

Haskell – wprowadzenie

zadania

Zadanie 1

Napisać funkcję dwuargumentową, która przyjmuje liczby x , y oraz zwraca wartość

$$\frac{x^2 + y^2}{2xy}$$

(w szczególności dobrać działającą sygnaturę – uwaga na typy, których oczekują operatory działań).

Wykorzystując napisaną definicję funkcji oraz ustalenie wartości na jednym z argumentów zdefiniować funkcję przyjmującą liczbę x oraz zwracającą

$$\frac{x^2 + 1}{2x}.$$

Wykorzystując te funkcje obliczyć odpowiednio $\frac{3^2+4^2}{2 \cdot 3 \cdot 4}$ oraz $\frac{5^2+1}{2 \cdot 5}$.

Zadanie 2

Wykorzystując funkcję

```
sumaWartosci :: (Int -> Int) -> (Int -> Int) -> Int ->  
  Int -> Int
```

```
sumaWartosci f g x y = (f x) + (g y)
```

zdefiniować funkcję, która przyjmuje liczby x , y oraz zwraca wartość $2x + 3y$ (ponownie, poprzez ustalenie wartości! tym razem ustalamy funkcję, nie liczby).

Wykorzystać zdefiniowaną funkcję do obliczenia wartości $2 \cdot 1000 + 3 \cdot 100$.

Zadanie 3

Zapisać funkcję

```
ocena :: Double -> String
ocena 2.0 = "niezaliczone"
ocena 5.0 = "brawo!"
ocena x = "wpisane masz " ++ show x
```

stosując dozory (guards) zamiast dopasowania do wzorca.

Zadanie 4

Liczby Stirlinga I rodzaju zdefiniowane są następującymi zależnościami:

$$\begin{cases} s(0,0) = 1, \\ s(n,0) = 0, \\ s(n,n) = 1, \\ s(n,k) = (n-1)s(n-1,k) + s(n-1,k-1), \end{cases} \quad \begin{array}{ll} \text{dla } n \geq 1 \\ \text{dla } n \geq 1 \\ \text{dla } n \geq 1, n \geq k, \\ k \geq 1 \end{array}$$

Napisać funkcję obliczającą $s(n, k)$.

Zadanie 4

Napisać stosując rekurencję funkcję o sygnaturze

```
iloczynListy :: [Integer] -> Integer
```

która przyjmuje listę liczb naturalnych i zwraca jej iloczyn.

Wykorzystać tę funkcję do obliczenia 1000!.

Zadanie 5

Napisać funkcję

```
merge :: [Int] -> [Int] -> [Int]
```

łączącą dwie posortowane niemalejąco listy w jedną posortowaną niemalejąco listę. Używając tej funkcji napisać funkcję

```
mergeSort :: [Int] -> [Int]
```

sortującą niemalejąco listę za pomocą algorytmu merge sort. Pomocna może być funkcja `length` zwracająca długość listy.

Zadanie 6

Napisać funkcję

`czyDoskonala :: Int -> Bool`

sprawdzającą, czy podana liczba jest doskonała (liczba naturalna jest doskonała, jeśli jest sumą wszystkich swoich dzielników mniejszych od samej liczby, np. 6 jest liczbą doskonałą, ponieważ $6 = 1 + 2 + 3$, a dowolna liczba pierwsza nie jest liczbą doskonałą).

Zadanie należy zrobić nie wykorzystując funkcji Haskella wykraczających poza te przedstawione na pierwszych zajęciach.

Wskazówka: Napisać pomocnicze funkcje, który zostaną wykorzystane, aby zrealizować funkcję z zadania, w szczególności funkcję `dzielniki :: (Int, [Int]) -> [Int]`, która wywołana na liczbie n oraz liście złożonej z samej jedynki zwróci listę wszystkich dzielników n .