Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»**

**(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики**

**Кафедра вычислительных технологий**

**Отчёт лабораторная №8**

**Дисциплина: Обработка больших данных**

Работу выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гаранина Л.В.

Направление подготовки 02.03.02 Фундаментальная информатика и

информационные технологии

Преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Яхонтов А.А.

Краснодар

2025

**Тема:** Элементарные способы работы с графами в R.

**Цель**: Научиться работать с графами. Освоить основные функции обработки графов – их задание, визуализация, оформление, извлечение информации о графе.

**Задание:**

1. Создайте кольцевой граф g со случайным числом вершин G\_size (от N+10 до (N/10+5)2+5N). Выведите число ребер и вершин этого графа. Постройте граф, выведите его матрицу смежности.
2. Создайте граф g1 из пустого графа с числом вершин G\_size желтого цвета. Добавьте ему N\*8 случайных ребер, сформированных из вектора вершин, окрасьте ребра красным цветом, нарисуйте граф и выведите его матрицу смежности. Добавьте графу g1 еще N\*10 случайных ребер, сформированных из вектора вершин, окрасьте ребра синим цветом, нарисуйте граф и выведите его матрицу смежности.
3. Добавьте ребра между вершиной 2N+23 и 2N+20, 2N+12 и N+15, 2N-1 и N+8, 2N и 2N+1, N+7 и N+13, окрасьте их в черный цвет (предварительно проверьте существуют ли такие вершины – функцией %in% либо match, для несуществующих вершин ребра не добавляйте). Нарисуйте граф. Выведите соседей N - й вершины, ребра, инцидентные этой вершине. Соединены ли вершины N+10 и N+12? Выведите матрицу смежности.
4. Добавьте еще одну вершину и подключите ее к той, которая имеет наибольшее количество связанных с ней узлов. Присвойте имена всем вершинам (например, буквы в алфавитном порядке – используйте заглавные и, если не хватит, строчные буквы). Выведите матрицу смежности. Выберите вершины, для которых значение связности меньше 5 и больше 2.
5. Испробуйте алгоритмы размещения Вашего графа (in\_circle, in\_tree, lattice).
6. Выполните измерение диаметра графа g1, выведите список самых коротких путей для каждой вершины и откалибруйте величины вершин согласно их степеней.
7. В Банановой республике очень много холмов, соединенных мостами. На химическом заводе произошла авария, в результате чего испарилось экспериментальное удобрение "зован". На следующий день выпал цветной дождь, причем он прошел только над холмами. В некоторых местах падали красные капли, в некоторых - синие, а в остальных - зеленые, в результате чего холмы стали соответствующего цвета. Президенту Банановой республики это понравилось, но ему захотелось покрасить мосты между вершинами холмов так, чтобы мосты были покрашены в цвет холмов, которые они соединяют. К сожалению, если холмы разного цвета, то покрасить мост таким образом не удастся. Посчитайте количество таких "плохих" мостов.

**Формат входных данных:**

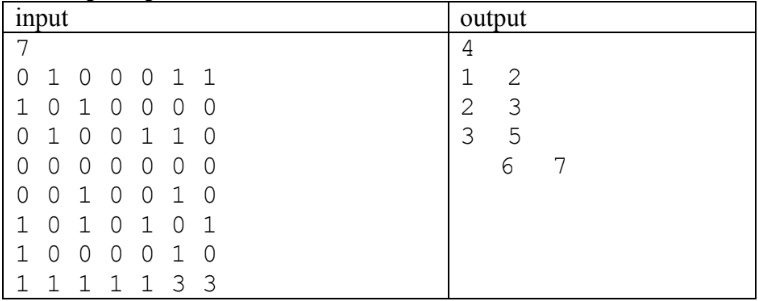
Число холмов N (20 <= N <= 200) необходимо ввести с клавиатуры. Далее сформируйте матрицу смежности, описывающую наличие мостов между холмами (1-мост есть, 0-нет).

Сформируйте вектор "color", содержащий N чисел, обозначающих цвет холмов: 1 - красный; 2 - синий; 3 – зеленый и т.п.

**Формат выходных данных:**

На выходе нужно получить количество "плохих" мостов и номера холмов "плохих" мостов.

**Пример**



**Ход работы**:

Создадим кольцевой граф с числом вершин, случайно выбранном из промежутка от N+10 до (N/10+5)2+5N. В данном случае получился граф с 26 вершинами и 26 ребрами. Он изображен на рисунке 1.



Рисунок 1 – Кольцевой граф

Далее создадим новый граф с таким же числом вершин (26) желтого цвета, добавим 8N красных ребер и к нему еще 10N синих ребер. Результат показан на рисунке 2.

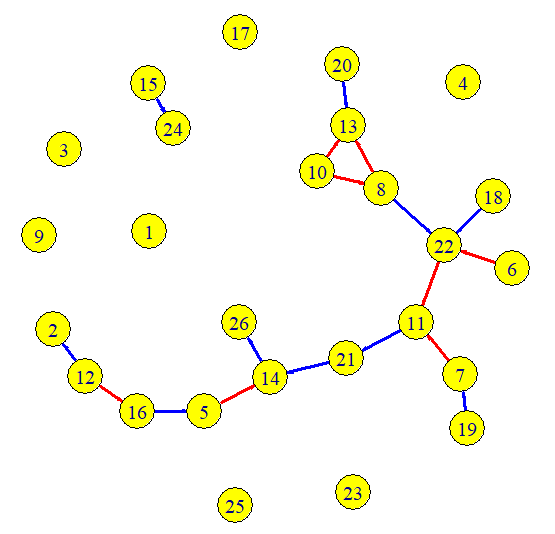


Рисунок 2 – Граф с 8N красных и 10N синих ребер

Проверим существование вершин 2N+23 и 2N+20, 2N+12 и N+15, 2N-1 и N+8, 2N и 2N+1, N+7 и N+13. Какие из них есть, между теми проведем черные ребра.

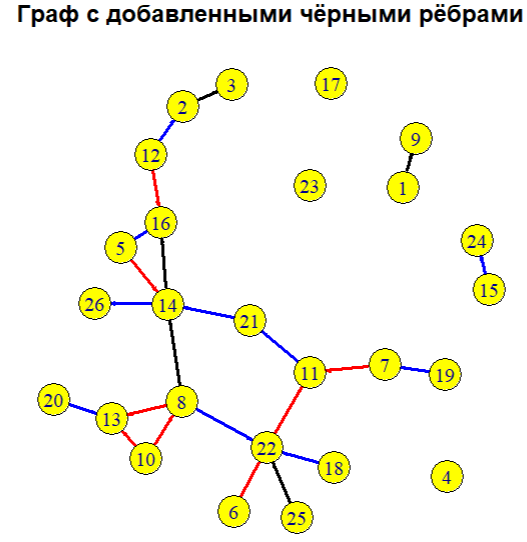


Рисунок 3 – Граф после добавления черных ребер

Выведем соседей вершины N.

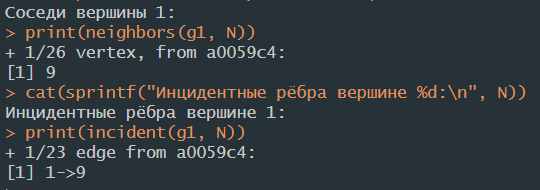


Рисунок 4 – Число соседей и инцидентных ребер

Проверим соединены ли вершины N+10 и N+12. В данном случае нет.

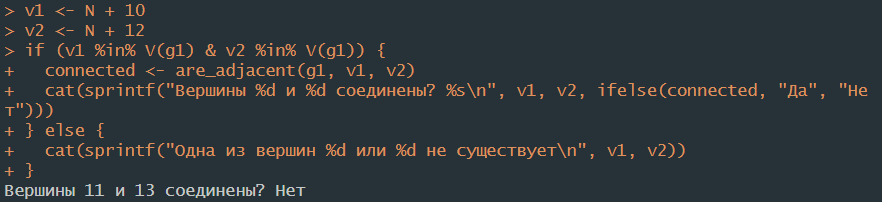


Рисунок 5 – Проверка соединения вершин

Также выведем матрицу смежности.

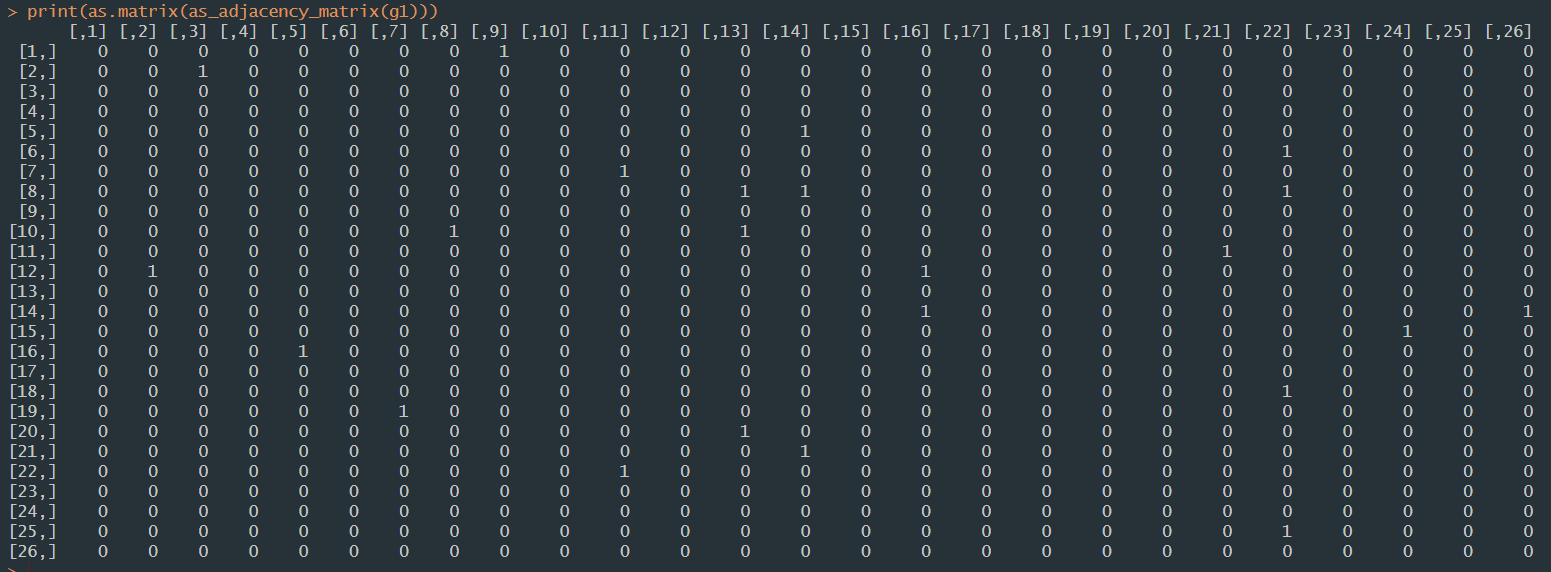


Рисунок 6 – Матрица смежности

Добавим еще одну вершину и соединим ее с той, которая имеет наивысшую степень. Дадим имена всем врешинамм.

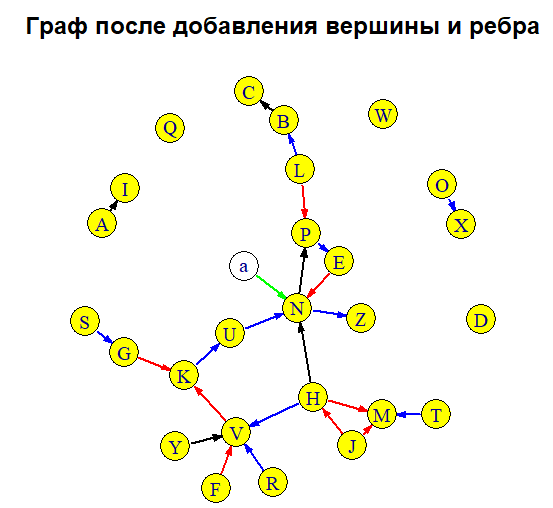


Рисунок 7 – Граф с новой вершиной

Выведем также вершины, степень которых больше 2, но меньше 5.



Рисунок 8 – Вывод вершин

Выведем также обновленную матрицу смежности.

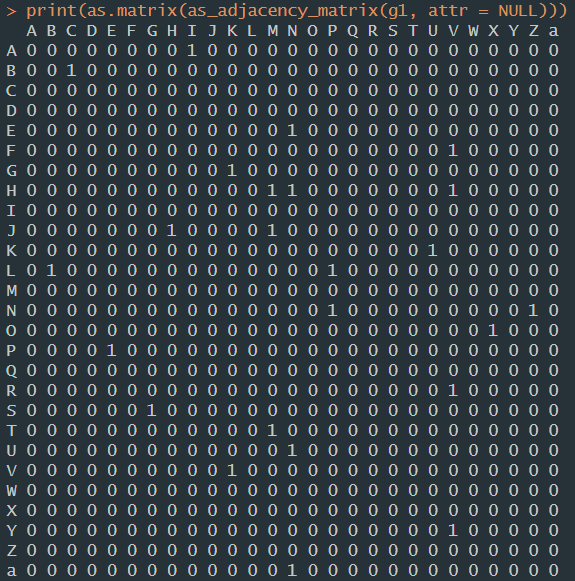


Рисунок 9 – Новая матрица смежности

Рассмотрим алгоритмы размещения вершин. Расположение в круг изображено на рисунке 10, в дерево – на рисунке 11, в виде решетчатого графа с соседями – на рисунке 12.

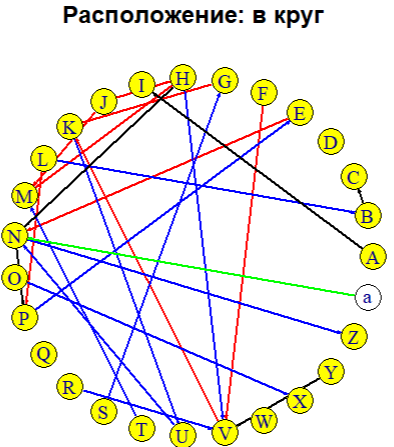


Рисунок 10 – Круговое расположение

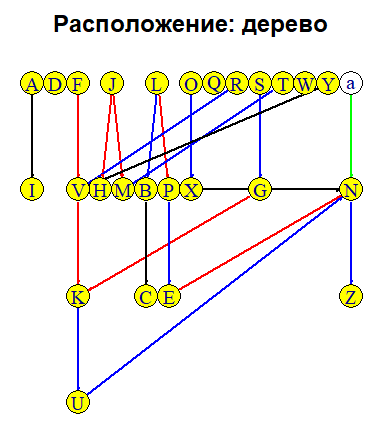


Рисунок 11- Расположение деревом

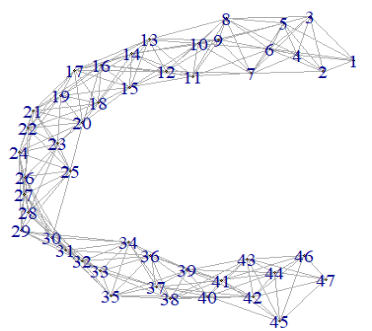


Рисунок 12 – Решетчатый граф

Найдем диаметр графа через метод diameter(), в данном случае 6. Также найдем Длины кратчайших путей от каждой вершины до других.

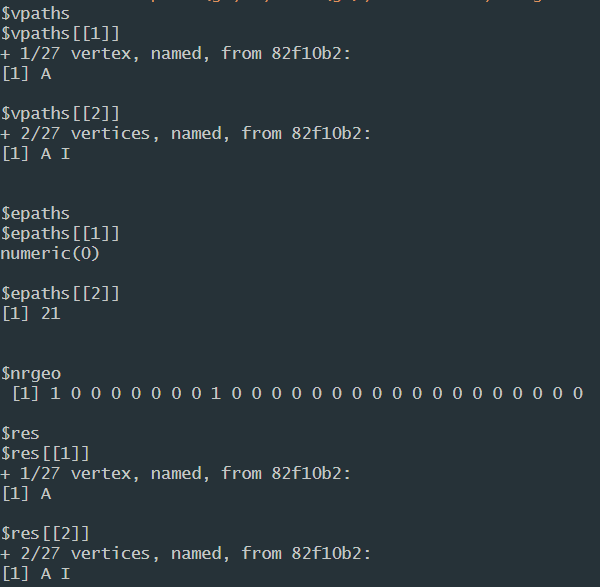


Рисунок 13 – Пример вывода кратчайших путей для вершины 1

Откалибруем величину вершин в соответствии со степенями вершин.

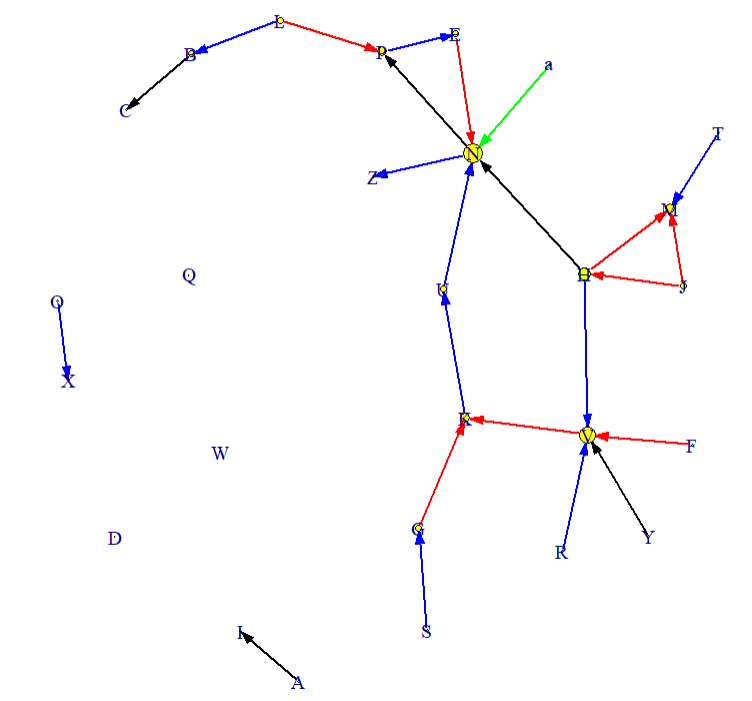


Рисунок 14 – Откалиброванные вершины

Теперь решим задачу по варианту. Сделаем возможность ввода числа холмов. Далее генерируется матрица смежности и каждой вершине присваивается код цвета – красный, синий или зеленый. Потом программа проходится по всем вершинам и сравнивает их цвета, если они не совпадает, то это считается плохим мостом.

Например для 100 мостов программа выводит результат, показанный на рисунке 15.

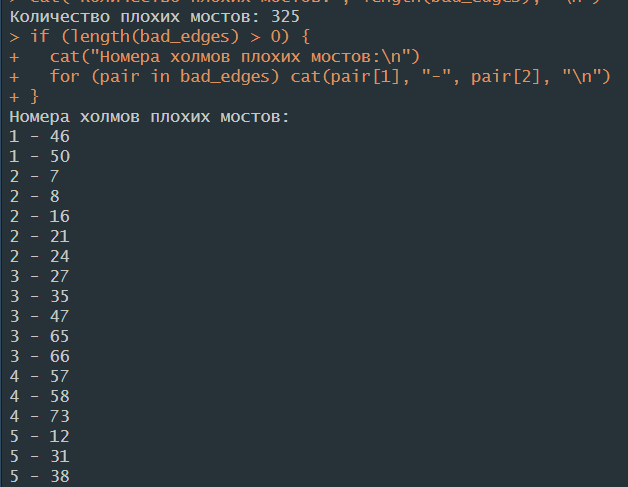


Рисунок 15 – Фрагмент вывода программы

Также выводится граф с раскрашенными вершинами и раскрашенными хорошими мостами, плохие мосты остаются серыми.

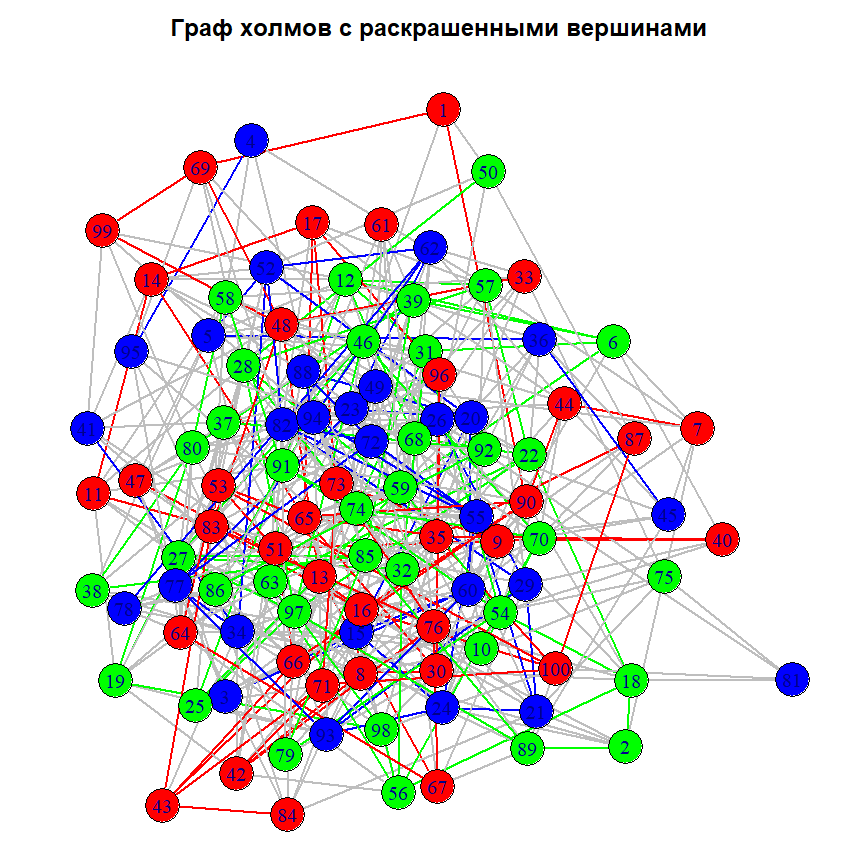


Рисунок 16 – Граф с правильными и неправильными мостами

**Вывод:** проделав лабораторную работу, были освоены основные функции обработки графов.

**Листинг программы:**

library(igraph)

N <- 1

par(mar = c(0, 0, 3, 0))

# 1. Создайте кольцевой граф g со случайным числом вершин G\_size (от N+10 до (N/10+5)^2+5N).

# Выведите число ребер и вершин этого графа. Постройте граф, выведите его матрицу смежности.

min\_size <- N + 10

max\_size <- (N / 10 + 5)^2 + 5 \* N

G\_size <- sample(min\_size:max\_size, 1)

g <- make\_ring(G\_size)

cat("Число вершин:", vcount(g), "\n")

cat("Число рёбер:", ecount(g), "\n")

E(g)$color <- "black"

plot(g, main = "Кольцевой граф", edge.width = 5, vertex.size = 20)

# 2. Создайте граф g1 из пустого графа с числом вершин G\_size желтого цвета. Добавьте ему 8N

# случайных ребер, сформированных из вектора вершин, окрасьте ребра красным цветом,

# нарисуйте граф и выведите его матрицу смежности. Добавьте графу g1 еще 10N случайных ребер,

# сформированных из вектора вершин, окрасьте ребра синим цветом, нарисуйте граф и выведите

# его матрицу смежности.

g1 <- make\_empty\_graph(n = G\_size)

V(g1)$color <- "yellow"

# Добавляем 8N красных рёбер

g1 <- add\_edges(g1, unlist(replicate(8 \* N, sample(1:G\_size, 2), simplify = FALSE)))

E(g1)$color <- "red"

plot(g1, main = "8N красных рёбер", edge.arrow.size = 0.1, edge.width = 3, vertex.size = 15)

print(as.matrix(as\_adjacency\_matrix(g1)))

# Добавляем 10N синих рёбер

g1 <- add\_edges(g1, unlist(replicate(10 \* N, sample(1:G\_size, 2), simplify = FALSE)))

E(g1)$color[(ecount(g1) - 10 \* N + 1):ecount(g1)] <- "blue"

plot(g1, main = "После добавления 10N синих рёбер", edge.arrow.size = 0.1, edge.width = 3, vertex.size = 15)

print(as.matrix(as\_adjacency\_matrix(g1)))

# 3. Добавьте ребра между вершиной 2N+23 и 2N+20, 2N+12 и N+15, 2N-1 и N+8, 2N и 2N+1, N+7 и N+13,

# окрасьте их в черный цвет (предварительно проверьте существуют ли такие вершины – функцией

# %in% либо match, для несуществующих вершин ребра не добавляйте). Нарисуйте граф.

# Выведите соседей N-й вершины, ребра, инцидентные этой вершине.

# Соединены ли вершины N+10 и N+12? Выведите матрицу смежности.

edge\_pairs <- list(

c(2\*N + 23, 2\*N + 20),

c(2\*N + 12, N + 15),

c(2\*N - 1, N + 8),

c(2\*N, 2\*N + 1),

c(N + 7, N + 13)

)

for (pair in edge\_pairs) {

if (all(pair %in% V(g1))) {

g1 <- add\_edges(g1, pair)

E(g1)$color[ecount(g1)] <- "black"

}

}

plot(g1, edge.arrow.size = 0.1, vertex.size = 15, edge.width = 3, main = "Граф с добавленными чёрными рёбрами")

cat(sprintf("Соседи вершины %d:\n", N))

print(neighbors(g1, N))

cat(sprintf("Инцидентные рёбра вершине %d:\n", N))

print(incident(g1, N))

# Проверка соединения вершин N+10 и N+12

v1 <- N + 10

v2 <- N + 12

if (v1 %in% V(g1) & v2 %in% V(g1)) {

connected <- are\_adjacent(g1, v1, v2)

cat(sprintf("Вершины %d и %d соединены? %s\n", v1, v2, ifelse(connected, "Да", "Нет")))

} else {

cat(sprintf("Одна из вершин %d или %d не существует\n", v1, v2))

}

cat("Матрица смежности:\n")

print(as.matrix(as\_adjacency\_matrix(g1)))

# 4. Добавьте еще одну вершину и подключите ее к той, которая имеет наибольшее количество

# связанных с ней узлов. Присвойте имена всем вершинам (например, буквы в алфавитном порядке –

# используйте заглавные и, если не хватит, строчные буквы). Выведите матрицу смежности.

# Выберите вершины, для которых значение связности меньше 5 и больше 2.

g1 <- add\_vertices(g1, 1)

# Найдём вершину с максимальной степенью

degrees <- degree(g1)

max\_deg\_vertex <- which.max(degrees)

# Подключим новую вершину (последняя) к этой вершине

new\_vertex\_id <- vcount(g1)

g1 <- add\_edges(g1, c(new\_vertex\_id, max\_deg\_vertex))

E(g1)$color[ecount(g1)] <- "green"

generate\_names <- function(n) {

base <- c(LETTERS, letters)

if (n <= length(base)) return(base[1:n])

names <- base

while(length(names) < n) names <- c(names, paste0(rep(base, each=length(base)), base))

names[1:n]

}

V(g1)$name <- generate\_names(vcount(g1))

cat("Матрица смежности:\n")

print(as.matrix(as\_adjacency\_matrix(g1, attr = NULL)))

# Вершины со степенью >2 и <5

filtered\_vertices <- V(g1)[degree(g1) > 2 & degree(g1) < 5]

cat("Вершины со степенью >2 и <5:", paste(filtered\_vertices$name, collapse = ", "), "\n")

plot(g1, edge.arrow.size = 0.1, edge.width = 2, vertex.label = V(g1)$name, main = "Граф после добавления вершины и ребра")

# 5. Испробуйте алгоритмы размещения Вашего графа (in\_circle, as\_tree, lattice)

# In circle (в круг)

plot(g1, layout = layout\_in\_circle, edge.arrow.size = 0.1, edge.width = 2, vertex.label = V(g1)$name, main = "Расположение: в круг")

# Как дерево (если граф можно представить как дерево)

plot(g1, layout = layout\_as\_tree,

edge.arrow.size = 0.1, edge.width = 2, vertex.label = V(g1)$name,

main = "Расположение: дерево")

# Создание линейного одномерного решётчатого графа с соседями (nei = 5)

g2 <- g1

g2 <- make\_lattice(length = 47, dim = 1, nei = 5, periodic = FALSE)

plot(g2, vertex.size = 2, layout = layout\_with\_kk, edge.arrow.size = 0.1)

# 6. Выполните измерение диаметра графа g1, выведите список самых коротких путей для каждой

# вершины и откалибруйте величины вершин согласно их степеней.

graph\_diameter <- diameter(g1)

cat("Диаметр графа:", graph\_diameter, "\n\n")

all\_paths\_list <- lapply(V(g1), function(v) {

all\_shortest\_paths(g1, v, to = V(g1), mode = "all", weights = NULL)

})

deg <- degree(g1, mode='all')

plot(g1, edge.arrow.size=0.1, vertex.size=deg, edge.width = 2)

# Вариант 6. В Банановой республике очень много холмов, соединенных мостами. На химическом заводе

# произошла авария, в результате чего испарилось экспериментальное удобрение "зован". На

# следующий день выпал цветной дождь, причем он прошел только над холмами. В некоторых

# местах падали красные капли, в некоторых - синие, а в остальных - зеленые, в результате

# чего холмы стали соответствующего цвета. Президенту Банановой республики это

# понравилось, но ему захотелось покрасить мосты между вершинами холмов так, чтобы

# мосты были покрашены в цвет холмов, которые они соединяют. К сожалению, если холмы

# разного цвета, то покрасить мост таким образом не удастся. Посчитайте количество таких

# "плохих" мостов.

# Формат входных данных:

# Число холмов N (20 <= N <= 200) необходимо ввести с клавиатуры. Далее сформируйте

# матрицу смежности, описывающую наличие мостов между холмами (1-мост есть, 0-нет).

# Сформируйте вектор "color", содержащий N чисел, обозначающих цвет холмов: 1 -

# красный; 2 - синий; 3 – зеленый и т.п.

# Формат выходных данных:

# На выходе нужно получить количество "плохих" мостов и номера холмов "плохих" мостов.

Num <- as.integer(readline("Введите число холмов (20-200): "))

if (Num < 20 || Num > 200) stop("Число холмов должно быть от 20 до 200")

p\_edge <- 0.1

upper\_tri <- matrix(rbinom(Num \* Num, 1, p\_edge), nrow = Num, ncol = Num)

upper\_tri[lower.tri(upper\_tri, diag = TRUE)] <- 0

adj\_matrix <- upper\_tri + t(upper\_tri)

cat("Сгенерированная матрица смежности:\n")

print(adj\_matrix)

color <- sample(1:3, Num, replace = TRUE)

print(color)

bad\_edges <- which(adj\_matrix == 1 & outer(color, color, `!=`), arr.ind = TRUE)

bad\_edges <- bad\_edges[bad\_edges[,1] < bad\_edges[,2], ]

cat("Количество плохих мостов:", nrow(bad\_edges), "\n")

if (nrow(bad\_edges) > 0) {

cat("Номера холмов плохих мостов:\n")

for (k in 1:nrow(bad\_edges)) {

cat(bad\_edges[k, 1], "-", bad\_edges[k, 2], "\n")

}

}

g <- graph\_from\_adjacency\_matrix(adj\_matrix, mode = "undirected")

V(g)$color <- c("red", "blue", "green")[color]

E(g)$color <- ifelse(color[ends(g, E(g))[,1]] == color[ends(g, E(g))[,2]], V(g)$color[ends(g, E(g))[,1]], "gray")

plot(g, vertex.size = 10, edge.width = 2,main = "Граф холмов с раскрашенными вершинами")