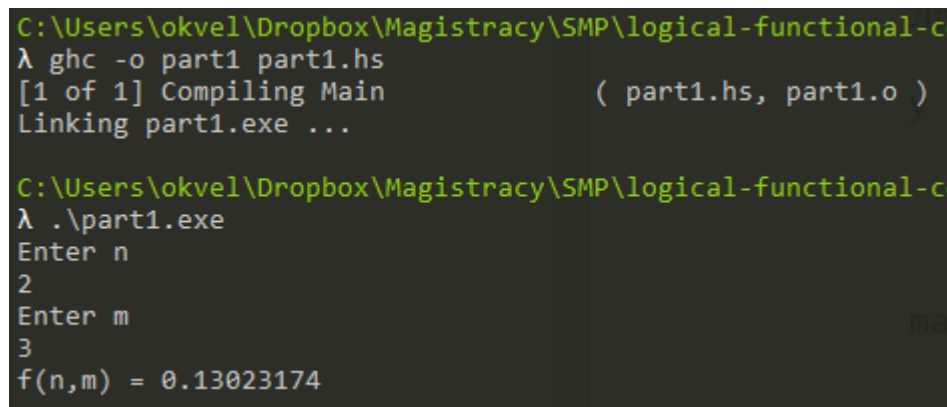
Для решения поставленных задач был выбран язык функционального программирования Haskell. Ниже представлен листинг первого, второго и третьего задания.

Листинг первого задания.

```
f :: Float -> Float -> Float
f n m = abs (log (sqrt (2 + 0.2**n + 0.4**m)) + sin (sqrt
(n^2 + m^2)) - cos (exp (cos (n + pi))))

main = do
    putStrLn("Enter n")
    n <- getLine
    putStrLn("Enter m")
    m <- getLine
    putStr("f(n,m) = ")
    print (f (read n) (read m))
```

На рисунке 1 изображены результаты работы программы.



```
C:\Users\okvel\Dropbox\Magistracy\SMP\logical-functional-c
λ ghc -o part1 part1.hs
[1 of 1] Compiling Main                ( part1.hs, part1.o )
Linking part1.exe ...

C:\Users\okvel\Dropbox\Magistracy\SMP\logical-functional-c
λ .\part1.exe
Enter n
2
Enter m
3
f(n,m) = 0.13023174
```

Рисунок 1 – Результаты выполнения программы

Листинг второго задания.

```
y x a
| x < 0 = a**x
| (1 <= x) && (x <= 3) = tan x
| otherwise = 0

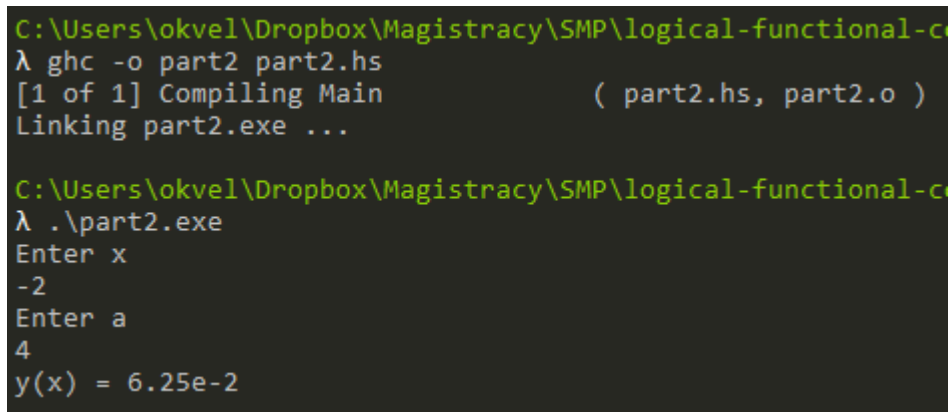
main = do
    putStrLn("Enter x")
    x <- getLine
    putStrLn("Enter a")
    a <- getLine
```

```

putStr("y(x) = ")
-- x = -2, a = 4 -> y = 0.0625
-- x = 1 -> y = 1.5574
print (y (read x) (read a))

```

На рисунке 2 изображены результаты работы программы.



```

C:\Users\okvel\Dropbox\Magistracy\SMP\logical-functional-c
λ ghc -o part2 part2.hs
[1 of 1] Compiling Main                ( part2.hs, part2.o )
Linking part2.exe ...

C:\Users\okvel\Dropbox\Magistracy\SMP\logical-functional-c
λ .\part2.exe
Enter x
-2
Enter a
4
y(x) = 6.25e-2

```

Рисунок 2 – Результаты выполнения второго задания.

Листинг третьего задания.

```

main = do
    let test = ("$", (2, 3), 4, 5, "a", ("e", "r"), "g",
"%", "v", (10, "b"))
    putStr("Initial tuple:  ")
    print(test)
    putStr("Tuple after swap: ")
    print (swap test)

frst :: (a,b,c,d,e,f,g,h,i,j) -> a
frst (x,_,_,_,_,_,_,_,_,_) = x
scnd :: (a,b,c,d,e,f,g,h,i,j) -> b
scnd (_,x,_,_,_,_,_,_,_,_) = x
thrd :: (a,b,c,d,e,f,g,h,i,j) -> c
thrd (_,_,x,_,_,_,_,_,_,_) = x
frth :: (a,b,c,d,e,f,g,h,i,j) -> d
frth (_,_,_,x,_,_,_,_,_,_) = x
ffth :: (a,b,c,d,e,f,g,h,i,j) -> e
ffth (_,_,_,_,x,_,_,_,_,_) = x
sxth :: (a,b,c,d,e,f,g,h,i,j) -> f
sxth (_,_,_,_,_,x,_,_,_,_) = x
snth :: (a,b,c,d,e,f,g,h,i,j) -> g
snth (_,_,_,_,_,_,x,_,_,_) = x
eigh :: (a,b,c,d,e,f,g,h,i,j) -> h
eigh (_,_,_,_,_,_,_,x,_,_) = x

```

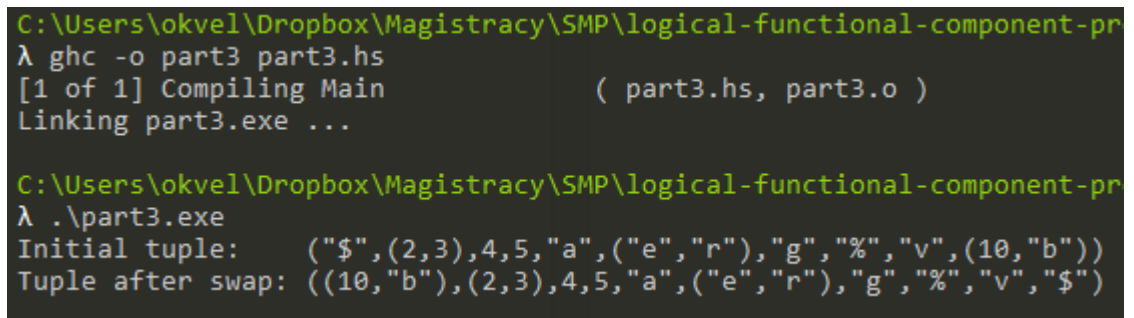
```

nnth :: (a,b,c,d,e,f,g,h,i,j) -> i
nnth (_,_,_,_,_,_,_,_,_,x,_) = x
lst  :: (a,b,c,d,e,f,g,h,i,j) -> j
lst  (_,_,_,_,_,_,_,_,_,x) = x

swap :: (a,b,c,d,e,f,g,h,i,j) -> (j,b,c,d,e,f,g,h,i,a)
swap a = (lst a,scnd a,thrd a,frth a,ffth a,sxth a,snth
a,eigh a,nnth a,frst a)

```

На рисунке 3 изображены результаты выполнения программы.



```

C:\Users\okvel\Dropbox\Magistracy\SMP\logical-functional-component-pr
λ ghc -o part3 part3.hs
[1 of 1] Compiling Main                ( part3.hs, part3.o )
Linking part3.exe ...

C:\Users\okvel\Dropbox\Magistracy\SMP\logical-functional-component-pr
λ .\part3.exe
Initial tuple:  ("$(, (2,3), 4, 5, "a", ("e", "r"), "g", "%", "v", (10, "b"))
Tuple after swap: ((10, "b"), (2,3), 4, 5, "a", ("e", "r"), "g", "%", "v", "$")

```

Рисунок 3 – Результаты выполнения третьего задания.

### **3 Вывод**

В ходе выполнения данной лабораторной работы были изучены основы функционального языка программирования Haskell и создано программное средство, позволяющее решать поставленные в лабораторной работе задачи.