

ТЕХНОЛОГИИ В ОБРАЗОВАНИИ

УНИВЕРСИТЕТ

МИКРОЭЛЕКТРОНИКА

ИННОВАЦИИ

КАТАЛИТИЧЕСКИЕ

МАТЕРИАЛЫ

ДИЗАЙН

ЛЕКАРСТВ

ТОЧКА

СБОРКИ

НАУЧНАЯ

ЛАБОРАТОРИЯ

ГЕОХИМИЯ

ИНЖИНИРИНГ

ГЕОФИЗИКА

ГИБРИДНЫЕ

МАТЕРИАЛЫ

ЭНЕРГОСБЕРЕЖЕНИЕ

НГУ

ВЫСОКИЕ

ЭНЕРГИИ

БИОТЕХНОЛОГИИ

МОДЕЛИРОВАНИЕ

НАНОТЕХНОЛОГИИ

СЕМИОТИКА

НАУКА

КОГНИТИВНЫЕ ТЕХНОЛОГИИ

МАТЕМАТИЧЕСКОЕ МОДЕЛИРОВАНИЕ

ЭЛЕМЕНТАРНЫЕ

ЧАСТИЦЫ

ГЕОЛОГИЯ

КВАНТОВЫЕ

ТЕХНОЛОГИИ

БИОЛОГИЯ

ТЕМНАЯ

МАТЕРИЯ

ФОТОНИКА

БИОМЕДИЦИНА

ПРИКЛАДНЫЕ

ИССЛЕДОВАНИЯ

РАЗВИТИЕ

АСТРОНОМИЯ

ГЛОБАЛЬНЫЕ ПРИОРИТЕТЫ

АСТРОФИЗИКА

БИОИНФОРМАТИКА

ЛАЗЕРНАЯ

ФИЗИКА

АРХЕОЛОГИЯ

ЭКОНОМИКА

ЗНАНИЙ

СОТРУДНИЧЕСТВО

АРКТИКА

IT

DEEP

LEARNING

ИЗУЧЕНИЕ

МОЗГА

КОГНИТИВНЫЕ

ТЕХНОЛОГИИ

N* Новосибирский
государственный
университет

***НАСТОЯЩАЯ НАУКА**



Get Programming with Haskell

TECHNOLOGIES IN EDUCATION
UNIVERSITY^{NSU}

MICROELECTRONICS
INNOVATIONS
CATALYTIC
MATERIALS
ASSEMBLY
POINT
SCIENTIFIC
LABORATORY
HYBRID
MATERIALS
GEOPHYSICS
ENGINEERING
ENERGY CONSERVATION
BIOTECHNOLOGY
GEOCHEMISTRY
NANOTECHNOLOGY
HIGH
ENERGIES
SEMIOTICS
SCIENCE
MATHEMATICAL
MODELING
DEVELOPMENT
ELEMENTARY
PARTICLES
THE ARCTIC REGIONS
DARK
MATTER
QUANTUM
TECHNOLOGIES
BIOMEDICINE
APPLIED
STUDIES
PHOTONICS
ASTRONOMY
GLOBAL PRIORITY
ASTROPHYSICS
BIOINFORMATICS
LASER
PHYSICS
KNOWLEDGE
ECONOMY
GEOLOGY
ARCHEOLOGY
COGNITIVE
TECHNOLOGIES

N* Novosibirsk
State
University
*THE REAL SCIENCE



Татьяна Антоновна Синёва
t.sineva@g.nsu.ru
@tsineva

Деканат: каб. 2346
Режим работы: 9:00 – 16:00 (понедельник
– пятница,
обед: 13:00 – 14:00)

Выдача справок:
Людмила Валентиновна Каменских
dekanat_mir@nsu.ru

*КАК ВЫЖИТЬ В ЭТОМ СЕМЕСТРЕ?!

I семестр:

1-я контрольная неделя - с 10 по 15 октября 2022 г.

2-я контрольная неделя - с 14 по 19 ноября 2022 г.

II семестр:

1-я контрольная неделя - с 13 по 18 марта 2023 г.

2-я контрольная неделя - с 17 по 22 апреля 2023 г.

*КАК ВЫЖИТЬ В ЭТОМ СЕМЕСТРЕ?!

I семестр:

Контрольная работа

2 месячных набора задач

II семестр:

Проектная деятельность с защитой проектов

в период с 27 по 30 марта (возможность получить 5 за семестр)

*А ОТДЫХАТЬ КОГДА?!

Ноябрь						
пн	вт	ср	чт	пт	сб	вс
	1	2	3*	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

Февраль						
пн	вт	ср	чт	пт	сб	вс
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22*	23	24	25	26
27	28					

УЧИМСЯ!!!

*А ОТДЫХАТЬ КОГДА?!

Зачётная неделя: 22.12.2022 – 31.12.2022

Сессия: 09.01.2023 – 25.01.2023

Пересдачи: 26.01.2023 – 10.02.2023

Каникулы: 26.01.2023 – 01.02.2023

<https://www.nsu.ru/n/sveden/education/>

НГУ

Сведения об образовательной организации

Образование

Высший колледж информатики

Бакалавриат

Календарный учебный график 2022

*Haskell

Язык Haskell — чистый функциональный язык программирования с «ленивой» семантикой исполнения и полиморфной статической типизацией. Язык назван в честь американского логика и математика Хаскелла Брукса Карри.

Справку по функциям стандартной библиотеки (и не только) можно получить с помощью онлайн системы [Hoogle](#).

*Что значит функциональный?

Все функции в Haskell следуют трём правилам, которые заставляют их вести себя как функции в математике:

- все функции должны принимать аргумент;
- все функции должны возвращать значение;
- функция возвращает одно и то же значение для одного аргумента.

*GHCi

> ghci
Prelude>

Prelude — стандартная библиотека полезных функций

*Команды

:q выйти из GHCi

?: список доступных команд и их краткое описание

:l <имя программы> загрузка программы с указанным именем

:r перегрузка текущего модуля

:t <имя функции> печать типа выражения

:i <имя функции> печать сигнатуры функции

*Отрицательные числа

Вычислите:

- $2 * (-5)$
- $2 * -5$
- $2 * -5$

*Функция logBase

Функции **logBase**, требуется передать основание логарифма и аргумент, на котором логарифм будет вычислен.

Какие из следующих вызовов обеспечат вычисление логарифма по основанию 2 от 8?

1. **logBase (2, 8)**
2. **(logBase, 2, 8)**
3. **logBase 2 8**
4. **(logBase 2) 8**
5. **logBase (2 8)**

*Возведение в степень

Вычислите:

- 100^{10}
- 2^0
- 2^{**1}
- $5^{(-1)}$
- $5^{**(-1)}$

*Сравнение

2 == 5

2 == True

'a' != 'A'

*Функции

- **div**
- **mod**
- **min**
- **max**
- **sqrt**
- **succ**
- **quot**
- **log**
- **logBase**

*Префиксная форма записи

(+) 2 3

(^) 6 8

(**) 4 (-1)

*Инфиксная форма записи

$123 \text{ `mod` } 10$

$(-123) \text{ `div` } 10$

$123 \text{ `quot` } 10$

*Остаток от деления. Всё ли так?

Вычислите:

- $\text{mod } 23 \ 10$
- $23 \ \text{`mod` } 10$
- $23 \ \text{`rem` } 10$
- $(-23) \ \text{`rem` } 10$
- $(-23) \ \text{`mod` } 10$
- $-23 \ \text{`mod` } 10$

*Пользовательские функции

Запустите Ваш текстовый редактор и создайте файл **Hello.hs**, содержащий следующую строку кода:

```
main = putStrLn "Hello, world!"
```

Вызовите интерпретатор GHCi с параметром — именем файла исходного кода:

```
>ghci Hello.hs
```

(Файл должен располагаться в том же каталоге, откуда происходит вызов интерпретатора.) Проверьте, что загрузка модуля прошла успешно, вызвав в интерпретаторе определенную вами функцию **main**:

```
GHCi> main  
Hello, world!
```

*Пользовательские функции

```
module Test where  
sayHello = putStrLn "TEST!!!"
```

:load Test

ИЛИ

:reload Test

*Test> sayHello

*Объявление функции

exponentiation :: Int -> Int
exponentiation b = b * b

*Упражнение

Определите функцию, вычисляющую следующее выражение:

$$y = \log_2 \log_3 \log_4 x$$

*Упражнение

Реализуйте функцию трех аргументов `lenVec3`, которая вычисляет длину трехмерного вектора. Аргументы функции задают декартовы координаты конца вектора, его начало подразумевается находящимся в начале координат.

Для извлечения квадратного корня воспользуйтесь функцией `sqrt`, определенной в стандартной библиотеке.

```
GHCI> lenVec3 2 3 6
```

```
7.0
```

*Что отличает функциональные языки от императивных?

Все функции в Haskell чистые, т.е. значение функции полностью определяется значениями переданных в нее аргументов. Никакие другие источники данных не могут влиять на возвращаемое значение функции.

Следствие: функция, которая не принимает ни одного аргумента - константа.

```
Prelude> let twentyTwo = 18 + 4
```

```
Prelude> twentyTwo
```

```
22
```

В Haskell нельзя определить функцию не принимающую аргументов, которая бы возвращала разные значения при разных вызовах!!!