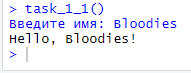
# **Часть 1: Ввод и вывод**

## **Задача 1:**

Напишите код, который запрашивает у пользователя его имя с клавиатуры и выводит на экран сообщение вида “Hello, name!”.

### **Работа программы:**



### **Листинг:**

task\_1\_1 <- function()

{

name <- readline("Введите имя: ")

cat(sprintf("Hello, %s!", name))

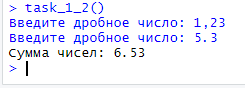
}

## **Задача 2:**

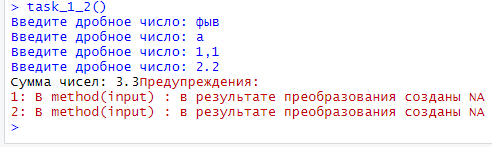
Напишите код, который запрашивает у пользователя два дробных числа (одно число — один запрос) и выводит на экран их сумму. Решите эту задачу в предположении, что:

* пользователь вводит дробные числа с десятичным разделителем в виде точки;
* пользователь вводит дробные числа с десятичным разделителем в виде запятой.

### **Работа программы:**



С проверкой на неправильный ввод



### **Листинг:**

task\_1\_2 <- function()

{

num1 <- suppressWarnings(validate(as.numeric, "numeric"))

num2 <- suppressWarnings(validate(as.numeric, "numeric"))

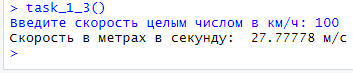
cat("Сумма чисел:", num1 + num2);

}

## **Задача 3:**

Напишите код, который запрашивает у пользователя значение скорости в километрах в час и выводит на экран значение скорости в метрах в секунду.

### **Работа программы:**



### **Листинг:**

task\_1\_3 <- function()

{

spd <- validate(as.integer, "speed")

cat("Преобразованная скорость:", (spd\*1000)/3600, "м/с")

}

# **Часть 2: Векторы**

## **Задача 1:**

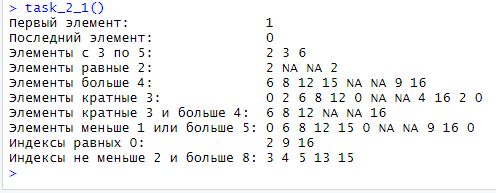
Дан вектор g, в котором хранятся следующие значения:

1, 0, 2, 3, 6, 8, 12, 15, 0, NA, NA, 9, 4, 16, 2, 0

Создайте вектор g. Используя R, выведите на экран:

* первый элемент вектора;
* последний элемент вектора;
* элементы вектора с третьего по пятый включительно;
* элементы вектора, которые равны 2;
* элементы вектора, которые больше 4;
* элементы вектора, которые кратны 3 (делятся на 3 без остатка);
* элементы вектора, которые больше 4 и кратны 3;
* элементы вектора, которые или меньше 1, или больше 5;
* индексы элементов, которые равны 0;
* индексы элементов, которые не меньше 2 и не больше 8.

### **Работа программы:**



### **Листинг:**

task\_2\_1 <- function()

{

vector <- c(1, 0, 2, 3, 6, 8, 12, 15, 0, NA, NA, 9, 4, 16, 2, 0)

cat('Первый элемент: ', vector[1], '\n')

cat('Последний элемент: ', vector[length(vector)], '\n')

cat('Элементы с 3 по 5: ', vector[3:5], '\n')

cat('Элементы равные 2: ', vector[vector == 2], '\n')

cat('Элементы больше 4: ', vector[vector > 4], '\n')

cat('Элементы кратные 3: ', vector[vector %% 2 == 0], '\n')

cat('Элементы кратные 3 и больше 4: ', vector[vector > 4 & vector %% 2 == 0], '\n')

cat('Элементы меньше 1 или больше 5:', vector[vector < 1 | vector > 5], '\n')

cat('Индексы равных 0: ', which(vector == 0), '\n')

cat('Индексы не меньше 2 и больше 8:', which(vector >= 2 & vector < 8), '\n')

}

## **Задача 2:**

Напишите код, который заменяет последний элемент любого вектора на пропущенное значение (NA). Ваш код должен работать для вектора любой длины.

### **Работа программы:**



### **Листинг:**

task\_2\_2 <- function()

{

vector <- c(1, 200, 0, TRUE, 5, 6, 7, NA, 2, -1, 20.0, "TEST")

vector[length(vector)] = NA

cat("Вектор с замененным последним элементом на NA:", vector)

}

## **Задача 3:**

Напишите код, который выводит на экран индексы пропущенных значений в векторе.

### **Работа программы:**



### **Листинг:**

task\_2\_3 <- function()

{

vector <- c(1, 200, NA, NA, TRUE, 5, NA, 7, NA, 2, -1, 20.0, "TEST")

cat("Индексы пропущенных элементов:", which(is.na(vector)))

}

## **Задача 4:**

Напишите код, который считает, сколько пропущенных значений в векторе.

### **Работа программы:**



### **Листинг:**

task\_2\_4 <- function()

{

vector <- c(1, 200, NA, NA, TRUE, 5, NA, 7, NA, 2, -1, 20.0, "TEST")

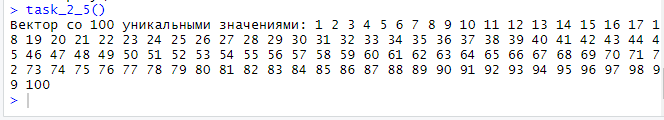
cat("Число пропущенных элементов:", length(vector[is.na(vector)]))

}

## **Задача 5:**

Напишите код, который позволяет создать вектор из id (уникальных номеров) респондентов, если известно, что в опросе участвовало 100 респондентов.

### **Работа программы:**



### **Листинг:**

task\_2\_5 <- function()

{

amount = 100;

vector <- c()

for (c in 1:amount)

vector[c] = c

cat("Вектор со", amount, "уникальными значениями:", vector)

}

## **Задача 6:**

*Решение задачи предполагает рациональное решение с помощью функций R, не нужно создавать вектора перечислением большого числа элементов вручную.*

Известно, что в таблице хранятся показатели по 3 странам за 5 лет (см. ниже).

Создайте вектор который мог бы послужить столбцом country в этой таблице. Создайте вектор, который мог бы послужить столбцом year в этой таблице.

Таблица выглядит так:

|  |  |  |
| --- | --- | --- |
|  | | |
|  | country | year |
|  | | |
| 1 | France | 2000 |
| 2 | France | 2001 |
| 3 | France | 2002 |
| 4 | France | 2003 |
| 5 | France | 2004 |
| 6 | Italy | 2000 |
| 7 | Italy | 2001 |
| 8 | Italy | 2002 |
| 9 | Italy | 2003 |
| 10 | Italy | 2004 |
| 11 | Spain | 2000 |
| 12 | Spain | 2001 |
| 13 | Spain | 2002 |
| 14 | Spain | 2003 |
| 15 | Spain | 2004 |

### **Работа программы:**



### **Листинг:**

task\_2\_6 <- function()

{

repeats = 5; # Количество повторений

yearFrom = 2000; # Год начала отсчета

countries = c()

for (i in c("France", "Italy", "Spain"))

countries <- c(countries, rep(i, repeats))

years = c();

for (i in 1:repeats)

years[i] = yearFrom + i - 1

table <- data.frame("Страна" = countries, "Год" = years)

print(table)

}

## **Задача 7:**

Исследователь решил создать вектор income, в котором сохранены доходы нескольких респондентов:

income <- c(10000, 32000, 28000, 150000, 65000, 1573)

Исследователю нужно получить вектор income\_class, состоящий из 0 и 1: 0 ставится, если доход респондента ниже среднего дохода, а 1 — если больше или равен среднему доходу. Создайте вектор income\_class.

*Подсказка:* сначала можно посчитать среднее значение по вектору income и сохранить его в какую-нибудь переменную. Пользоваться встроенной функцией mean() нельзя.

### **Работа программы:**



### **Листинг:**

{

income <- c(10000, 32000, 28000, 150000, 65000, 1573)

average = sum(income) / length(income)

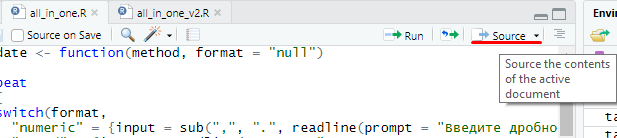
income\_class <- replace(temp <- replace(income, income < average, 0), temp >= average, 1)

cat("Вектор income\_class:", income\_class)

}

# **Способы запуска:**

1. запуск всего кода, чтобы инициализировать функции



1. запуск заданий:

task\_[1]\_[2]()

[1] – номер части

[2] – номер задания из части

