

Языки программирования (осень 2018)

В начало ► Мои курсы ► ЯП 2018 ► Оценка лабораторных работ ► Лабораторная работа №3 ► Работа

Лабораторная работа №3

Моя работа

Инструкции для работы ▼

Для того, чтобы отправить работу на оценку, нажмите "Начало подготовки Вашей работы".

На открывшейся странице:

- в поле **Название** появившегося окна укажите точное название загружаемого файла (с расширением, без пробелов);
- поле **Содержимое работы** оставьте пустым;
- из папки с решением перетащите загружаемый файл в поле **Приложение** или загрузите файл в это поле, используя кнопку "Добавить.." в меню этого поля;
- выполнив перечисленные пункты нажмите кнопку "Сохранить".

При необходимости, пока не окончена фаза представления работ, можно откорректировать представление работы нажав кнопку "Редактировать работу"

Lab3.hs

представлено: Воскресенье, 21 Октябрь 2018, 20:54

-  Lab3.hs



Самооценка

от Максим Кулаков

Оценка: 86,11 из 100,00

Форма оценки ▼

Критерий 1

Определение может иметь вид

`nat = [1 ..]`

или

`nat = [1, 2 ..]`

или

```
nat = iterate (+1) 1
```

или

```
nat = 1 : map (+1) nat
```

Все эти определения являются правильными.

- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 1

5

Комментарий к Критерий 1

Критерий 2

Определение `fibonacci` может иметь вид

```
fibonacci = 0 : 1 : zipWith (+) fibonacci (tail fibonacci)
```

- Не стоит ставить за решение больше трех баллов, если оно использует вспомогательную функцию.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 2

5

Комментарий к Критерий 2

Критерий 3

Определение `factorial` может иметь вид

```
factorial = 1 : zipWith (*) nat factorial
```

- Не стоит ставить за решение больше трех баллов, если оно использует вспомогательную функцию.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 3

5

Комментарий к Критерий 3

Критерий 4

Определение `powerSeq` может иметь вид

```
powerSeq x = 1 : map (* x) (powerSeq x)
```

или

```
powerSeq x = iterate (*x) 1
```

- Не стоит ставить за решение больше трех баллов, если оно использует вспомогательную функцию.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 4

5

Комментарий к Критерий 4

Критерий 5

Определение `findCloseEnough` может иметь вид

```
findCloseEnough eps stream =  
  second $ find (closeEnough eps) (zipWith (,) stream (tail stream))  
  where  
    closeEnough eps (a, b) = abs (a - b) <= eps  
    second (Just (a, b)) = b
```

- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 5

5

Комментарий к Критерий 5

Критерий 6

Определение `streamSum` может иметь вид

```
streamSum stream = 0 : zipWith (+) stream (streamSum stream)
```

- Не стоит ставить за решение больше трех баллов, если оно использует вспомогательную функцию.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень

совпадения решения с приведенными вариантами.

Оценка для Критерий 6

5

Комментарий к Критерий 6

Критерий 7

Определение `expSummands` может иметь вид

```
expSummands x = zipWith divIntInt (powerSeq x) factorial
  where divIntInt x y = x / fromIntegral y
```

- Не стоит ставить за решение больше двух баллов, если оно не использует `zipWith` или `factorial`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 7

5

Комментарий к Критерий 7

Критерий 8

Определение `expStream` может иметь вид

```
expStream = streamSum . expSummands
```

- Стоит снизить оценку на 1 балл, если есть лишнее оборачивание в функцию
- Не стоит ставить за решение больше двух баллов, если оно не использует `streamSum` или `expSummands`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 8

4

Комментарий к Критерий 8

лишнее оборачивание в функцию

Критерий 9

Определение `expAppr` может иметь вид

```
expAppr eps = (findCloseEnough eps) . expStream
```

или

```
expAppr eps x = findCloseEnough eps (expStream x)
```

- Так как оборачивание функции здесь не слишком очевидно, не стоит за это снижать оценку.
- Не стоит ставить за решение больше двух баллов, если оно не использует `findCloseEnough` или `expStream`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 9

5

Комментарий к Критерий 9

Критерий 10

Определение `derivativeAppr` может иметь вид

```
derivativeAppr f dx =  
  \ x -> (f (x + dx) - f x) / dx
```

или

```
derivativeAppr f dx x = (f (x + dx) - f x) / dx
```

- Не стоит ставить за решение больше трех баллов, если оно сложнее, чем в представленных вариантах.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 10

5

Комментарий к Критерий 10

Критерий 11

Определение `derivativeStream` может иметь вид

```
derivativeStream f x = map (\dx -> derivativeAppr f dx x) (powerSeq 0.5)
```

- Не стоит ставить за решение больше двух баллов, если оно не использует `map` или `derivativeAppr`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень

совпадения решения с приведенными вариантами.

Оценка для Критерий 11

5

Комментарий к Критерий 11

Критерий 12

Определение `derivative` может иметь вид

```
derivative f = (findCloseEnough epsilon') . derivativeStream f
```

или

```
derivative f x = findCloseEnough epsilon' (derivativeStream f x)
```

- Так как оборачивание функции здесь не слишком очевидно, здесь не стоит за это снижать оценку.
- Не стоит ставить за решение больше двух баллов, если оно не использует `findCloseEnough` или `derivativeStream`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 12

5

Комментарий к Критерий 12

Критерий 13

Определение `funAkStream` может иметь вид

```
funAkStream g = iterate (funTmp g) (\x -> x)  
  where funTmp f2 f1 = \x -> (derivative f1 x) / (f2 x)
```

Определение `invF` может иметь вид

```
invF f y0 x =  
  let  
    str1 = expSummands (x - f y0)  
    str2 = funAkStream (derivative f)  
    strLast = streamSum (zipWith (\ak -> \y -> (ak y0) * y) str2 str1)  
  in  
    findCloseEnough epsilon strLast
```

- Не стоит ставить оценку больше трех баллов, если решение для `funAkStream` не использует `iterate`.

- Стоит снизить оценку на два балла, если решение для `invF` не использует `expSummands`, `streamSum` или `findCloseEnough`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 13

1

Комментарий к Критерий 13

Критерий 14

Определение `average` может иметь вид

$$\text{average } x \ y = (x + y) / 2$$

- Не стоит ставить за решение больше двух баллов, если его логика сложнее, чем у представленного выше.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 14

5

Комментарий к Критерий 14

можно было обойтись без конвертации

Критерий 15

Определение `averageDump` может иметь вид

$$\text{averageDump } f \ x = \text{average } (f \ x) \ x$$

- Не стоит ставить за решение больше двух баллов, если его логика сложнее, чем у представленного выше.
- Не стоит ставить за решение больше двух баллов, если оно не использует `average`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 15

5

Комментарий к Критерий 15

Критерий 16

Критерий 16

Определение `newtonTransform` может иметь вид

```
newtonTransform g x = x - g x / derivative g x
```

- Не стоит ставить за решение больше двух баллов, если его логика сложнее, чем у представленного выше.
- Не стоит ставить за решение больше двух баллов, если оно не использует `derivative`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 16

5

Комментарий к Критерий 16

Критерий 17

Определение `eitken` может иметь вид

```
eitken (x1 : xs @ (x2 : x3 : _)) =  
  (x1 * x3 - x2 * x2) / (x1 - 2 * x2 + x3) : eitken xs
```

- Определение `eitken` может быть составлено на основе вызовов функции `zipWith`. За это оценку снижать не нужно.
- Не следует ставить за решение более 1 балла, если оно использует `zipWith3`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 17

4

Комментарий к Критерий 17

пробел

Критерий 18

Определение `fixedPoint` может иметь вид

```
fixedPoint = iterate
```

- Не следует ставить за решение больше трех баллов, если оно не использует `iterate`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 18

5

Комментарий к Критерий 18

Критерий 19

Определение `fixedPointOfTransform` может иметь вид

```
fixedPointOfTransform f transform x0 =  
  findCloseEnough epsilon' (fixedPoint (transform f) x0)
```

- Не следует ставить за решение больше трех баллов, если оно не использует `findCloseEnough` или `fixedPoint`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 19

5

Комментарий к Критерий 19

Критерий 20

Определение `sqrt1` может иметь вид

```
sqrt1 x = fixedPointOfTransform (\y -> x / y) averageDump 1
```

- Не следует ставить за решение больше трех баллов, если оно не использует `fixedPointOfTransform` или `averageDump`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 20

5

Комментарий к Критерий 20

Критерий 21

Определение `cubert1` может иметь вид

```
cubert1 x = fixedPointOfTransform (\y -> x / y / y) averageDump 1
```

- Не следует ставить за решение больше трех баллов, если оно не использует `fixedPointOfTransform` или `averageDump`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 21

5

Комментарий к Критерий 21

Критерий 22

Определение `sqrt2` может иметь вид

```
sqrt2 x = fixedPointOfTransform (\y -> y * y - x) newtonTransform 1
```

- Не следует ставить за решение больше трех баллов, если оно не использует `fixedPointOfTransform` или `newtonTransform`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 22

5

Комментарий к Критерий 22

Критерий 23

Определение `cubert2` может иметь вид

```
cubert2 x = fixedPointOfTransform (\y -> y * y * y - x) newtonTransform 1
```

- Не следует ставить за решение больше трех баллов, если оно не использует `fixedPointOfTransform` или `newtonTransform`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 23

5

Комментарий к Критерий 23

Критерий 24

Определение `extremum` может иметь вид

```
extremum f =  
  let  
    f' = derivative f  
    localExtremum = fixedPointOfTransform f' newtonTransform 0.1  
    extremumType val  
      | val > epsilon = "minimum"  
      | val < -epsilon = "maximum"  
      | otherwise = "inflection"  
  in (localExtremum, extremumType $ derivative f' localExtremum)
```

- Не следует ставить за решение больше трех баллов, если оно не использует `derivative`, `fixedPointOfTransform` или `newtonTransform`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 24

5

Комментарий к Критерий 24

Критерий 25

Определение `myPi` может иметь вид

```
myPi = (findCloseEnough epsilon'' (eitken (streamSum piSummands))) * 4  
  where  
    odds = 1.0 : map (+2) odds  
    signs = 1.0 : map (* -1) signs  
    piSummands = map (1/) (zipWith (*) odds signs)
```

- Не следует ставить за решение больше трех баллов, если оно не использует `findCloseEnough`, `eitken` или `streamSum`.
- Ваша оценка должна быть беспристрастной. Вполне нормально, если решение отличается от приведенных выше вариантов. Вы проверяете правильность решения и его стиль, а не степень совпадения решения с приведенными вариантами.

Оценка для Критерий 25

4

Комментарий к Критерий 25

пробелы

НАВИГАЦИЯ

В начало



■ Личный кабинет


Страницы сайта

Мои курсы


Графика Осень 2018

ЯП 2018

Участники

 Значки

 Компетенции

 Оценки

Общее

Форумы курса

Материалы по тематике курса

Лабораторные работы

Оценка лабораторных работ

 Лабораторная работа №0

 Лабораторная работа №1

 Лабораторная работа №2

 **Лабораторная работа №3**

■ Моя работа

 Лабораторная работа №4

 Лабораторная работа №5

 Лабораторная работа №6

Раздел 1. Standard ML

Раздел 2. Haskell

Раздел 3. LISP

Раздел 4. Ruby

Раздел 5. PROLOG

Реферат

Аттестация

Бонусы

Напоминалки

ИКБ

On-line

Вы зашли под именем Максим Кулаков (Выход)

ЯП 2018