АНОТАЦІЯ

Кваліфікаційна робота включає пояснювальну записку (73 сторінки, 15 рисунків., 21 псевдокод, 11 таблиць, 5 графіків, 4 додатки).

Метою даного дипломного проекту є створення модифікації генетичного алгоритму визначення хроматичного числа графа.

В роботі розглянуто та проаналізовано етапи роботи класичного генетичного алгоритму. Розібрано різні види селекції та схрещування, найбільш поширені методи кодування даних, існуючі умови завершення роботи алгоритму. З усіх можливих варіантів вибрано такі, що найбільш підходять до вирішення задачі знаходження хроматичного числа, вибір обґрунтовано.

В програмі передбачена можливість: задавати кількість вершин та ребер в графі; створювати випадкові графи; задавати кількість початкової популяції, кількість батьків, що мутують при кожній ітерації, кількість батьків для кросоверу; зміни штрафу за некоректний колір вершини та штрафу за кожен новий колір; вибирати функцію мутації та схрещування; вказати бажане хроматичне число та перевірити його на конфліктність. Розроблено такі тестові моделі, що найбільш підходять для демонстрації функцій алгоритму. Проаналізована швидкість та точність в залежності від параметрів. Підібрано оптимальний набір функцій мутації та кросоверу, їх послідовність та ознака початку дії кожної функції.

Для реалізації генетичного алгоритму було обрано мову програмування Python.

Даний алгоритм може бути використаний для складання розкладів, розподілу частот, регістрів у мікропроцесорах, обчислення похідних, розпаралелювання обчислень за числовими методами.

***Ключові слова*:** алгоритм, генетичний алгоритм, хроматичне число, граф, мутація, кросовер, фітнес-функція, Python.

ABSTRACT

The qualifying work includes an explanatory note (73 pages, 15 figures, 21 pseudocodes, 11 tables, 5 charts, 4 annexes).

The purpose of this diploma project is to create a modification of the genetic algorithm for determining the chromatic number of a graph.

The stages of the work of the classical genetic algorithm are considered and analyzed in this work. Different types of breeding and crossover are analyzed, the most common methods of encoding data, the existing conditions for the completion of the algorithm. Of all possible options, those that are most suited to the solution of the problem of finding a chromatic number are selected, the choice is justified.

The program provides the ability to: set the number of vertices and edges in the graph; create random graphs; set the number of initial population, the number of parents, mutate each iteration, the number of parents for the crossover; change the fine for the wrong color of the summit and fine for each new color; select mutation and crossing function; specify the desired chromatic number and check it for conflicts. The following test models are developed that are most suitable for demonstrating the functions of the algorithm. The speed and accuracy are analyzed, depending on the parameters. The optimal set of mutation and crossover functions, their sequence and a sign of the beginning of each function's operation are selected.

To implement the genetic algorithm was selected the Python programming language.

This algorithm can be used for scheduling, frequency distribution, registers in microprocessors, derivative calculations, parallelization of computations by numerical methods.

Key words: algorithm, genetic algorithm, chromatic number, graph, mutation, crossover, fitness function, Python.