**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИИ**  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
**«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ**  
**ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»**  
**(БГТУ им. В.Г.Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №6

Функциональное и логическое программирование

Выполнил: ст. группы ПВ-21  
Ковалев Павел Александрович

Проверил: Поляков В.К.

**Белгород 2020**

**Вариант 8**

**Задания**

1.Напишите следующие программы:

1)Программа, считывающая два числа и возвращающая их сумму.

2)Программа, распечатывающая переданные в нее аргументы командной строки.

3)Программа, которая принимает в команднной строке имя файла и распечатывает его на экране.

4)Программа, принимающая в команндной строке число n и имя файла и выводящая на экран первые n строк файла.

2.Реализуйте программы, выполняющие задания вашего варианта из первой лабораторой работы. Параметры функций должны считываться с клавиатуры.

**Код программы**

1.

1)

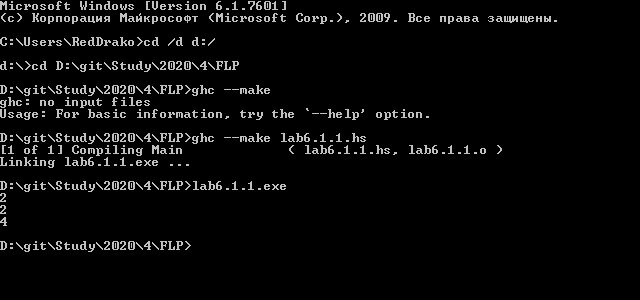
import System.IO

main = do

a <- readLn

b <- readLn

print (a + b)

return (a + b)

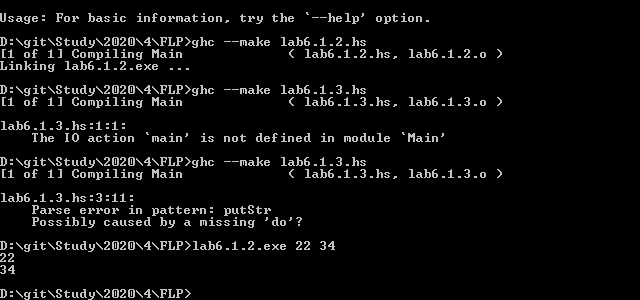
2)

import System.Environment

import Data.List

main = do

args <- getArgs

print args

3)

import System.IO

import System.Environment

main = do

name <- getArgs

fromHandle <- openFile name ReadMode

contents <- hGetContents fromHandle

hPutStr stdout contents

4)

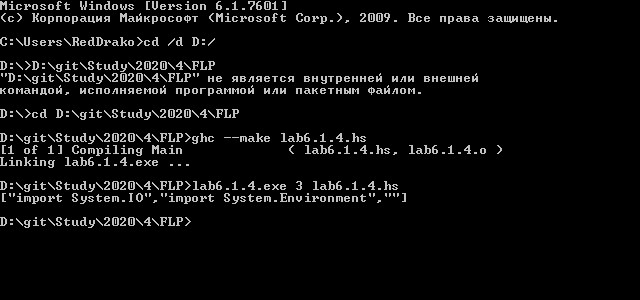
import System.IO

import System.Environment

main = do

[n, path] <- getArgs

content <- readFile path

print (take (read n) (lines content))

2.

import System.IO

import System.Environment

import Data.Char

import Data.Maybe

import Data.List

main = do

putStrLn "Enter an x1,y1: "

line <- getLine

let x1 = read line::Double

line <- getLine

let y1 = read line::Double

putStrLn "Enter an x2,y2: "

line <- getLine

let x2 = read line::Double

line <- getLine

let y2 = read line::Double

putStrLn "Enter an x3,y3: "

line <- getLine

let x3 = read line::Double

line <- getLine

let y3 = read line::Double

print (isRectangular (x1,y1) (x2,y2) (x3,y3)) >> main

isRectangular ::(Double,Double) -> (Double,Double) -> (Double,Double) -> Bool

isRectangular (a,b) (c,d) (e,f) = if (c-a)\*(e-a)+(d-b)\*(f-b)==0 || (a-c)\*(e-c)+(b-d)\*(f-d)==0 || (a-e)\*(c-e)+(b-f)\*(d-f)==0 then True else False

