Haskell – wprowadzenie

zadania

Zadanie 1

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

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

(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

$$\frac{25 + 10y}{y^2}.$$

Wykorzystując te funkcje obliczyć odpowiednio $\frac{2^2+2\cdot 2\cdot 3}{3^2}$ oraz $\frac{25+10\cdot 2}{4}$.

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 $4x + 3y$ (ponownie, poprzez ustalenie wartości! tym razem ustalamy funkcje, nie liczby).
Wykorzystać zdefiniowaną funkcję do obliczenia wartości $4 \cdot 1000 + 3 \cdot 100$.

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:

$$\begin{cases} s(0,0) = 1, & \\ s(n,0) = 0, & \text{dla} \quad n \geqslant 1 \\ s(n,n) = 1, & \text{dla} \quad n \geqslant 1 \\ s(n,k) = (n-1)s(n-1,k) + s(n-1,k-1), & \text{dla} \quad n \geqslant 1, n \geqslant k, \\ & \qquad\quad k \geqslant 1 \end{cases}$$

Napisać funkcję obliczającą $s(n,k)$.

Zadanie 4

Napisać stosując rekurencję funkcję o sygnaturze

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

która przyjmuje listę liczba 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*.