**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ России**

**Федеральное государственное автономное образовательное учреждение высшего профессионального образования**

**«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»**

**Инженерно-технологическая Академия**

**Институт компьютерных технологий и информационной безопасности**

**Кафедра Систем Автоматизированного Проектирования**

**Отчет по лабораторной работе № 3**

**на тему: «Основные положения теории эволюционного моделирования»**

по дисциплине «Методы машинного обучения»

Выполнили:

студенты группы КТбо4-4

Батагов С.С.\_\_\_\_\_\_\_\_\_\_\_\_\_

Москаленко М.В.\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил:

доцент кафедры САПР

Лебедев О.Б. \_\_\_\_\_\_\_\_\_\_\_\_\_

Таганрог 2022

Содержание

[ВВЕДЕНИЕ 2](#_Toc122197185)

[1 ТЕОРЕТИЧЕСКАЯ ЧАСТЬ 3](#_Toc122197186)

[1.1 Типы хромосом 3](#_Toc122197187)

[1.2 Стратегии создания начальной популяции 5](#_Toc122197188)

[2 ПРАКТИЧЕСКАЯ ЧАСТЬ 6](#_Toc122197189)

[2.1 Главная функция 6](#_Toc122197190)

[2.2 Функция создания начальной популяции 6](#_Toc122197191)

[2.3 Функция развития популяции 7](#_Toc122197192)

[2.4 Функция для синтеза видов хромосом 8](#_Toc122197193)

[ЗАКЛЮЧЕНИЕ 9](#_Toc122197194)

[Приложение А 10](#_Toc122197195)

ВВЕДЕНИЕ

***Цель работы:***

Ознакомиться с основными терминами и положениями теории эволюционного моделирования. Научиться создавать начальные популяции альтернативных решений. Уметь кодировать и декодировать альтернативные решения (особи, хромосомы) задач.

***Задания к лабораторной работе***

Написать программу, реализующую различные стандартные виды стратегий создания начальной популяции. Продемонстрировать и объяснить работу программы на примерах.

Написать программу, реализующую различные варианты развития популяции: микро-, макро- и метаэволюцию. Продемонстрировать и объяснить работу программы на примерах.

Написать программу, позволяющую синтезировать различные виды хромосом (бинарные, числовые, векторные и т.д.). Продемонстрировать и объяснить работу программы на примерах.

Написать программу для представления различных типов данных, их кодирования и декодирования. Продемонстрировать и объяснить работу программы на примерах.

На основе полученных знаний предложить новую модификацию стратегии создания начальной популяции. Написать программу, реализующую разработанную стратегию. Продемонстрировать и объяснить работу программы на примерах.

# ТЕ­ОРЕТИЧЕСКАЯ ЧАСТЬ

## Типы хромосом

В настоящее время установлено, что наследственные факторы – гены являются специфическими молекулами ДНК или функционально обособленными участками таких молекул и что устойчивая передача генов от родителей к потомкам зависит в первую очередь от способности молекул ДНК к авторепродукции.

Участки молекул, занимаемые отдельными генами, называются локусами. Для многих локусов известно не одно, а несколько устойчивых состояний генов. Такие состояния генов называют аллеломорфными генами, аллеломорфами, или просто аллелями.

По методам представления генов хромосомы можно условно разделить на три типа:

Двоичные хромосомы — это такие хромосомы, гены которых могут принимать только два значения {0; 1}

В подсистеме генетического поиска двоичным хромосомам соответствует тип данных GA­Binary­Chromosome, в котором используется побитовая упаковка, т.е. на каждый ген отводится один бит. В некоторых случаях используется три значения {0; 1, \*}. Здесь \* определяется как символ «не имеет значения» 1 или 0.

Числовые хромосомы– это такие хромосомы, гены которых могут принимать целочисленные значения в заданном интервале. В зависимости от диапазона допустимых значений генов (аллелей) можно предусмотреть возможность использования разных типов целых чисел для представления генов: однобайтовое целое (8 бит), позволяющее представить диапазон чисел [0, 255], или [-128, +127]; двухбайтовое целое (16 бит), позволяющее представить диапазон чисел [0, 65 535], или [-32 768, +32 767]; четырехбайтовое целое (32 бита), позволяющее представить диапазон чисел [0, 4 294 967 295], или [-2 147 483 648, +2 147 483 647]. Такое представление позволяет существенно сократить затраты памяти и как следствие – повысить эффективность реализуемых алгоритмов.

Стоит отметить, что числовые хромосомы подразделяются ещё на два типа:

Гомологичные хромосомы — это хромосомы, имеющие общее происхождение, морфологически и генетически сходные, и поэтому не образующие недопустимых решений при применении стандартных генетических операторов. В гомологичных числовых хромосомах каждый ген может принимать целые значения в заданном числовом интервале, при этом для различных генов могут быть заданы различные интервалы [ai, bi), где , n – число генов в хромосоме.

Негомологичные хромосомы — это такие хромосомы, гены которых могут принимать значения в заданном интервале. При этом интервал одинаков для всех генов, но в хромосоме не может быть двух генов с одинаковым значением. Для негомологичных хромосом применяют различные специальные генетические операторы, не создающие недопустимых решений.

Векторные хромосомы — это такие хромосомы, гены которых представляют собой векторы целых чисел. При этом аллели генов обладают свойствами негомологичной числовой хромосомы, т.е. числа вектора могут принимать значения в заданном интервале, и вектор не может содержать двух одинаковых чисел. Тем не менее, хотя гены в векторных хромосомах негомологичны, сами хромосомы являются гомологичными и им соответствует тип данных GA­Vectorial­Chromosome.

## Стратегии создания начальной популяции

Эффективность генетического алгоритма – степень реализации запланированных действий алгоритма и достижение требуемых значений целевой функции. Эффективность во многом определяется структурой и составом начальной популяции. При создании начального множества решений происходит формирование популяции на основе четырех основных принципов:

«одеяла» — генерируется полная популяция, включающая все возможные решения в некоторой заданной области;

«дробовика» — подразумевает случайный выбор альтернатив из всей области решений данной задачи.

«фокусировки» — реализует случайный выбор допустимых альтернатив из заданной области решений данной задачи.

Комбинирования — состоит в различных совместных реализациях первых трех принципов.

Отметим, что популяция обязательно является конечным множеством. Развитие популяции происходит на микро-, макро- и метауровнях эволюции.

Микроэволюция — это создание одной хромосомы и реализация на ее основе эволюционного поиска.

Макроэволюция — это создание одной популяции и реализация на ней эволюционного поиска.

Метаэволюция — это создание множества популяций и реализация на нем эволюционного поиска.

Элементы в ГА часто называют родителями. Родители выбираются из популяции на основе заданных правил, а затем смешиваются (скрещиваются) для производства «детей» (потомков). Дети и родители в результате генерации, т.е. одного цикла (подцикла) эволюции, создают новую популяцию. Генерация, то есть процесс реализации одной итерации алгоритма, называется поколением.

# ПРАКТИЧЕСКАЯ ЧАСТЬ

Для выполнения данной лабораторной работы были разработаны программы с упрощённым эволюционным поиском.

Для написания программ использовался язык JavaScript и библиотека readline-sync для ввода символов в консоль. Исходный код программы представлен в приложении А.

### Главная функция

Главная функция, представленная на Рисунке 1, позволяет выбрать действие, которое необходимо выполнить, например произвести синтез различных видов хромосом.

Изображение выглядит как текст

Автоматически созданное описание

### Функция создания начальной популяции

Функция создания начальной популяции (Рисунок 2) состоит из оператора switch с выбором и вызовом методов решения необходимой задачи.

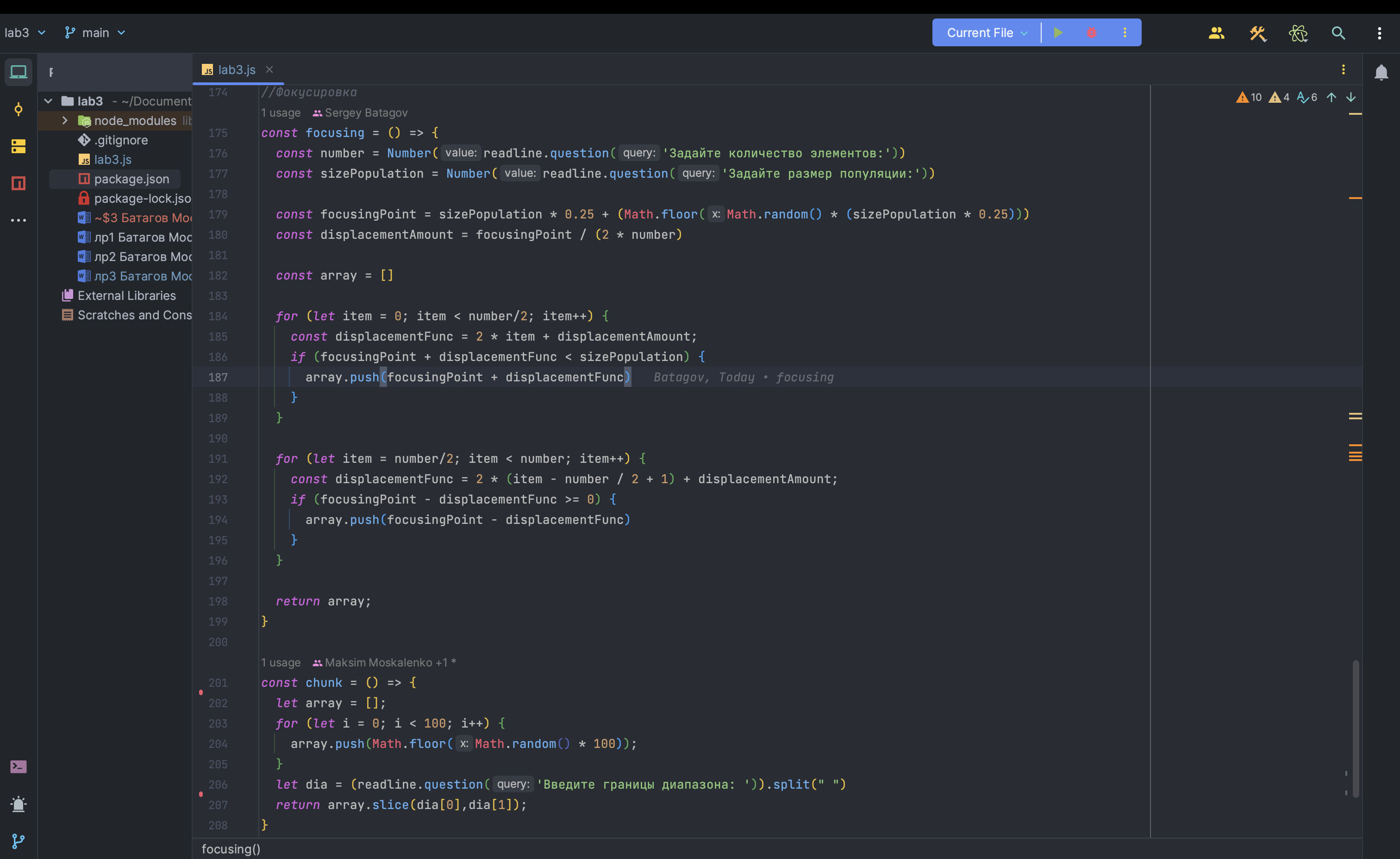
Изображение выглядит как текст

Автоматически созданное описание

На Рисунках 3–4 представлены реализованные методы.

Изображение выглядит как текст

Автоматически созданное описание



Принцип одеяла заключается в генерации данных с постепенно увеличивающимся значениями. То есть генерируется полная популяция, включающая все возможные решения в некоторой заданной области.

Принцип «дробовика» – подразумевает случайный выбор альтернатив из всей области решений данной задачи.

«Фокусировка» – реализует случайный выбор допустимых альтернатив из заданной области решений данной задачи.

Комбинированный принцип создания популяции. Создаётся массив значений, в котором выбирается нужный диапазон значений.

На Рисунках 5–8 представлен результат работы программы.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

### Функция развития популяции

Для данной программы был упрощён эволюционный поиск, так как элементы, которые участвуют в генетическом алгоритме, не были рассмотрены, поэтому в данном случае результат является определение максимального значения в хромосоме, определение хромосомы, имеющую максимальную длину, и определение популяции, с большим количеством хромосом.

На Рисунке 6, представлен код программы, которая состоит из 3 функций, каждая из которых отвечает за свой вариант развития популяции.

Каждая функция описывает свой вариант развития, однако принцип работы остаются тем же. После создания популяции, определяется упрощённый эволюционный путь, и на его основе создаётся новый массив данных, исходя из ранее выбранного варианта развития. На Рисунке 7-/// представлены результаты работы программы.

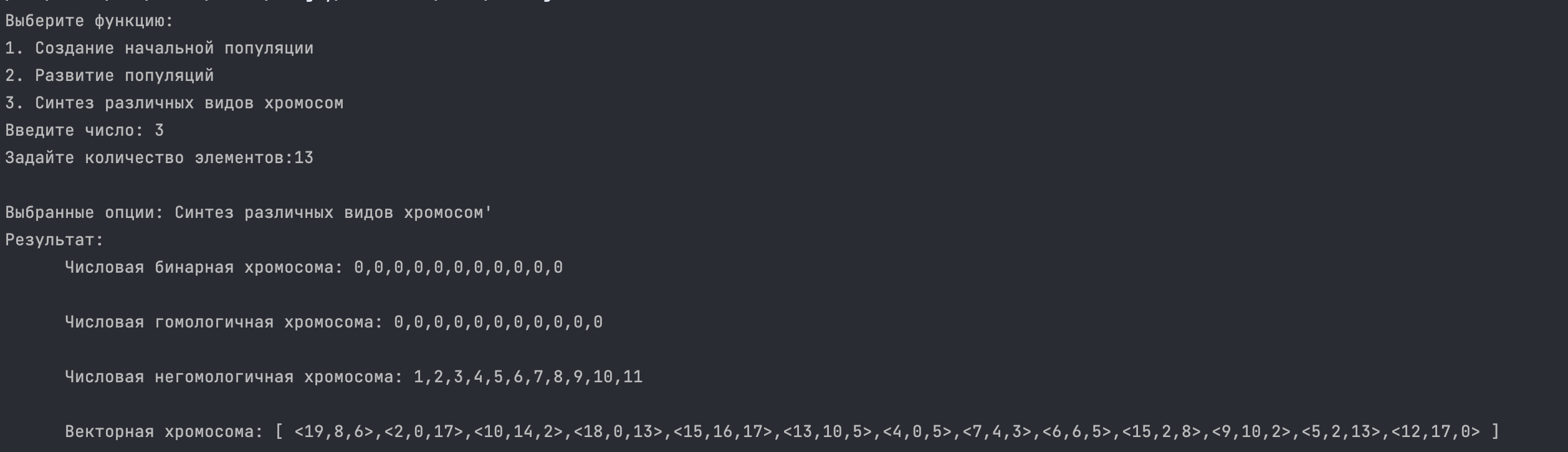
### Функция для синтеза видов хромосом

Данная функция основана на Разработанная программа основана на теоретическом материале, приведённом в методическом пособии. Существует несколько видов хромосом: числовая (гомологические, негомологические), векторные, бинарные. На Рисунке 13 представлен код программы для генерации представленных видов хромосом.

Изображение выглядит как текст

Автоматически созданное описание

Результат работы программы представлен на Рисунке 13



ЗАКЛЮЧЕНИЕ

В данной лабораторной работе были рассмотрены основные типы хромосом индивидуумов и создание и развитие популяции с теоретической точки зрения. Результаты проведенного исследования помогут в выполнении следующих лабораторных работ.

Приложение А

*// Реализация методов создания новой популяции  
let* readline = require('readline-sync');  
  
*const* SelectMethod = () => {  
 *const* method = Number(readline.question(  
 'Выберите функцию: \n' +  
 '1. Создание начальной популяции \n' +  
 '2. Развитие популяций \n' +  
 '3. Синтез различных видов хромосом \n' +  
 'Введите число: '  
 ))  
 *return* ResultProgram(method);  
}  
  
*const* ResultProgram = (method) => {  
 *switch* (method){  
 *case* 1:  
 *return* CreatePopulation()  
 *case* 2:  
 *return* RisePopulation()  
 *case* 3:  
 *return* SynthesisСhromosomeSpecies()  
 *default*:  
 console.log('Проверьте введенные данные')  
 }  
}  
  
*const* CreatePopulation = () => {  
 *switch* (  
 Number(readline.question('Создание начальной популяции. Выберите метод:\n' +  
 '1. "Одеяло" \n' +  
 '2. "Дробовик" \n' +  
 '3. "Фокусировка" \n' +  
 '3. "Комбинированный" \n' +  
 'Введите число: '))  
 ){  
 *case* 1:  
 *return* {  
 selectedMethod: "Создание начальной популяции. Метод 'Одеяло'",  
 result: blanket()  
 }  
 *case* 2:  
 *return* {  
 selectedMethod: "Создания начальной популяции. Метод 'Дробовик'",  
 result: shotgun()  
 }  
 *case* 3:  
 *return* {  
 selectedMethod: "Создания начальной популяции. Метод 'Фокусировка'",  
 result: focusing()  
 }  
 *case* 4:  
 *return* {  
 selectedMethod: "Создания начальной популяции. Метод 'Комбинированный'",  
 result: chunk()  
 }  
 *default*:  
 console.log('Проверьте введенные данные')  
 }  
}  
  
*const* RisePopulation = () => {  
 *switch* (Number(readline.question('Развитие популяций:\n' +  
 '1. "micro" \n' +  
 '2. "macro" \n' +  
 '3. "meta" \n' +  
 'Введите число: '))  
 ) {  
 *case* 1:  
 *let* array = [];  
 *let* countMicro = Math.floor(Math.random() \* 25);  
 *for* (*let* i = 0; i < countMicro; i++) {  
 array.push(Math.floor(Math.random() \* 25));  
 }  
 *return* {  
 selectedMethod: "Развитие популяций. Метод 'micro'",  
 result: array  
 };  
 *case* 2:  
 *let* tmp = [];  
 *let* countMacro = Math.floor(Math.random() \* 5);  
 *let* countMacro2 = Math.floor(Math.random() \* 25);  
 *for* (*let* i = 0; i < countMacro; i++) {  
 *for* (*let* j = 0; j < countMacro2; j++) {  
 tmp.push(Math.floor(Math.random() \* 25));  
 }  
 }  
 *return* {  
 selectedMethod: "Развитие популяций. Метод 'micro'",  
 result: tmp  
 };  
 *case* 3:  
 *let* mettmp = [];  
 *let* countMeta = Math.floor(Math.random() \* 5);  
 *let* countMeta2 = Math.floor(Math.random() \* 5);  
 *let* countMeta3 = Math.floor(Math.random() \* 25);  
 *for* (i = 0; i < countMeta; i++) {  
 *for* (j = 0; j < countMeta2; j++) {  
 *for* (k = 0; k < countMeta3; k++) {  
 mettmp.push(Math.floor(Math.random() \* 25));  
 }  
 }  
 }  
 *return* {  
 selectedMethod: "Развитие популяций. Метод 'micro'",  
 result: mettmp  
 };  
 }  
}  
  
*const* SynthesisСhromosomeSpecies = () => {  
 *const* number = Number(readline.question('Задайте количество элементов:'))  
 *const* binarn = [];  
 *const* numberGomo = [];  
 *const* numberGetero = [];  
 *const* lenArr = (Math.floor(Math.random() \* number))  
  
 *for* (*let* item = 0; item < lenArr; item++) {  
 binarn.push(Math.floor(Math.random()));  
 }  
  
 *for* (*let* item = 0; item < lenArr; item++) {  
 numberGomo.push(Math.floor(Math.random()));  
 }  
  
 *for* (*let* item = 0; item < lenArr; item++) {  
 numberGetero.push(item + 1);  
 }  
 *const* randomNumber = () => {  
 *return* Math.floor(Math.random() \* 20)  
 }  
 *const* vector = [];  
 *for* (*let* item = 0; item < number; item++) {  
 vector.push([randomNumber(), randomNumber(), randomNumber()]);  
 }  
  
 *return* {  
 selectedMethod: "Синтез различных видов хромосом'",  
 result: `  
 Числовая бинарная хромосома: ${binarn} \n  
 Числовая гомологичная хромосома: ${numberGomo} \n  
 Числовая негомологичная хромосома: ${numberGetero} \n  
 Векторная хромосома: [ ${vector.map(item=>`<` + item + '>')} ] \n  
 `  
 }  
}  
  
*//Одеяло  
const* blanket = () => {  
 *const* number = Number(readline.question('Задайте количество элементов: '))  
 *const* interval = Number(readline.question('Задайте диапазон изменения значений (одно число): '))  
 *const* array = []  
 *for* (*let* item = 0; item < number-1; item++) {  
 array.push(interval/(2\*number)+interval/number\*item)  
 }  
 *return* array  
}  
  
*//Дробовик  
const* shotgun = () => {  
 *const* number = Number(readline.question('Задайте количество элементов:'))  
 *const* arrayRandomNumber = []  
 *for* (*let* item = 0; item < number; item++) {  
 arrayRandomNumber.push(Math.floor(Math.random() \* 100))  
 }  
 *const* array = []  
 *const* count = (Math.floor(Math.random() \* arrayRandomNumber.length));  
 *for* (*let* item = 0; item < count; item++) {  
 array.push(arrayRandomNumber[(Math.floor(Math.random() \* arrayRandomNumber.length))])  
 }  
 *return* array;  
}  
  
*//Фокусировка  
const* focusing = () => {  
 *const* number = Number(readline.question('Задайте количество элементов:'))  
 *const* sizePopulation = Number(readline.question('Задайте размер популяции:'))  
  
 *const* focusingPoint = sizePopulation \* 0.25 + (Math.floor(Math.random() \* (sizePopulation \* 0.25)))  
 *const* displacementAmount = focusingPoint / (2 \* number)  
  
 *const* array = []  
  
 *for* (*let* item = 0; item < number/2; item++) {  
 *const* displacementFunc = 2 \* item + displacementAmount;  
 *if* (focusingPoint + displacementFunc < sizePopulation) {  
 array.push(focusingPoint + displacementFunc)  
 }  
 }  
  
 *for* (*let* item = number/2; item < number; item++) {  
 *const* displacementFunc = 2 \* (item - number / 2 + 1) + displacementAmount;  
 *if* (focusingPoint - displacementFunc >= 0) {  
 array.push(focusingPoint - displacementFunc)  
 }  
 }  
  
 *return* array;  
}  
  
*const* chunk = () => {  
 console.log("Комбинированный прицнип выбора конкретного участка хромосомы");  
 *let* array = [];  
 *for* (*let* i = 0; i < 100; i++) {  
 array.push(Math.floor(Math.random() \* 100));  
 }  
 *let* dia = (readline.question('Введите границы диапазона: ')).split(" ")  
 console.log(array.slice(dia[0],dia[1]))  
 *return* array.slice(dia[0],dia[1]);  
}  
  
*const* { selectedMethod, result } = SelectMethod();  
console.log(`\nВыбранные опции: ${selectedMethod}`)  
console.log(`Результат: ${result}`)