Домашнє завдання № 0

Мова Haskell. Робота з iнтерпретатором ghci

Варіант № 12

**Мета роботи**

Ознайомитись з основними типами мови. Ознайомитись зi структурою та функцiями Glasgow Haskell Compiller. Набути навичок роботи з iнтерпретатором ghci та визначення найпростiших функцiй.

**Завдання**

1. Наведiть приклади виразiв вказаного типу. Кожен список має мiстити кiлька елементiв. Перегляньте тип прикладiв, як їх визначає ghci. Прокоментуйте.

2. Визначте два варiанти вказаних далi функцiй. Перший варiант – з одним аргументом-кортежем, другий – без використання кортежiв чи спискiв, див. ”Неформальний всуп.pdf”, стор.14.

1.12. [[(Bool,String)],[Double])]

2.12 Функцiя визначає, чи двi точки, заданi координатами, знаходяться у

одному квадрантi.

1.14 [([Double],(Bool,Char),Integer)]

2.14 Функцiя за довжиною чотирьох вiдрiзкiв визначає, чи можна на них

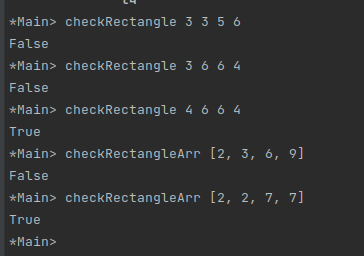
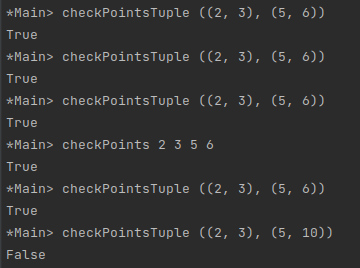
побудувати прямокутник.

**Код програми**

module Main where  
  
  
main :: IO ()  
main = do  
 putStrLn "hello world"  
  
  
  
-- Variant 12 ----------------------------------------------------------------------------------------------------------  
  
{-  
[[(Bool,String)],[Double]]  
[[(True, "kek")], [3.6]]  
-}  
  
  
checkPoints :: Float -> Float -> Float -> Float -> Bool  
checkPoints x1 y1 x2 y2 =  
 x2 - x1 == y2 - y1  
  
checkPointsTuple :: ((Float, Float), (Float, Float)) -> Bool  
checkPointsTuple tuple = do  
 let fstPoint = fst tuple  
 let sndPoint = snd tuple  
 fst sndPoint - fst fstPoint == snd sndPoint - snd fstPoint

-- Variant 14 ----------------------------------------------------------------------------------------------------------  
  
  
{-  
[([Double],(Bool,Char),Integer)]  
[([5.6], (True, "m"), 4)]  
-}  
  
  
checkRectangle :: Float -> Float -> Float -> Float -> Bool  
checkRectangle side1 side2 side3 side4 =  
 side1 == side2 && side3 == side4 || side1 == side3 && side2 == side4 || side1 == side4 && side3 == side2  
  
  
checkRectangleArr :: [Float] -> Bool  
checkRectangleArr arr  
 | null arr = error "No values!"  
 | length arr /= 4 = error "Must be transmitted 4 values"  
 | otherwise = arr !! 0 == arr !! 1 && arr !! 2 == arr !! 3 || arr !! 0 == arr !! 2 && arr !! 1 == arr !! 3 || arr !! 0 == arr !! 3 && arr !! 2 == arr !! 1

**Приклад роботи**



**Висновок**: під час виконання цієї лабораторної работи було опрацьовано матеріал про типи мови Hskell. А також було отримано знання про структуру та функцiї Glasgow Haskell Compiller, набуто навички роботи

iнтерпретатором ghci та визначенні найпростiших функцiй. В результаті було виконано варіанти №12 та №14 завдань цієї лабораторної роботи.