Круть Катерина
ТІ-01
Домашня робота №0

Домашнє завдання № 0
Мова Haskell. Робота з інтерпретатором ghci
Варіант № 12

Мета роботи

Ознайомитись з основними типами мови. Ознайомитись зі структурою та функціями Glasgow Haskell Compiller. Набути навичок роботи з інтерпретатором ghci та визначення найпростіших функцій.

Завдання

1. Наведіть приклади виразів вказаного типу. Кожен список має містити кілька елементів. Перегляньте тип прикладів, як їх визначає ghci. Прокоментуйте.
2. Визначте два варіанти вказаних далі функцій. Перший варіант – з одним аргументом-кортежем, другий – без використання кортежів чи списків, див. ”Неформальний всуп.pdf”, стор.14.

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

2.12 Функція визначає, чи дві точки, задані координатами, знаходяться у одному квадранті.

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

2.14 Функція за довжиною чотирьох відрізків визначає, чи можна на них побудувати прямокутник.

Круть Катерина
ТІ-01
Домашня робота №0

Код програми

```haskell
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)]
-}
```

Круть Катерина
ТІ-01
Домашня робота №0

```haskell
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
```

Круть Катерина
ТІ-01
Домашня робота №0

Приклад роботи

```
*Main> checkPointsTuple ((2, 3), (5, 6))
True
*Main> checkPointsTuple ((2, 3), (5, 6))
True
*Main> checkPointsTuple ((2, 3), (5, 6))
True
*Main> checkPoints 2 3 5 6
True
*Main> checkPointsTuple ((2, 3), (5, 6))
True
*Main> checkPointsTuple ((2, 3), (5, 10))
False
```

```
*Main> checkRectangle 3 3 5 6
False
*Main> checkRectangle 3 6 6 4
False
*Main> checkRectangle 4 6 6 4
True
*Main> checkRectangleArr [2, 3, 6, 9]
False
*Main> checkRectangleArr [2, 2, 7, 7]
True
*Main>
```

Висновок: під час виконання цієї лабораторної работи було опрацьовано матеріал про типи мови Hskell. А також було отримано знання про структуру та функції Glasgow Haskell Compiller, набуто навички роботи інтерпретатором ghci та визначенні найпростіших функцій. В результаті було виконано варіанти №12 та №14 завдань цієї лабораторної роботи.