# Оглавление

[1. Оглавление 1](#_Toc56699874)

[2. Io 2](#_Toc56699875)

[2.1. Общая характеристика и история создания 2](#_Toc56699876)

[2.2. Система типов и особенности архитектуры 2](#_Toc56699877)

[2.3. Примеры кода и интересные факты 3](#_Toc56699878)

[3. Idris 3](#_Toc56699879)

[3.1. Общая характеристика и история создания 3](#_Toc56699880)

[3.2. Система типов и особенности архитектуры 3](#_Toc56699881)

[3.3. Примеры кода и интересные факты 3](#_Toc56699882)

[4. Swift 4](#_Toc56699883)

[4.1. Общая характеристика и история создания 4](#_Toc56699884)

[4.2. Система типов и особенности архитектуры 4](#_Toc56699885)

[4.3. Примеры кода и интересные факты 5](#_Toc56699886)

[5. Список литературы 5](#_Toc56699887)

[6. Тест 5](#_Toc56699888)

[7. Задание на “5” 5](#_Toc56699889)

# Io

## Общая характеристика и история создания

Рисунок 1

“Io (первая буква — заглавная «i») — скриптовый, чисто объектно-ориентированный язык программирования с динамической типизацией. Язык был разработан Стивом Декортом (англ. Steve Dekorte) в 2002 году. Реализация является кроссплатформенной, открытой (лицензия BSD) и рассчитана на лёгкость встраивания в качестве скриптового языка.” [1]

“Язык Io появился в 2002 году, когда его автор Стив Декорт (Steve Dekorte), пытаясь лучше понять принципы функционирования и реализации языков программирования, решил совместить теорию с практикой и написать собственный простой и лаконичный язык. Декорт спроектировал Io, как чистый объектно-ориентированный язык, основанный на прототипах, с динамической типизацией, в котором устранено различие понятий "класс" и "экземпляр". В Wikipedia отмечается влияние на Io таких языков, как Smalltalk (принцип "всё есть объект" и динамическая типизация), Self и NewtonScript (объектная модель, основанная на прототипах), Lisp (равнозначность инструкций и данных в коде программы), а также Lua и Act1.” [2]

## Система типов и особенности архитектуры

“Строки в языке Io представлены в виде последовательностей символов, заключённых в двойные кавычки. Если сам символ двойной кавычки содержится в строке, то требуется его экранирование

При записи чисел допускается применение практически всех стандартных форматов, также в Io поддерживается запись числовых значений в шестнадцатеричном представлении.” [2]

Числовые форматы, поддерживаемые в языке Io:

1. Io> 12 + 18
2. ==> 30
3. Io> 3.14 \* 2
4. ==> 6.2800000000000002
5. Io> .987 + 11.003
6. ==> 11.9900000000000002
7. Io> 2e4 - 2e3
8. ==> 18000
9. Io> 1.655e-3 - 5.112e-4
10. ==> 0.0011438
11. Io> 0xFF + 0x3
12. ==> 258

## Примеры кода и интересные факты

Цикл While:

# Idris

1. Io> a := 10
2. ==> 10
3. Io> **while**( a < 20,
4. )-> a println
5. )-> a = a + 2
6. )-> )
7. 10
8. 12
9. 14
10. 16
11. 18
12. ==> 20
13. Io>

## Общая характеристика и история создания

Рисунок 2

“Idris – это чисто функциональный язык программирования с зависимыми типами, необязательным отложенным вычислением и такими функциями, как проверка целостности. Идрис можно использовать в качестве помощника по проверке, но он разработан как язык программирования общего назначения, похожий на Haskell.” [3]

## Система типов и особенности архитектуры

“Язык сочетает особенности основных популярных языков функционального программирования с возможностями, заимствованными из систем автоматического доказательства, фактически размывая границу между этими двумя классами языков.” [4]

«Система типов Idris похожа на систему Agda, а доказательства аналогичны системе Coq, включая тактику (функции / процедуры доказательства теорем) посредством отражения разработчика. По сравнению с Agda и Coq, Idris отдает приоритет управлению побочными эффектами и поддержке встроенных предметно-ориентированных языков. Idris компилируется в C полагаясь на настраиваемый копирующий сборщик мусора с использованием алгоритма Чейни) и в JavaScript как на браузере, так и на Node.js). Существуют сторонние генераторы кода для других платформ, включая JVM, CIL и LLVM.» [3]

## Примеры кода и интересные факты

“Синтаксис языка (как и у Agda) близок к синтаксису языка Haskell. Программа Hello, world! на Idris выглядит следующим образом:” [3]

1. module Main
3. main : IO ()
4. main = putStrLn "Hello, World!"

# Swift

## Общая характеристика и история создания

Рисунок 3

“Swift — открытый мультипарадигмальный компилируемый язык программирования общего назначения. Создан компанией Apple в первую очередь для разработчиков iOS и macOS. Swift работает с фреймворками Cocoa и Cocoa Touch и совместим с основной кодовой базой Apple, написанной на Objective-C. Swift задумывался как более лёгкий для чтения и устойчивый к ошибкам программиста язык, нежели предшествовавший ему Objective-C. Программы на Swift компилируются при помощи LLVM, входящей в интегрированную среду разработки Xcode 6 и выше. Swift может использовать рантайм Objective-C, что делает возможным использование обоих языков (а также С) в рамках одной программы.” [6]

## Система типов и особенности архитектуры

|  |  |
| --- | --- |
| Int8 | представляет целые числа со знаком размером не более 8 бит (от -128 до 127) |
| UInt8 | представляет целые положительные числа размером не более 8 бит (от 0 до 255) |
| Int16 | представляет целые числа со знаком размером не более 16 бит (от -32768 до 32767) |
| UInt16 | представляет целые положительные числа размером не более 16 бит (от 0 до 65535) |
| Int32 | представляет целые числа со знаком размером не более 32 бита (от -2147483648 до 2147483647) |
| UInt32 | представляет целые положительные числа размером не более 32 бита (от 0 до 4294967295) |
| Int64 | представляет целые числа со знаком размером не более 64 бита (от -9223372036854775808 до 9223372036854775807) |
| UInt64 | представляет целые положительные числа размером не более 64 бита (от 0 до 18446744073709551615) |
| Int | представляет целые числа со знаком, например, 1, -30, 458. На 32-разрядных платформах эквивалентен Int32, а на 64-разрядных - Int64 |
| UInt | представляет целые положительные числа, например, 1, 30, 458. На 32-разрядных платформах эквивалентен UInt32, а на 64-разрядных - UInt64 |
| Float | 32-битное число с плавающей точкой, содержит до 6 чисел в дробной части |
| Double | 64-битное число с плавающей точкой, содержит до 15 чисел в дробной части |
| Float80 | 80-битное число с плавающей точкой |
| Bool | представляет логическое значение true или false |
| String | представляет строку |
| Character | представляет отдельный символ |

## Примеры кода и интересные факты

“Расширения в Swift:” [5]

1. **struct**: Foo {
2. let value : Int
3. }
5. extension Foo : Printable {
6. var description : String {
7. get {**return** "Foo"}
8. }
9. }
11. extension Foo : Equatable {
13. }
15. func ==(lhs: Foo, rhs: Foo) -> Bool {
16. **return** lhs.value == rhs.value
17. }

 “Вызов C кода напрямую:” [5]

1. @asmname("my\_c\_func")
2. func my\_c\_func(UInt64, CMutablePointer<UInt64>) -> CInt;

# Список литературы

* 1. ru.wikipedia.org/wiki/Io
  2. www.ibm.com/developerworks/ru/library/l-io\_01/index.html
  3. ru.qaz.wiki/wiki/Idris\_(programming\_language)
  4. ru.wikipedia.org/wiki/Idris\_(язык\_программирования)
  5. habr.com/ru/company/yandex/blog/228709/
  6. ru.wikipedia.org/wiki/Swift\_(язык\_программирования)

# Тест

# Задание на “5”