ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| кандидат технических наук |  |  |  | П.А. Степанов |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №5  **Решение логических задач** |
| по дисциплине: Функциональное и логическое программирование |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. | 4134к |  |  |  | Костяков Н.А. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

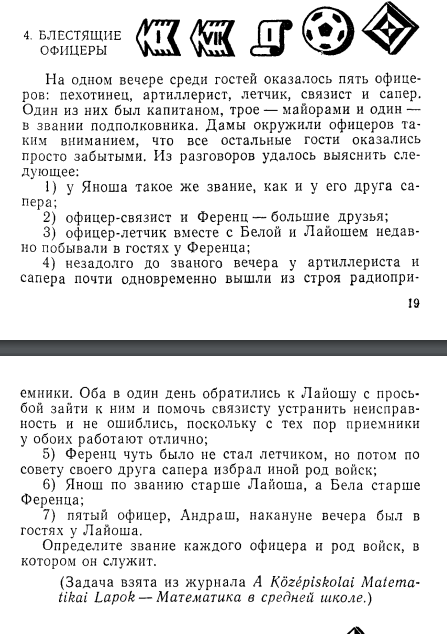
Санкт-Петербург

2023

# Цель работы

Составить программу для решения одной из логических задач № 1–4, 19, 20, 27, 31, 33, 40–43, 56, 58, 59, 61, 62, 64–67, 76–85, условия которых приведены в книге: Бизам Д., Герцер Я. Игра и логика. — М.: Мир, 1975 г.

# Задача 4



# Листинг программы

same\_rank(yanosh, saper).

same\_rank(saper, yanosh).

friend(saper, yanosh).

friend(yanosh, saper).

friend(linkman, ferenc).

friend(ferenc, linkman).

friend(saper, ferenc).

friend(ferenc, saper).

friend(andrash, layosh).

not\_pilot(ferenc).

not\_pilot(bela).

not\_pilot(layosh).

not\_linkman(ferenc).

not\_linkman(layosh).

not\_artilerian(layosh).

not\_saper(layosh).

not\_saper(ferenc).

grater(yanosh, layosh).

grater(bela, ferenc).

can\_be\_colonel(X):- not(same\_rank(X, Y)).

lower\_rank(X):- not(grater(X, Y)).

colonel(X):- can\_be\_colonel(X), not(lower\_rank(X)).

major(X):- (same\_rank(X, Y)); grater(Z, X), colonel(Z).

captain(X):- not(same\_rank(X, Y)), grater(Z,X), major(Z).

pilot(X):- not(not\_pilot(X)), not(marine(X)).

saper(X):- not(not\_saper(X)), not(grater(X, Z)), not(pilot(X)), not(marine(X)), not(artilerian(X)).

linkman(X):- not(saper(X)), not(pilot(X)), not(not\_linkman(X)).

marine(X):- not\_saper(X), not\_artilerian(X), not\_linkman(X), not\_pilot(X).

artilerian(X):- not(saper(X)), not(pilot(X)), not(not\_artilerian(X)), not(marine(X)), not(linkman(X)).

check\_name(Name, Predicate) :-

  current\_predicate(Predicate/1), % проверяем, что предикат существует

  call(Predicate, Name), % вызываем предикат с именем в качестве аргумента

  write(Name), write(' is a '), write(Predicate), nl. % выводим результат в консоль

  % создание списка имен и предикатов

names([bela, ferenc, layosh, andrash, yanosh]).

predicates([colonel, major, captain, artilerian, saper, linkman, marine, pilot]).

% цикл по спискам

loop:-

  names(Names), % получаем список имен

  predicates(Predicates), % получаем список предикатов

  member(Name, Names), % выбираем имя из списка

  member(Predicate, Predicates), % выбираем предикат из списка

  check\_name(Name, Predicate),

  fail.

# Вывод программы

