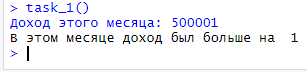
# **Задания**

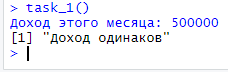
## **Задача 1:**

В переменной income сохранено значение дохода компании за последний месяц в рублях.

1. Создайте переменную income и присвойте ей любое (желательно разумное) значение.
2. Создайте переменную log\_income, в которой содержится натуральный логарифм значения дохода.
3. Создайте переменную income\_pre и присвойте ей значение 500000. Напишите строку кода, которая позволит узнать, в каком месяце, текущем или предыдущем, доход компании был больше.

### **Работа программы:**







### **Листинг:**

task\_1 <- function()

{

income <- as.numeric(readline("Доход этого месяца: "))

#log\_income

income\_pre <- 500000

if(income\_pre>income)

{

cat("В предыдущем месяце доход был больше на ", (income\_pre-income))

}else if (income\_pre<income)

{

cat("В этом месяце доход был больше на ", (income-income\_pre))

}else

{

print("Доход одинаков")

}

}

## **Задача 2:**

В двух переменных сохранены некоторые значения:

x <- 2

y <- 4

Напишите код, который позволит поменять значения в переменных x и y местами, то есть получить следующее:

x

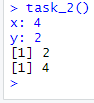
## [1] 4

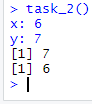
y

## [1] 2

* *Внимание:* Ваш код должен работать для любых значений x, y. Создавать дополнительные переменные можно!

### **Работа программы:**





### **Листинг:**

task\_2 <- function()

{

x <- as.numeric(readline("x: "))

y <- as.numeric(readline("y: "))

x\_exch <- y

y\_exch <- x

y <- y\_exch

x <- x\_exch

print(x)

print(y)

}

## **Задача 3:**

Даны следующие данные

x <- 3.5

y <- "2,6"

z <- 1.78

h <- TRUE

Определите типы переменных.

### **Работа программы:**



### **Листинг:**

task\_3 <- function()

{

x <- 3.5 # numeric

y <- "2,6" # character

z <- 1.78 # numeric

h <- TRUE # logical

class(x);

class(y);

class(z);

class(h);

}

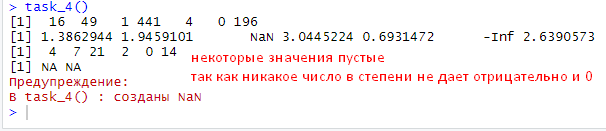
## **Задача 4:**

Создайте вектор q, состоящий из следующих значений: 4, 7, -1, 21, 2, 0, 14.

* Создайте вектор q\_sq, состоящий из квадратов значений вектора q.
* Создайте вектор q\_log, состоящий из натуральных логарифмов значений вектора q. Напишите (просто комментарием), почему в векторе q\_log есть пропущенные значения. Возможно, Вам потребуется почитать про логарифмы (Википедии хватит).
* Выведите на экран неотрицательные значения вектора q.
* Выведите на экран индексы элементов вектора q, которые кратны 7.

Выведите на экран элементы вектора q\_log, которые кратны 2 и больше 5.

### **Работа программы:**



### **Листинг:**

task\_4 <- function()

{

q <- c(4, 7, -1, 21, 2, 0, 14)

q\_sq <- q^2

print(q\_sq)

q\_log <- log(q,exp(1))

print(q\_log)

print(q[q >= 0])

which(q%%7 == 0)

print(q\_log[q\_log %%2 == 0] & q\_log[q\_log > 5])

}

## **Задача 5:**

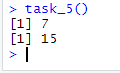
Политолог Мебейн (Walter R. Mebane) считает, что большая доля избирательных участков со значениями явки, заканчивающихся на 0 или 5, свидетельствует о фальсификациях результатов выборов. Аргументирует он это чисто психологическими причинами: если значения явки сочиняют люди, то они более склонны записывать круглые числа и числа, кратные 5.

Перед Вами вектор значений явки на избирательных участках в районе F страны Флатландии:

turnout <- c(100, 124, 121, 130, 150, 155, 144, 132, 189, 145, 125, 110, 118, 129, 127)

1. Выведите на экран индексы избирательных участков, где явка, согласно Мебейну, выглядит подозрительной (значения явки, кратные 10 или 5).
2. Определите долю таких подозрительных участков, выразите ее в процентах и округлите ответ до второго знака после запятой.

### **Работа программы:**



### **Листинг:**

task\_5 <- function()

{

turnout <- c(100, 124, 121, 130, 150, 155, 144, 132, 189, 145, 125, 110, 118, 129, 127)

which((turnout %%5 == 0) | (turnout %%10 == 0))

index <- length(turnout[turnout %%5 == 0])

print(index)

dole <- length(turnout)

print(dole)

part <- round((index/dole)\*100, 2)

}

## **Задача 6:**

Дан вектор z:

z <- c(8, NA, 7, 10, NA, 15, NA, 0, NA, NA, 87)

Выведите на экран индексы элементов вектора z, которые являются пропущенными значениями.

### **Работа программы:**



### **Листинг:**

task\_6 <- function()

{

z <- c(8, NA, 7, 10, NA, 15, NA, 0, NA, NA, 87)

which(is.na(z))

}

## **Задача 7:**

Дан вектор s:

s <- c("4,5", "6,8", "9,2", "1,75")

Получите, основываясь на векторе s, числовой вектор n (тип numeric).

### **Работа программы:**



### **Листинг:**

task\_7 <- function()

{

s <- c("4,5", "6,8", "9,2", "1,75")

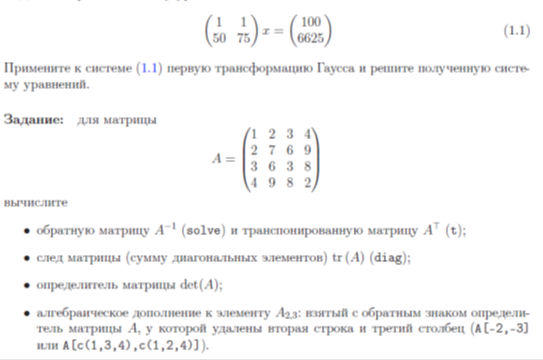
n <- as.numeric(gsub(",", ".", s))

cat(n)

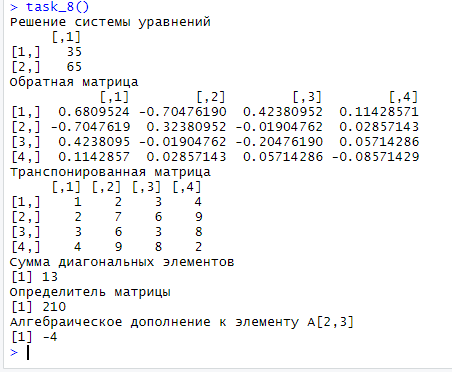
}

## **Задача 8:**

Решите систему уравнений



### **Работа программы:**



### **Листинг:**

task\_8 <- function()

{

Y <- c(1, 50, 1, 75);

dim(Y) <- c(2, 2);

Z <- c(100, 6625);

dim(Z) <- c(2, 1);

cat("Решение системы уравнений\n")

print(solve(Y, Z))

A <- c(1, 2, 3, 4, 2, 7, 6, 9, 3, 6, 3, 8, 4, 9, 8, 2);

dim(A) <- c(4, 4);

cat("Обратная матрица\n")

print(solve(A))

cat("Транспонированная матрица\n")

print(t(A))

cat("Сумма диагональных элементов\n")

print(sum(diag(A)))

cat("Определитель матрицы\n")

print(det(A))

Ax <- c(1, 3, 4, 2, 6, 9, 4, 8, 2);

dim(Ax) <- c(3, 3);

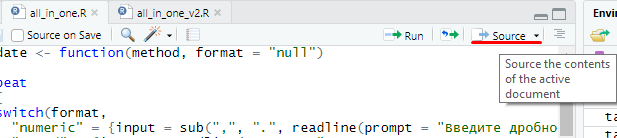
cat("Алгебраическое дополнение к элементу A[2,3]\n")

print(det(-Ax))

}

# **Способы запуска:**

1. запуск всего кода, чтобы инициализировать функции



1. запуск заданий:

task\_[1]()

[1] – номер задания

