Итоговая лабораторная работа.

1. Напишите функцию, которая переводит градусы из шкалы Фаренгейта в шкалу Цельсия (и обратно, с возможностью выбора направления конвертации). Входными данными могут быть как единичные значения, так и векторы значений градусов. Программа должна возвращать результат в виде таблицы данных, где в левом столбце указано(ы) значение(я) градусов в исходном формате, а в правом столбце - в том формате, в который переводим.

```r
#task 1

# Функция для конвертации из шкалы Фаренгейта в шкалу Цельсия и наоборот
convert_temp <- function(temp, direction = "FC") {
  # Проверка направления конвертации
  if (direction == "FC") {
    converted_temp <- (temp - 32) * 5/9  # Фаренгейт в Цельсий
  } else if (direction == "CF") {
    converted_temp <- temp * 9/5 + 32  # Цельсий в Фаренгейт
  } else {
    stop("Invalid input")  # Обработка некорректного направления конвертации

  }

  # Создание таблицы данных с исходными и сконвертированными значениями
  data <- data.frame(Original = temp, Converted = converted_temp)
  return(data)
}

# Примеры использования функции
# Конвертация из Фаренгейта в Цельсий
fahrenheit_temperatures <- c(32, 50, 68, 86)
conversion1 <- convert_temp(fahrenheit_temperatures, "FC")
print(conversion1)

# Конвертация из Цельсия в Фаренгейт
celsius_temperatures <- c(0, 10, 20, 30)
conversion2 <- convert_temp(celsius_temperatures, "CF")
print(conversion2)
```

```
> # Примеры использования функции
> # Конвертация из Фаренгейта в Цельсий
> fahrenheit_temperatures <- c(32, 50, 68, 86)
> conversion1 <- convert_temp(fahrenheit_temperatures, "FC")
> print(conversion1)
  Original Converted
1       32         0
2       50        10
3       68        20
4       86        30
> # Конвертация из Цельсия в Фаренгейт
> celsius_temperatures <- c(0, 10, 20, 30)
> conversion2 <- convert_temp(celsius_temperatures, "CF")
> print(conversion2)
  Original Converted
1        0        32
2       10        50
3       20        68
4       30        86
>
```

```
# Функция для конвертации из шкалы Фаренгейта в шкалу Цельсия и наоборот
convert_temp <- function(temp, direction = "FC") {
  # Проверка направления конвертации
  if (direction == "FC") {
    converted_temp <- (temp - 32) * 5/9  # Фаренгейт в Цельсий
  } else if (direction == "CF") {
    converted_temp <- temp * 9/5 + 32  # Цельсий в Фаренгейт
  } else {
    stop("Invalid input")  # Обработка некорректного направления конвертации

  }

  # Создание таблицы данных с исходными и сконвертированными значениями
  data <- data.frame(Original = temp, Converted = converted_temp)
  return(data)
}

# Примеры использования функции
# Конвертация из Фаренгейта в Цельсий
fahrenheit_temperatures <- c(32, 50, 68, 86)
conversion1 <- convert_temp(fahrenheit_temperatures, "FC")
print(conversion1)

# Конвертация из Цельсия в Фаренгейт
celsius_temperatures <- c(0, 10, 20, 30)
conversion2 <- convert_temp(celsius_temperatures, "CF")
print(conversion2)
```

2. Вычислите $\sum_{i=1}^{20}\sum_{j=1}^{i}\frac{i^4}{(3+ij)}$ наилучшим способом с точки зрения скорости исполнения кода.

```
# Выполнение задание без циклов с использованием встроенных функций sapply и lapply
i <- 1:20
j <- lapply(i, function(x) 1:x)
result <- sum(sapply(seq_along(i), function(x) {
  sum(i[[x]]^4 / (3 + i[[x]] * j[[x]]))
}))
print(result)
```

```
> i <- 1:20
> j <- lapply(i, function(x) 1:x)
> result <- sum(sapply(seq_along(i), function(x) {
+   sum(i[[x]]^4 / (3 + i[[x]] * j[[x]]))
+ }))
> print(result)
[1] 137295.9
```

3. Загрузите встроенную в R базу данных по автомобилям `mtcars`. С помощью библиотеки ggplot2 постройте точечную диаграмму, которая:

- демонстрирует связь между показателями «число лошадиных сил» (`hp`) и «вес» (`wt`);
- учитывает информацию о числе цилиндров у автомобиля (`cyl`);
- учитывает информацию о типе коробки передач - автомат или механика (`am`); легенда графика должна быть корректной и информативной. Точки, соответствующие автомобилям с автоматической коробкой передач, должны быть зеленого цвета ("green"), а с ручной – красного ("red").

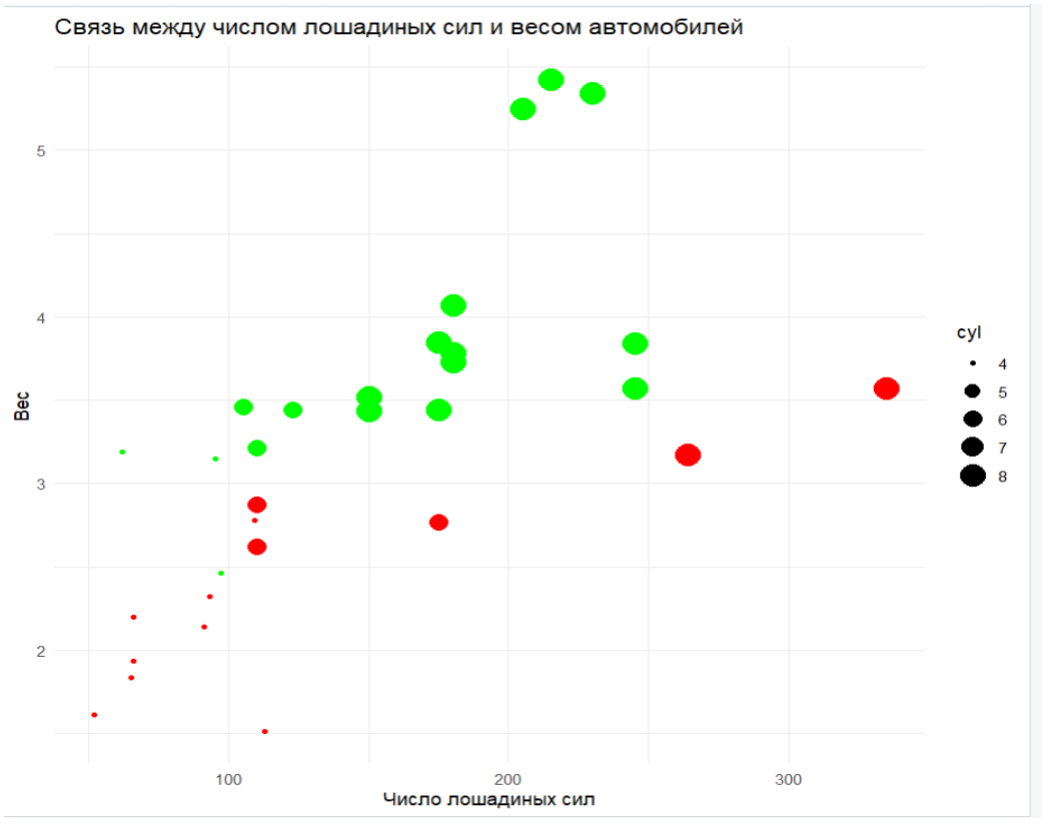
Подпишите оси графика. Добавьте заголовок графика.

```
# Загрузка библиотеки ggplot2
library(ggplot2)

# Загрузка встроенной базы данных mtcars
data(mtcars)

# Создание нового столбца для цвета в зависимости от типа коробки передач
mtcars$color <- ifelse(mtcars$am == 0, "green", "red")

# Построение графика
ggplot(mtcars, aes(x = hp, y = wt, color = color)) +
  geom_point(aes(size = cyl)) +
  scale_color_identity("Тип коробки передач", labels = c("Автомат",
"Механика"), breaks = c("green", "red")) +
  labs(title = "Связь между числом лошадиных сил и весом автомобилей",
       x = "Число лошадиных сил",
       y = "Вес") +
  theme_minimal()
```



Этот код создает точечную диаграмму, которая демонстрирует связь между числом лошадиных сил (hp) и весом (wt) автомобилей. Размер точек соответствует числу цилиндров (cyl), а цвет точек указывает на тип коробки передач (am): зеленый для автоматической и красный для механической. Заголовок графика и подписи осей также добавлены в соответствии с требованиями.

4. Напишите функцию, которая запрашивает у пользователя размерность матрицы (пользователь вводит в строке ввода число строк `nrows` и столбцов `ncols` через пробел), и создает единичную матрицу заданной размерности. Если невозможно создать единичную матрицу заданной размерности, на экран выводится соответствующее сообщение и создается нулевая матрица заданной размерности. Затем функция запрашивает у пользователя число элементов матрицы, которые нужно случайным образом выбрать и заменить на NA. Функция должна возвращать в итоге матрицу со случайным набором элементов NA, а также выводить на экран объем памяти, занимаемый данной матрицей.

Эта функция сначала запрашивает у пользователя размерность матрицы. Если невозможно создать единичную матрицу заданной размерности, она создает нулевую матрицу. Затем функция запрашивает у пользователя число элементов матрицы, которые нужно заменить на NA, и выполняет замену. В конце функция выводит объем памяти, занимаемый матрицей, и возвращает матрицу.

```
create_matrix <- function() {
  # Запрос размерности матрицы у пользователя
  cat("Введите число строк и столбцов через пробел: ")
  dims <- as.integer(strsplit(readline(), " ")[[1]])

  # Проверка корректности ввода
  if (any(is.na(dims)) || length(dims) != 2) {
    cat("Некорректный ввод. Пожалуйста, введите два целых числа через пробел.\n")
    return(NULL)
  }

  # Проверка, можно ли создать единичную матрицу заданной размерности
  if (dims[1] != dims[2]) {
    cat("Невозможно создать единичную матрицу заданной размерности. Создается нулевая матрица.\n")
    mat <- matrix(0, nrow = dims[1], ncol = dims[2])
  } else {
    mat <- diag(dims[1])
  }

  # Запрос числа элементов для замены на NA
  cat("Введите число элементов для замены на NA: ")
  n <- as.integer(readline())

  # Проверка корректности ввода
  if (is.na(n) || n < 0) {
    cat("Некорректный ввод. Пожалуйста, введите неотрицательное целое число.\n")
    return(NULL)
```

```
  }


  # Замена случайных элементов на NA
  if (n > length(mat)) {
    cat("Число элементов для замены больше общего числа элементов в матрице. Все элементы будут заменены на NA.\n")
    n <- length(mat)
  }
  idx <- sample(length(mat), n)
  mat[idx] <- NA


  # Вывод объема памяти, занимаемого матрицей
  cat("Объем памяти, занимаемый матрицей: ", object.size(mat), " байт\n")


  return(mat)
}
```

```
> mat_1 <- create_matrix()
Введите число строк и столбцов через пробел:
3 3
Введите число элементов для замены на NA:
2
Объем памяти, занимаемый матрицей:  344  байт
>
```

5. Загрузите встроенную в R базу данных `airquality`. Удалите из таблицы данных все строки, содержащие значения NA. Для получившейся таблицы данных напишите код, который выводит на экран 2 самых жарких месяца, а затем выводит таблицу данных, состоящую только из наблюдений для этих двух месяцев.

```
# Загрузка встроенной базы данных airquality
data(airquality)

# Удаление строк с NA
airquality <- na.omit(airquality)

# Нахождение двух самых жарких месяцев
hot_months <- sort(tapply(airquality$Temp, airquality$Month, mean), decreasing = TRUE)[1:2]

# Вывод на экран двух самых жарких месяцев
cat("Два самых жарких месяца: ", names(hot_months), "\n")

# Создание таблицы данных только для этих двух месяцев
hot_data <- airquality[airquality$Month %in% as.numeric(names(hot_months)), ]

# Вывод таблицы данных на экран
print(hot_data)
```

```
> print(hot_data)
    Ozone Solar.R Wind Temp Month Day
62    135     269  4.1   84     7   1
63     49     248  9.2   85     7   2
64     32     236  9.2   81     7   3
66     64     175  4.6   83     7   5
67     40     314 10.9   83     7   6
68     77     276  5.1   88     7   7
69     97     267  6.3   92     7   8
70     97     272  5.7   92     7   9
71     85     175  7.4   89     7  10
73     10     264 14.3   73     7  12
74     27     175 14.9   81     7  13
76      7      48 14.3   80     7  15
77     48     260  6.9   81     7  16
78     35     274 10.3   82     7  17
79     61     285  6.3   84     7  18
80     79     187  5.1   87     7  19
81     63     220 11.5   85     7  20
82     16       7  6.9   74     7  21
85     80     294  8.6   86     7  24
86    108     223  8.0   85     7  25
87     20      81  8.6   82     7  26
88     52      82 12.0   86     7  27
89     82     213  7.4   88     7  28
90     50     275  7.4   86     7  29
91     64     253  7.4   83     7  30
92     59     254  9.2   81     7  31
93     39      83  6.9   81     8   1
94      9      24 13.8   81     8   2
95     16      77  7.4   82     8   3
99    122     255  4.0   89     8   7
100    89     229 10.3   90     8   8
101   110     207  8.0   90     8   9
104    44     192 11.5   86     8  12
105    28     273 11.5   82     8  13
106    65     157  9.7   80     8  14
108    22      71 10.3   77     8  16
109    59      51  6.3   79     8  17
110    23     115  7.4   76     8  18
111    31     244 10.9   78     8  19
112    44     190 10.3   78     8  20
113    21     259 15.5   77     8  21
114     9      36 14.3   72     8  22
116    45     212  9.7   79     8  24
117   168     238  3.4   81     8  25
118    73     215  8.0   86     8  26
120    76     203  9.7   97     8  28
121   118     225  2.3   94     8  29
122    84     237  6.3   96     8  30
123    85     188  6.3   94     8  31
>
```